

Advancing Vision and Language Models

through Commonsense Knowledge, Efficient
Adaptation, and Transparency



Zhi Zhang

Advancing Vision and Language Models through Commonsense Knowledge, Efficient Adaptation, and Transparency



UNIVERSITY OF AMSTERDAM



INSTITUTE FOR LOGIC, LANGUAGE AND COMPUTATION

Advancing Vision and Language Models through Commonsense Knowledge, Efficient Adaptation and Transparency

Zhi Zhang

Advancing Vision and Language Models through Commonsense Knowledge, Efficient Adaptation and Transparency

ILLC Dissertation Series DS-2025-07



INSTITUTE FOR LOGIC, LANGUAGE AND COMPUTATION

For further information about ILLC-publications, please contact

Institute for Logic, Language and Computation
Universiteit van Amsterdam
Science Park 107
1098 XG Amsterdam
phone: +31-20-525 6051
e-mail: illc@uva.nl
homepage: <http://www.illc.uva.nl/>

The research for/publication of this doctoral thesis received financial assistance from the China Scholarship Council (CSC) which is subsidized by China's Ministry of Education.

Copyright © 2025 by Zhi Zhang

Cover design by Zhi Zhang.

Printed and bound by Ipskamp Printing.

ISBN: 978-94-6473-922-0

Advancing Vision and Language Models through Commonsense Knowledge,
Efficient Adaptation, and Transparency

ACADEMISCH PROEFSCHRIFT

ter verkrijging van de graad van doctor

aan de Universiteit van Amsterdam

op gezag van de Rector Magnificus

prof. dr. ir. P.P.C.C. Verbeek

ten overstaan van een door het College voor Promoties ingestelde commissie,

in het openbaar te verdedigen in de Aula der Universiteit

op vrijdag 3 oktober 2025, te 11.00 uur

door Zhi Zhang

geboren te Guizhou

Promotiecommissie

<i>Promotor:</i>	prof. dr. R. Fernández Rovira	Universiteit van Amsterdam
<i>Copromotor:</i>	dr. E.V. Shutova	Universiteit van Amsterdam
<i>Overige leden:</i>	dr. R.A. Bernardi	University of Trento
	prof. dr. S. Belongie	University of Copenhagen
	dr. M.A.F. Lewis	Universiteit van Amsterdam
	prof. dr. F. Roelofsen	Universiteit van Amsterdam
	prof. dr. C. Monz	Universiteit van Amsterdam

Faculteit der Natuurwetenschappen, Wiskunde en Informatica

Contents

Acknowledgments	ix
1 Introduction	1
1.1 Part I: Commonsense Knowledge Enhanced Multimodal Reasoning .	2
1.2 Part II: Efficient Adaptation	3
1.2.1 PartII.I: Parameter-efficient fine-tuning for task-specific adaptation	4
1.2.1 PartII.II: Sparse training for efficient multitask learning	5
1.3 Part III: Transparency	5
1.4 List of publications	6
1.5 Software and repositories	8
2 Background	9
2.1 Language, vision and vision-language encoders	9
2.1.1 Language models	9
2.1.2 Vision models	10
2.1.3 Vision-language models	11
2.1.4 The influence of computer vision and NLP on each other . . .	12
2.2 Commonsense knowledge	13
2.3 Efficient adaption	14
2.3.1 Parameter-efficient fine-tuning	14
2.3.2 Multitask learning	15
2.4 Transparency	16
2.4.1 Interpretability	16
2.4.2 Mechanistic interpretability	17
2.4.3 Interpretability for different-modality models	18
2.4.4 Mechanistic Interpretability for different-modality models . .	20

3	Commonsense Knowledge Enhanced Multimodal Reasoning	23
3.1	Introduction	23
3.2	Related Work	25
3.3	Methodology	25
3.3.1	Image-based fact search	25
3.3.2	Commonsense Knowledge Enhanced Transformer	26
3.4	Results	26
3.5	Analysis	29
3.6	Conclusion	30
3.7	Limitations	30
4	Gradient-based Parameter Selection for Efficient Fine-Tuning	31
4.1	Introduction	31
4.2	Related work	34
4.3	Approach	35
4.3.1	Gradient-based parameter selection	36
4.3.2	Masked fine-tuning	38
4.4	Experiments	38
4.4.1	Experimental settings	39
4.4.2	Performance on image classification	40
4.4.3	Semantic segmentation	42
4.4.4	Impacts of different selection schemes	43
4.4.5	Ablation study	45
4.5	Conclusion	45
5	GPU-Efficient Sparse Training for Parameter-Efficient Fine-Tuning	47
5.1	Introduction	47
5.2	Related Work	50
5.3	Methodology	51
5.3.1	Preliminaries	51
5.3.2	Top- k selection	52
5.3.3	Featherlight adaptation	53
5.4	Neuron-wise Sparse Adaptation: Comparative Analysis	54
5.5	Experiments	57
5.5.1	Commonsense reasoning	59
5.5.2	Arithmetic reasoning	60
5.5.3	Natural language understanding	60
5.6	Conclusion	60
5.7	Limitations	61

6	Sparse Training for Gradient Conflict Mitigation in MTL	63
6.1	Introduction	63
6.2	Related work	65
6.3	Approach	66
6.3.1	Background	66
6.3.2	Sparse training for multi-task learning	67
6.3.3	Theoretical analysis for sparse training	68
6.3.4	Parameter selection per neuron (PSN)	69
6.4	Experiments	70
6.4.1	EXPERIMENTAL SETUP	70
6.4.2	Incidence of gradient conflict	71
6.4.3	Performance on diverse benchmarks	72
6.4.4	Ablation study	74
6.5	Conclusion	77
7	Cross-modal Information Flow in Multimodal LLMs	79
7.1	Introduction	79
7.2	Related work	82
7.3	Tracing information flow in MLLMs	83
7.3.1	Background: MLLMs	83
7.3.2	Attention knockout	85
7.4	Experimental setting	86
7.5	Contribution of different modalities to the final prediction	88
7.6	How is the linguistic and visual information integrated?	91
7.7	How is the final answer generated?	93
7.8	Conclusion	94
8	Conclusions	95
9	Appendices	99
A	Appendix to Chapter 3	99
A.1	Referring expression comprehension	99
A.2	UNITER	99
A.3	Input embedding	100
A.4	Datasets	100
A.5	Experimental settings	101
A.6	Impact of CK-T structure	102
A.7	Introducing facts in traditional REC tasks based on detection	102
A.8	McNemar Test	102
A.9	Example searched fact using different methods	103
B	Appendix to Chapter 4	105
B.1	Details of experiments	105
B.2	Additional experiments	109

B.3	Visualizations	114
B.4	Details of the evaluation datasets	116
B.5	Extended related work	118
B.6	Discussion	120
B.7	Limitations and societal impacts	120
C	Appendix to Chapter 5	122
C.1	Datasets	122
C.2	Results on natural language understanding tasks	124
C.3	Results of commonsense and math reasoning on LLaMA3-8B	124
C.4	Hyperparameters	124
C.5	Advantages of NeuroAda	128
C.6	Algorithm	129
D	Appendix to Chapter 6	131
D.1	Proof for Equation	131
D.2	Limitations	131
D.3	Broader Impacts	131
D.4	Detailed experiment setting	132
D.5	Extended related work	133
D.6	Detailed experiment results	134
E	Appendix to Chapter 7	148
E.1	Dataset collection	148
E.2	Informaion flow for different window size k	149
E.3	Changes in probability of the last sub-word generation	150
E.4	Constructing multimodal semantic representations	152
E.5	Experiments on other models	155
E.6	The fine-grain analysis for information flow	162
E.7	The influence of <i>images</i> on the semantics of <i>Questions</i>	171
E.8	Difference with unimodal LLM	172
E.9	More complex tasks	173
E.10	Attention sink	173
	Bibliography	176
	Samenvatting	219
	Abstract	221

Acknowledgments

I would like to sincerely thank my supervisor, Ekaterina Shutova. When I first arrived, my English was poor, and she patiently explained things to me again and again until I understood. Throughout my PhD, whenever I failed or felt lost, she was always there to encourage me and help me find new directions. Her feedback was always insightful, often providing exactly what I needed to move forward. She also supported every decision I made, which gave me the courage to try, to make mistakes, and to grow. From her, I have learned so much — her sharp eye for research and her rigorous, thoughtful approach to problems. These are lessons I will carry with me into the future.

I am also grateful to the professors I had the chance to work with during my PhD. Xiantong Zhen and Helen Yannakoudakis kindly stepped in as supervisors while Katia was on maternity leave. Their advice, encouragement, and timely guidance helped me keep my research on track. I am deeply thankful to Shanghang Zhang, who welcomed me during my visit at Peking University and provided invaluable academic guidance. Her mentorship throughout the process of preparing and submitting a paper gave me confidence and clarity. I would also like to thank Raquel Fernández, my promoter, for her continuous support with applications and formal processes during my PhD.

I want to thank my colleagues in our group, who made my time there not only productive but also enjoyable. Rochelle Choenni was the first person I met, and I was immediately impressed by her sharp mind and dedication. Her laughter was always contagious and made the lab feel lighter and more welcoming. I am especially grateful to Srishti Yadav, the only one in the lab working on research so close to mine. We had countless discussions, both in the office and online, and I particularly treasure the time we attended a conference together in the U.S.—presenting our work side by side was both exciting and comforting. I also want to thank Ivo Verhoeven, who brought people together through lab lunches and conversations beyond research. I am especially grateful for his help in polishing the Dutch version of my dissertation abstract, and since he knew a bit about China, we often had fun extra topics to chat about. Finally, I want to thank Alina Leidinger, Xiaoyu Tong, and Milan Miletic for the many small exchanges and moments of support that brightened my time in the lab.

Beyond academia, I am deeply grateful to the friends who were by my side. When I first arrived in Amsterdam, I was fortunate to meet Chen Xiao, Wenqing Liu, Mengfei Xu, and Wenwen Guan. They helped me through the most difficult early days of moving to a new place. We cooked together, talked for hours, played cards, and could always joke about anything, no matter how absurd. These moments brought comfort and joy during the uncertainties of the pandemic, and their friendship made that period much easier and brighter.

As time went on, I was lucky to meet Zhenghao Yu, Pengnan Hu, and Yuanling Bao, with whom I had some of the best times of my PhD. We traveled together — from catching crabs in the Dutch countryside, swimming in Palma, and visiting museums in Paris, to exploring Spain and chasing the northern lights across Sweden, Norway, and Denmark. We even picked mushrooms in the mountains and cut chives in the fields. And of course, every time we gathered, our little tradition was playing mahjong, which always filled the room with laughter. What meant the most to me was not just the trips, but the way they treated me — always looking out for me, being patient with my flaws, and showing a kind of acceptance I deeply appreciated. Whether we were traveling, cooking, or simply talking for hours, I felt supported and understood, which made my PhD life much brighter. Later on, I was also fortunate to get to know Xinyu Wang and Fangxue Ma, two really nice friends, and we had a good time together, too, including an unforgettable trip to Switzerland.

After moving to my second place in Amsterdam, I was again fortunate to meet a wonderful group of neighbors and friends: Di Wu, Xiaoyi Feng, Junru Cui, Ran Zhang, and Beihong Yang. We cooked together, shared our daily lives, traveled, played games, danced to Just Dance, sang songs, had barbecues, and even raised cats together. I am especially grateful for the trust they placed in me — allowing me to see their most vulnerable sides, which brought not only laughter but also tears of happiness. These became some of the most unforgettable memories of my years in the Netherlands. We were so close that we even had each other's house keys, and when someone was away, we sometimes gathered in their place for a party. This openness, warmth, and mutual care turned our neighborhood into a true home for me, and I will always treasure those moments. Later, I was also fortunate to meet Shenyuan Zhang, the only true extrovert among us, with whom I shared unforgettable trips through Germany, France, and Italy. Our road trip along the Mediterranean coast left a particularly strong impression — the sea, the food, the music, and the endless conversations made it truly unforgettable.

I was also lucky to have wonderful neighbors who brought so much joy and warmth to my everyday life. I am especially grateful to Yan Dong, Xiaojuan Qiu, and Kezhuan Deng — our “Amsterdam Iron Triangle.” The hotpot nights and long conversations we shared are among my happiest memories, full of laughter and companionship. I owe special thanks to Yan Dong, who was always kind and helpful — once, while helping me repair my bike, I accidentally hurt his fingernail, yet he never blamed me and still patiently fixed it. I am also grateful to Fengling Zhang, such a kind and generous person — we played table tennis and cooked together, and I deeply appreciate her help in offering his lab space so I could repair my mouse. I am also thankful to Jiaxin Li,

my neighbor across the hall, who always brought me delicious treats after her trips, shared her homemade chili oil, and even grew plants together with me — all of which brought life and warmth into my days. I also want to thank Minoxue Zhang, who offered me valuable HR advice, and Caixia Wei, for the many chats and the unforgettable Halloween parties.

I also wish to give special thanks to Congfeng Cao, Yixian Shen, Pengyu Zhang, and Lu Zhang. The weekends we spent working and hanging out together in the office, drinking tea, chatting, and keeping each other company are memories I will never forget. From those long days in the office to grabbing dinner, sharing hotpot at home, playing cards, watching movies, and pushing through paper deadlines, all of these moments brought me warmth and joy. They made me feel truly accompanied and supported, and I will always look back on them with gratitude.

I would like to thank the many friends, colleagues, and professors from other labs who supported me along the way — Sandro, Jelle, Tom, Aditya, Yongtuo, Jaap, Ece, Joris, Anna, Marianne, John, Michael, Haitian, Yibin, Yan, Weijia, Sha, Jiayi, Yingjun, Zehao, Ivona, Qi, Weijie, Fei, Puyu, Oskar, Yang, Shuai, Baohao, Jie, and many others whose names I cannot list one by one. I am truly grateful for all the moments we shared — from playing board games and getting coffee, to going to conferences, grabbing drinks, and simply enjoying each other's company.

最后，我要由衷地感谢自己拥有一个温暖的大家庭。在我求学在外的这些年里，家人们虽各自经历了许多变故，却始终给予我无条件的支持与理解。感谢爷爷奶奶，自我年少时便以慈爱与关怀相伴，他们的言传身教为我一生的成长奠定了坚实的根基。感谢我的父母，他们无论我作出怎样的选择，从未有一丝迟疑，总是坚定地鼓励与支持。感谢李孃的到来为我的生活带来巨大的改善与变化，也为我的成长注入了新的动力与希望。感谢大姑，她以母亲般的关怀细心照料我，让我在成长的道路上常怀安心与温暖；感谢小姑，她如同朋友般理解与支持我，为我的成长增添了无数欢笑与陪伴。感谢二爸、三爸与么爸，他们始终的关怀与厚爱，为我带来安全感。感谢大姑爹给予的那些人生格言。感谢卓孃，她的爽朗与笑语，为我的日常生活平添了无数轻松与愉悦。除此之外，还有许多未能一一提及的亲人，他们在不同的时刻给予我帮助与慰藉。正是这样一个热热闹闹、彼此牵挂、充满爱的大家庭，为我的成长带来了无数滋养与力量，让我无论走到何处，都始终怀揣前行的底气与内心的幸福感。

行文至此，挥笔为终；三十寒暑，学途告竣。人生既逾三分之一，虽非大才，愿无大过。愿此后岁月，常有一二清风，拂我十万八千梦。感怀往昔，曾黔中乡野之子；行虽迟缓，志未尝移；途多崎岖，步不曾止。然念及祖母，孩无祖母，无以至今日，时至今日，却无祖母。愿九泉有知，亦可含笑而慰。

愿山河毓秀无恙，祖国昌盛久长。

Singapore
August, 2025.

Zhi Zhang
张智

Artificial intelligence (AI) has witnessed remarkable progress in recent years, largely driven by the emergence of foundation models across various modalities. These models are typically pre-trained on large-scale, self-supervised datasets, including vision models (e.g., ViT (Chen et al., 2024), SAM (Kirillov et al., 2023b)), language models (e.g., BERT (Devlin et al., 2018), LLaMA (Touvron, 2023)), and multimodal models that integrate vision and language (e.g., UNITER (Chen et al., 2020c), CLIP (Radford et al., 2021a), LLaVA (Liu et al., 2023c)). Advancements in these fields (computer vision, natural language processing, and multimodal learning) are highly interdependent, with progress in one domain often accelerating breakthroughs in the others (Liu et al., 2024b, 2023c; Radford et al., 2021a).

Multimodal systems, particularly those combining visual and textual inputs, are commonly pre-trained on image-caption pairs to align representations across modalities (Chen et al., 2020c; Radford et al., 2021a). While this alignment enables strong performance on recognition tasks that rely on surface-level pattern matching, these models often struggle with reasoning tasks that require deeper understanding beyond perceptual signals. (Deng et al., 2025; Marino et al., 2019; Yang et al., 2025; Zellers et al., 2019a). For instance, although a model may accurately identify visual entities, it may fail to answer questions that require inference or background knowledge“such as why a person is performing an action, or what might happen next“due to a lack of embedded commonsense knowledge. Bridging this gap by enriching multimodal systems with diverse forms of commonsense knowledge, including physical, temporal, and social reasoning, remains an essential and ongoing research direction (Lymperaïou and Stamou, 2024; Marino et al., 2019; Schwenk et al., 2022).

More recently, the increasing scale of foundation models has endowed them with unprecedented capacity to encode and transfer knowledge across diverse domains and tasks (Dosovitskiy et al., 2020b; Dubey et al., 2024b; Touvron, 2023). This enhanced representational power has demonstrated strong potential to achieve state-of-the-art performance across a wide range of downstream applications (Dosovitskiy et al., 2020b; Dubey et al., 2024b; Touvron, 2023). However, the sheer size and complexity of these

models introduce substantial challenges for efficient adaptation. Whether fine-tuned for a single task or jointly optimized for multitask learning, the adaptation process often incurs considerable computational overhead. The large number of parameters significantly increases training and inference time, while also placing heavy demands on memory, storage, and energy resources (Hu et al., 2021a; Li and Liang, 2021; Lian et al., 2022; Zhang et al., 2023c), raising critical concerns regarding the scalability, sustainability, and practical deployability of such models in real-world scenarios.

In parallel, the increasing complexity and scale of multimodal large language models (MLLMs) have amplified the demand for greater transparency and interpretability. A critical aspect of this challenge lies in understanding the internal mechanisms through which MLLMs process and integrate visual and linguistic information, including the locations within the model where such interactions occur and the ways in which they influence the final predictions. Gaining such insights is essential for enhancing the reliability, robustness, and trustworthiness of these systems, particularly in high-stakes domains such as healthcare (Guan et al., 2022; Luo et al., 2024), autonomous systems (Atakishiyev et al., 2024; Guan et al., 2022), and security (Ahmadian et al., 2023; Szegedy et al., 2013).

Based on the above challenges, this dissertation investigates three key research directions: (I) the integration of commonsense knowledge to enhance multimodal reasoning, (II) the development of efficient adaptation methods for both single-task and multitask scenarios, and (III) the improvement of transparency in multimodal large language models. Collectively, these research efforts aim to advance the development of AI systems that are not only robust and scalable but also interpretable and trustworthy.

Part I: Commonsense Knowledge Enhanced Multimodal Reasoning

Commonsense knowledge primarily refers to the fundamental body of practical knowledge about everyday situations and events that is commonly shared across the general population (Minsky, 2000; Shwartz, 2021; Wang and Zhao, 2023). Different categories of commonsense knowledge have been identified, including social commonsense, which involves inferring human intentions from their actions (e.g., recognizing that “*a shopper checking their wallet at a checkout counter intends to make a purchase*”); temporal commonsense, which concerns reasoning about the typical duration and ordering of events (e.g., “*recovering from surgery generally takes longer than recovering from a common cold*”); and physical commonsense, which pertains to understanding the physical properties and interactions of objects (e.g., “*a metal rod is heavier than a plastic rod of the same size*”) (Shwartz, 2021; Wang and Zhao, 2023).

Despite significant progress in vision-language modeling, current multimodal systems often perform well on surface-level tasks but exhibit limited capability in handling inferences that require commonsense knowledge (Deng et al., 2025; Marino et al., 2019; Yang et al., 2025; Zellers et al., 2019a). For example, when asked “Which object in the image is a soft starchy food item?”, humans are likely to infer plausible candidates such as bananas or potatoes, even though such attributes may not be explicitly observable in

the image (Wang et al., 2020b; Zellers et al., 2019a), but existing models frequently fail to make such inferences (Deng et al., 2025). This limitation stems from the fact that most multimodal models are predominantly trained on image-caption pairs or visual question answering (VQA) datasets. These models tend to capture shallow statistical co-occurrences and conceptual alignment between visual and linguistic modalities (Chen et al., 2020c; Liu et al., 2023c; Lu et al., 2019; Marino et al., 2019), but seldom acquire or reason over abstract concepts, such as understanding that “*knives are used for cutting*” (Marino et al., 2019).

Chapter 3: To address this gap between superficial pattern recognition and human-like common-sense reasoning, it is essential to enrich multimodal models with common-sense knowledge. In our research, we aim to bridge that gap by injecting object-level commonsense knowledge into multimodal reasoning systems. Specifically, we propose incorporating commonsense knowledge of individual objects directly into their visual representations, thereby enhancing the model’s reasoning ability in referring expression comprehension (REC) – a task that requires accurately localizing an object in an image based on a natural language description.

Part II: Efficient Adaptation

The field of deep learning has undergone a paradigm shift with the advent of large-scale models, such as Large Language Models (LLMs) (Devlin et al., 2018; He et al., 2020b; Touvron, 2023) and visual foundation models (Dosovitskiy et al., 2020b; Kirillov et al., 2023a; Radford et al., 2021a). These models have achieved impressive performance across a wide range of downstream tasks spanning natural language processing, computer vision, and multimodal learning. Their success is largely attributed to the adaptation of models comprising billions of parameters, which have been pre-trained on massive datasets (Devlin et al., 2018; Dosovitskiy et al., 2020b; Touvron, 2023).

The conventional approach for adapting pre-trained large models to downstream tasks is full fine-tuning, wherein all model parameters are updated using task-specific data. While this method often achieves strong task performance, applying full fine-tuning across a broad range of downstream tasks demands substantial computational resources, including high-performance GPUs, significant memory capacity, and prolonged training time, which collectively result in considerable energy consumption (Bartoldson et al., 2023; Hu et al., 2021a; Zhang et al., 2023d). Furthermore, maintaining multiple fully fine-tuned models for different tasks becomes increasingly impractical due to storage constraints and the computational overhead associated with inference in real-world applications (Bartoldson et al., 2023; Zhang et al., 2023d). In addition, full fine-tuning generally necessitates large-scale, high-quality task-specific datasets to mitigate the risk of overfitting, which is a common challenge when training on small datasets relative to the model’s large capacity (Fu et al., 2023).

To address these limitations, efficient adaptation techniques have become a major focus in deep learning research. These techniques aim to achieve comparable or superior

performance to full fine-tuning (Ding et al., 2023; He et al., 2021a; Hu et al., 2021a, 2022c; Jia et al., 2022a; Li and Liang, 2021; Lian et al., 2022; Su et al., 2021) while reducing computational costs (Dingliwa et al., 2022; Frankle and Carbin, 2018; Han et al., 2015b; Li et al., 2016a), memory usage (Iscen et al., 2020; Mercea et al., 2024; Ramesh et al., 2024; Sprechmann et al., 2018), runtime (Karmanov et al., 2024; Mercea et al., 2024) or the need for extensive task-specific data (Adadi, 2021; Chen et al., 2022; Ivison et al., 2022; Lin et al., 2024; Zhang et al., 2023d). Among the emerging strategies for efficient adaptation, sparse training has attracted increasing attention due to its potential to reduce both training and inference costs by constraining the number of active parameters during optimization. Sparse training approaches aim to update only a small subset of model parameters either statically or dynamically, thereby significantly lowering the computational and memory demands compared to full fine-tuning (Frankle and Carbin, 2019; Mostafa and Wang, 2019). Furthermore, this selective adaptation paradigm inherently lowers the demand for both data and training time, as fewer parameters require gradient computation and optimization (Dettmers and Zettlemoyer, 2019; Varma T et al., 2022). Given these advantages, incorporating sparse training into the adaptation process presents a promising direction for enhancing both single-task and joint multitask learning across a broad range of downstream tasks.

Part II.I: Parameter-efficient fine-tuning for task-specific adaptation

Parameter-efficient fine-tuning (PEFT) has emerged as a prominent paradigm for adapting large pre-trained models to downstream tasks (Hu et al., 2021a; Jia et al., 2022a; Li and Liang, 2021; Lian et al., 2023; Su et al., 2021). Most PEFT methods keep the backbone model frozen and introduce lightweight modules to facilitate task-specific adaptation. For instance, adapter-based methods insert residual modules within fully-connected layers (Bapna et al., 2019; Houlsby et al., 2019; Hu et al., 2021a; Pfeiffer et al., 2020a,b; Rebuffi et al., 2017; Rücklé et al., 2020); prompt-based methods inject learnable prompts into the input or intermediate layers (Ding et al., 2021; Gao et al., 2020; Hu et al., 2021b; Jia et al., 2022a; Ju et al., 2022; Li and Liang, 2021; Liu et al., 2023d, 2022b); and representation editing methods learn transformation parameters to directly modify internal representations of the model (Lian et al., 2022; Liu et al., 2022a; Wu et al., 2024a,c). In contrast, sparse training selectively updates a subset of the model’s original parameters while keeping the pre-trained architecture intact and without introducing additional modules (Hu et al., 2021a; Jia et al., 2022a; Lian et al., 2023; Su et al., 2021).

Chapters 4 and 5: While PEFT explicitly targets the reduction of computational and memory costs by fine-tuning a minimal number of parameters to match or surpass the performance of full fine-tuning, sparse training shares some of these efficiency goals but does not necessarily pursue minimal parameter tuning as its primary objective. In our research, we aim to bridge these two paradigms by evaluating the performance of sparse training on both visual and language models under a PEFT setting, where the number of trainable parameters is deliberately constrained to be as small as possible.

Part II.II: Sparse training for efficient multitask learning

Multitask learning (MTL) aims to train a single model to perform multiple tasks simultaneously, leveraging shared knowledge across tasks to enhance overall performance. However, a fundamental challenge in MTL is negative interference, where jointly training all tasks within a unified model can lead to performance improvements in some tasks at the expense of others (Parisotto et al., 2016; Rusu et al., 2015; Wang et al., 2020d). From an optimization perspective, this phenomenon is often attributed to gradient conflicts where gradients from different tasks point in divergent directions in the parameter space (Yu et al., 2020). When the update directions of two tasks differ significantly (e.g., the angle between their gradients is large), the resulting joint update may favor one task while hindering another (Yu et al., 2020).

Chapter 6: In our research, we explore the mitigation of gradient conflicts in multitask learning from the perspective of sparse training. The main intuition is that sparse training can be interpreted as projecting a high-dimensional optimization problem onto a lower-dimensional subspace. This projection not only reduces the optimization complexity but also restricts the gradient updates of each task to a subset of the model parameters. By limiting the parameter overlap among tasks, sparse training inherently reduces potential interference, thereby facilitating more stable and cooperative multitask optimization, and ultimately improving overall task performance.

Part III: Transparency

Multimodal large language models (MLLMs) (Bai et al., 2023a; Dai et al., 2023; Li et al., 2023a; Liu et al., 2024a,b) have recently achieved remarkable success across a wide range of vision-language tasks, from image captioning to visual question answering (VQA). This progress is largely attributed to the combination of powerful auto-regressive large language models (LLMs) (Touvron et al., 2023a; Zhang et al., 2022b; Zheng et al., 2023) with strong visual encoders (Dosovitskiy et al., 2020a; Fang et al., 2023; Radford et al., 2021b). In these architectures, visual features extracted by the image encoder are concatenated into the input sequence ahead of the word embeddings, enabling the LLM to generate responses conditioned on both visual and linguistic inputs.

Despite these advancements, the internal working mechanisms of MLLMs remain largely opaque. Enhancing transparency in multimodal reasoning is critical for applications where interpretability is essential, such as medical image analysis (Guan et al., 2022; Luo et al., 2024), autonomous driving (Atakishiyev et al., 2024; Guan et al., 2022), and cybersecurity encompassing both attack and defense strategies (Ahmadian et al., 2023; Szegedy et al., 2013). In particular, gaining deeper insights into how MLLMs integrate linguistic and visual information is vital for the development of more efficient, robust, and trustworthy multimodal systems.

Chapter 7: To address this gap, our study systematically investigates how the integration between visual and linguistic information occurs within state-of-the-art MLLMs. Specifically, we employ a *attention knockout* technique (Geva et al., 2023), where the specific attention edge between input tokens is blocked, and the change in confidence score of the final prediction can reflect and identify where and how visual and linguistic information interact.

1.4 List of publications

During my PhD I authored the following conference papers:

1. **Zhi Zhang**, Helen Yannakoudakis, Xiantong Zhen, and Ekaterina Shutova. “CK-Transformer: Commonsense Knowledge Enhanced Transformers for Referring Expression Comprehension.” *In Findings of the European Chapter of the Association for Computational Linguistics, EACL 2023*. URL <https://aclanthology.org/2023.findings-eacl.196>
2. **Zhi Zhang**, Qizhe Zhang, Zijun Gao, Renrui Zhang, Ekaterina Shutova, Shiji Zhou, and Shanghang Zhang. “Gradient-based parameter selection for efficient fine-tuning.” *In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024*. URL https://openaccess.thecvf.com/content/CVPR2024/papers/Zhang_Gradient-based_Parameter_Selection_for_Efficient_Fine-Tuning_CVPR_2024_paper.pdf
3. **Zhi Zhang**, Srishti Yadav, Fengze Han, and Ekaterina Shutova. “Cross-modal Information Flow in Multimodal Large Language Models.” *In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2025*. URL <https://arxiv.org/abs/2411.18620>
4. **Zhi Zhang**, Jiayi Shen, Congfeng Cao, Gaole Dai, Shiji Zhou, Qizhe Zhang, Shanghang Zhang, and Ekaterina Shutova. “Proactive Gradient Conflict Mitigation in Multi-Task Learning: A Sparse Training Perspective.” *Under review at ICCV 2025*. URL <https://arxiv.org/abs/2411.18615>
5. **Zhi Zhang**, Yixian Shen, Congfeng Cao, and Ekaterina Shutova. “NeuroAda: Featherlight Adaptation via Activating Each Neuron’s Potential.” *Under review at EMNLP 2025*.

In addition, I contributed as a co-author to the following conference and journal papers:

6. Srishti Yadav, **Zhi Zhang**, Daniel Hershcovich, and Ekaterina Shutova. “Beyond Words: Exploring Cultural Value Sensitivity in Multimodal Models.” *In Findings*

- of the Association for Computational Linguistics: NAACL 2025. URL <https://arxiv.org/abs/2502.14906>
7. Gaole Dai, Chun-Kai Fan, Yiming Tang, **Zhi Zhang**, Yuan Zhang, Yulu Gan, Qizhe Zhang, Cheng-Ching Tseng, Shanghang Zhang, and Tiejun Huang. “SAN: Hypothesizing Long-Term Synaptic Development and Neural Engram Mechanism in Scalable Model’s Parameter-Efficient Fine-Tuning.” *In Proceedings of the 42nd International Conference on Machine Learning, ICML 2025*. URL <https://arxiv.org/abs/2409.06706>
 8. Xiaoyu Tong, **Zhi Zhang**, Martha Lewis, and Ekaterina Shutova. “Hummus: A Dataset of Humorous Multimodal Metaphor Use.” *Under review at TACL 2025*. URL <https://arxiv.org/abs/2504.02983>
 9. Hanqun Cao, Xinyi Zhou, Zijun Gao, Chenyu Wang, Xin Gao, **Zhi Zhang**, Chunbin Gu, Ge Liu, and Pheng-Ann Heng. “PLAME: Leveraging Pretrained Language Models to Generate Enhanced Protein Multiple Sequence Alignments.” *Under review at NeurIPS 2025*.
 10. Qizhe Zhang, Ruichuan An, Bocheng Zou, **Zhi Zhang**, and Shanghang Zhang. “HyperAdapter: Generating Task-Oriented Adapters for Rehearsal-Free Continual Learning” *Under review at NeurIPS 2025*.
 11. Shiji Zhou, Tianbai Yu, **Zhi Zhang**, Heng Chang, Xiao Zhou, Dong Wu, and Han Zhao. “Efficient Utility-Preserved Machine Unlearning with Implicit Gradient Surgery.” *Under review at NeurIPS 2025*.
 12. Shiji Zhou*, **Zhi Zhang***, Yongliang Wu, Lianzhe Wang, Heng Chang, Wenbo Zhu, Xiao Zhou, and Xu Yang. “Robust and Effective Machine Unlearning with Sparse Regularized Multi-Objective Optimization.” *Under review at NeurIPS 2025*.
 13. Jinjin Cao, Zhiyang Chen, Zijun Wang, Xueji Fang, and Liyuan Ma, **Zhi Zhang**, Weijian Luo, Guo-Jun Qi. “When Images Speak Louder: Mitigating Language Bias-induced Hallucinations in VLMs through Cross-Modal Guidance.” *Under review at ACM MM 2025*.
 14. Congfeng Cao, **Zhi Zhang**, Jelke Bloem, and Khalil Sima’an. “How Aligned Are Unimodal Language and Graph Encodings of Chemical Molecules?” *Under review at EMNLP 2025*.
 15. Zhenghao Yu, Mouraya Hussein, Yuanling Bao, AnaAlcal’ Lalinde, Pascal Kroon, Eva Thuiller, **Zhi Zhang**, Luis Enjuanes, Sonia Zu’iga, Ben Berkhout, and ElenaHerrera-Carrillo. “Broad-spectrum CRISPR-Cas13d-mediated inhibition of human coronaviruses” *Under review at Molecular Therapy-Nucleic Acids 2025*.

* denotes joint first authorship.

1.5 Software and repositories

During my PhD, I contributed to the development and maintenance of Python packages and GitHub repositories that implement the models and algorithms proposed in my research publications. The following repositories contain codebases corresponding to the respective works:

- Commonsense knowledge enhanced multimodal reasoning:
<https://github.com/FightingFighting/CK-Transformer>
- Efficient adaptation:
<https://github.com/FightingFighting/GPS>
- Transparency:
<https://github.com/FightingFighting/cross-modal-information-flow-in-MLLM>

As this research focuses on the challenges, including integrating external factual knowledge in multimodal models, enabling pre-trained model efficiently adapt to new tasks, understanding the internal working mechanism of multimodal models, from the field of language, vision, and their combination, we provide the backgrounds of them in this chapter.

2.1 Language, vision and vision-language encoders

The models used in this research comes from the fields of natural language processing (NLP), computer vision (CV), and vision-language modeling. In the following sections, we provide a detailed introduction to each. In recent years, these fields have witnessed remarkable advancements. These developments are deeply interconnected, with advancements in one domain often acting as a catalyst for progress in another.

2.1.1 Language models

Previous models Before the Transformer architecture was introduced, language models mainly relied on traditional statistical methods and recurrent neural network-based architectures. N-gram Models (Shannon, 1948) estimated probabilities of fixed-length word sequences but suffered from data sparsity. RNN (Recurrent Neural Networks) (Rumelhart et al., 1986) captured long-range dependencies but faced vanishing gradient and efficiency issues. LSTM/GRU (Long Short-Term Memory/Gated Recurrent Unit) (Cho et al., 2014; Hochreiter and Schmidhuber, 1997) improved RNNs’ ability to model long-term dependencies but remained computationally challenging for long sequences.

Transformer and its evolution In 2017, Vaswani et al. (2017a) introduced the Transformer architecture, revolutionizing language modeling. The key innovations included self-attention mechanisms that efficiently captured long-range dependencies, positional encoding to address the lack of positional information in CNNs and RNNs, and fully parallel computation that significantly improved training efficiency over RNNs. This

architecture led to the rise of numerous Transformer-based pre-trained language models, which have become central to modern NLP technology.

Evolution of pre-trained language models BERT (Devlin et al., 2019) was a bidirectional Transformer model that introduced Masked Language Modeling (MLM) and Next Sentence Prediction (NSP) tasks for better text understanding. At the same time, GPT (Radford et al., 2019) was released, which employed an autoregressive generation approach, excelling in natural language generation. T5 (Raffel et al., 2020) unified text generation, classification, and translation by formatting all tasks as text-to-text conversions. BART (Lewis et al., 2019) combined BERT’s bidirectional encoding with GPT’s autoregressive decoding to enhance text generation quality. After that, GPT-3 and Large-Scale Models (Brown et al., 2020) scaled up to 175B parameters, demonstrating strong zero-shot and few-shot learning abilities. More recent, ChatGPT and GPT-4 (OpenAI, 2023) integrated RLHF (Reinforcement Learning with Human Feedback) to make interactions more natural and human-like. LLaMA, Mistral, and Other Open-Source Models (Touvron, 2023) were released by companies like Meta, providing lightweight, efficient models that promoted further advancements in open LLM research.

2.1.2 Vision models

Convolutional neural networks (CNNs) and their evolution The adoption of deep learning in vision tasks was significantly influenced by the success of AlexNet (Krizhevsky et al., 2012), which demonstrated the effectiveness of deep CNNs in image classification. Subsequent models such as VGGNet (Simonyan and Zisserman, 2014) and ResNet (He et al., 2016) further improved performance by increasing network depth and introducing residual connections, respectively. CNNs have remained the dominant approach for various vision tasks, including object detection, segmentation, and face recognition (He et al., 2017; Ren et al., 2015). More recently, ConvNeXt (Liu et al., 2022c) demonstrated that with modern design principles, CNNs can still achieve competitive performance against transformer-based models.

The rise of vision transformers (ViTs) A paradigm shift occurred with the introduction of Vision Transformers (ViTs) by Dosovitskiy et al. (2020b). Unlike CNNs, ViTs leverage self-attention mechanisms to model long-range dependencies in images, achieving competitive performance on image classification benchmarks. This approach was further improved with hierarchical transformer architectures like Swin Transformer (Liu et al., 2021d), which introduced local attention windows to enhance computational efficiency. ViTs have demonstrated remarkable generalization capabilities, prompting widespread adoption in tasks such as segmentation (Xie et al., 2021) and object detection (Carion et al., 2020). Additionally, hybrid models combining CNNs and transformers have emerged to leverage the strengths of both architectures (Graham et al., 2021). Recent advancements in this domain include Masked Autoencoders (He et al., 2022), which utilize self-supervised learning to improve feature representations, and SAM

(Kirillov et al., 2023b), a powerful vision model designed for general segmentation tasks, significantly advancing the field of universal vision modeling.

2.1.3 Vision-language models

Early developments in vision-language models Early efforts in vision-language modeling focused on handcrafted features and rule-based approaches. The initial methods relied on convolutional neural networks (CNNs) for image processing and recurrent neural networks (RNNs) for text generation. Classical models such as Show and Tell (Vinyals et al., 2015) introduced an encoder-decoder framework for image captioning, where CNNs extracted visual features, and RNNs generated textual descriptions based on the extracted visual features. However, these approaches faced limitations in capturing complex multimodal relationships.

Evolution of transformer-based vision-language models The advent of transformer architectures revolutionized vision-language modeling. The introduction of the Vision Transformer (Dosovitskiy et al., 2020b) and transformer-based NLP models like BERT (Devlin et al., 2019) paved the way for multimodal transformers. Early transformer-based VLMs such as ViLBERT (Lu et al., 2019) and LXMERT (Tan and Bansal, 2019a) adopted a dual-stream architecture, where separate encoders processed vision and language inputs before fusing their representations through cross-modal attention mechanisms. This approach allowed for a more comprehensive understanding of multimodal data. Subsequent models, such as UNITER (Chen et al., 2020c) and VisualBERT (Li et al., 2019), introduced single-stream architectures, where a shared transformer encoder processed both image and text inputs jointly. This integration improved computational efficiency and enhanced multimodal representation learning. The rise of contrastive learning further improved vision-language models. CLIP (Radford et al., 2021b) and ALIGN (Jia et al., 2021) leveraged large-scale image-text pairs to train models that aligned visual and textual features in a shared embedding space. These models exhibited remarkable zero-shot generalization, enabling classification, retrieval, and other vision-language tasks without explicit task-specific fine-tuning.

Multimodal large language models The recent advancements in multimodal large language models (MLLMs) have been largely driven by the rapid progress in large language models (LLMs). The increasing scale and capability of LLMs have facilitated their extension into multimodal domains, enabling more sophisticated interactions between vision and language. Unified frameworks like Flamingo (Alayrac et al., 2022), LLaMA-3 (Dubey et al., 2024a) and BLIP (Li et al., 2022a) integrate vision processing into powerful LLMs to enhance multimodal reasoning and understanding across diverse tasks. A significant breakthrough in this domain has been the emergence of models such as LLaVA (Liu et al., 2023c), Qwen-VL (Bai et al., 2023b), which use a single MLP and cross-attention module to connect vision encoders like CLIP (Radford et al., 2021b) and the pre-trained LLMs like LLaMA (Touvron, 2023) respectively, allowing them to process and generate image-grounded textual responses. These models excel

in tasks such as visual reasoning, detailed image captioning, and multimodal dialogue generation.

2.1.4 The influence of computer vision and NLP on each other

Influence of computer vision on NLP models Recent advances in NLP have been influenced by methods from CV. CNNs, originally designed for image processing (LeCun et al., 1998), have been adapted for text classification (Kim, 2014). Contrastive learning, widely used in CV (Chen et al., 2020b; He et al., 2020a), has inspired self-supervised approaches in NLP (Gao, 2021). Normalization techniques have also transferred from CV to NLP, with LayerNorm (Ba et al., 2016) replacing BatchNorm (Ioffe and Szegedy, 2015) in Transformer architectures (Devlin et al., 2018). More recently, diffusion models, first developed for image generation (Ho et al., 2020), have been applied to text generation (Li et al., 2022b). These cross-domain influences demonstrate the growing convergence between NLP and CV.

Influence of NLP on vision models Inspired by NLP models like BERT (Devlin et al., 2018), Vision Transformers (ViTs) leverage self-attention mechanisms for improved feature learning. Masked image modeling, an adaptation of masked language modeling, has also enhanced vision models. For example, BEiT (Bao et al., 2021) and MAE (He et al., 2022) apply the concept of masking and reconstructing missing patches in images. Another major influence from NLP is the use of large-scale pretraining strategies, which have shaped the development of foundation vision models, like Florence (Yuan et al., 2021).

Influence of NLP and CV on vision-language models Early methods relied on CNNs for image feature extraction and RNNs for text processing, with simple fusion at the task level. For example, in image captioning (Vinyals et al., 2015), CNNs extract image features, which are then processed by RNNs for sequence modeling. The rise of Transformers in NLP, such as BERT (Devlin et al., 2018), enhanced semantic understanding. This facilitated the development of the structures combining Faster R-CNN (Ren et al., 2015) and Transformer, which allow object detection and textual information to be jointly optimized within the same framework, leading to models such as ViLBERT (Lu et al., 2019) and LXMERT (Tan and Bansal, 2019a) (two-stream architectures), as well as UNITER (Chen et al., 2020c) and VisualBERT (Li et al., 2019) (single-stream architectures). Subsequently, the development of Vision Transformers (ViT) (Dosovitskiy et al., 2020b) and Swin Transformer (Liu et al., 2021d) in the field of computer vision allowed image processing to move beyond the limitations of CNNs, strengthening global information modeling capabilities. This paved the way for unified architectures like LLaVA (Liu et al., 2023c) and Qwen-VL (Bai et al., 2023b), integrating text and vision into a single Transformer framework for more efficient multimodal learning. The continued development of language models and vision models has paved the way for more capable multimodal models, bridging the gap between vision and language comprehension and fostering the rise of general-purpose

AI systems with high-level multimodal reasoning capabilities.

2.2 Commonsense knowledge

Commonsense knowledge is the fundamental knowledge that humans accumulate about the world through daily experiences (Minsky, 2000). For example, we know that “fire can burn the skin”, “bananas are edible” and “cars require fuel to run”. Commonsense knowledge can be mainly categorized into three types: social commonsense, temporal commonsense, and physical commonsense (Shwartz, 2021). Social commonsense is the human ability to infer others’ mental states, such as motivations, intentions, and future actions, as well as internalized norms of acceptable behavior, like recognizing that “it’s impolite to interrupt someone who is speaking” (Shwartz, 2021). While humans apply such knowledge implicitly in our actions and decisions, machines must be explicitly taught these social conventions. Temporal commonsense refers to the implicit understanding of time-related information in natural language, which often relies on commonsense knowledge rather than explicit statements (Shwartz, 2021). For example, “Kane is baking a cake” indicates an event lasting a few hours, whereas “Kane is renovating his house” suggests a process spanning weeks or months. Such knowledge typically involves durations, sequences, and frequencies etc. of events. Physical commonsense captures intuitive knowledge about the physical properties and affordances of everyday objects (Shwartz, 2021). For example, most people understand that ice will melt if left out at room temperature. Such knowledge underpins many aspects of reasoning about the physical world.

Several commonsense knowledge bases have been developed, such as Wikipedia (Vrande‘i and Kr‘tzech, 2014), ConceptNet (Speer et al., 2017a), and WebChild (Tandon et al., 2017a). These resources provide structured or unstructured knowledge across various domains. Wikipedia (Vrande‘i and Kr‘tzech, 2014), as the world’s largest online encyclopedia, contains extensive unstructured knowledge. In contrast, ConceptNet and WebChild include structured knowledge. Specifically, ConceptNet (Speer et al., 2017a) is a graph-structured commonsense knowledge base that formats information as triplets, for example (a start node, a relation, an end node). The relations are drawn from a closed set, including IsA, HasA, MadeOf, PartOf, CapableOf, and UsedFor. WebChild (Tandon et al., 2017a) extracts commonsense knowledge from web data to construct a large-scale knowledge base, where the knowledge is further categorized into three classes: properties (e.g., HasSize, HasShape, HasTaste), comparatives (e.g., SmallerThan, FasterThan), and part-whole relations (e.g., SubstanceOf, PhysicalPartOf, MemberOf).

Language models are typically pre-trained on large-scale text corpora in a self-supervised manner, either by predicting the next word in a sequence (e.g., GPT; Radford et al., 2018) or by recovering masked words within a sentence (e.g., BERT; Devlin et al., 2018). A growing body of research has investigated whether such models inherently capture commonsense knowledge. Empirical studies have demonstrated that language

models exhibit the ability to encode certain aspects of commonsense knowledge, as evidenced by their performance on tasks such as completing incomplete commonsense facts (Petroni et al., 2019b), ranking plausible candidate facts (Davison et al., 2019), associating concepts with a given set of properties (Weir et al., 2020), and even mining new commonsense facts from text (Trinh and Le, 2019). Nonetheless, the commonsense knowledge acquired by language models is often noisy and brittle. For instance, models tend to exhibit insensitivity to negated statements, failing to distinguish between affirmative and negated versions of facts (Ettinger, 2020; Kassner and Schütze, 2020). To address these limitations, recent research has explored integrating external commonsense knowledge into language models to enhance downstream natural language processing tasks, including story generation (Guan et al., 2020), recommendation systems (Yang et al., 2024), question answering (Bian et al., 2021), event representation learning (Ding et al., 2019), and commonsense reasoning (Liu et al., 2021b).

Vision-language models With the rapid advancement of vision-language models, including early transformer encoder-only architectures such as UNITER (Chen et al., 2020d), VL-BERT (Su et al., 2020), and LXMERT (Tan and Bansal, 2019b), as well as more recent multimodal large language models (transformer decoder-only) like LLaVA (Liu et al., 2023c) and Qwen-VL (Bai et al., 2023b), substantial progress has been achieved in various multimodal tasks. Despite these advances, most existing models primarily rely on data-driven alignment between visual and linguistic modalities through large-scale image-text pair training, which often limits their capacity for explicit commonsense reasoning (Ye et al., 2023). While some studies suggest that pre-trained multimodal models may exhibit a degree of commonsense understanding, with preliminary evidence indicating a potential to associate visual concepts with linguistic knowledge, there remains a substantial gap in achieving robust and generalizable multimodal reasoning capabilities (Yang and Silberer, 2022; Yun et al., 2021). Recent efforts have begun to address this limitation by enhancing the commonsense reasoning abilities of multimodal models in both vision-language and purely linguistic tasks, such as image-text retrieval (Ye et al., 2023), reading comprehension (Yariv et al., 2024), visual commonsense reasoning (Yariv et al., 2024), and knowledge-based visual question answering (VQA) (Rao et al., 2023). In our research, we propose a method to integrate structured commonsense knowledge into the visual representations of objects within an image, aiming to enhance the multimodal reasoning capabilities of models, with a particular focus on improving performance in the task of referring expression comprehension.

2.3 Efficient adaption

2.3.1 Parameter-efficient fine-tuning

The pre-training and fine-tuning paradigm has become a standard approach for adapting large-scale models, initially pre-trained on vast amounts of data, to downstream tasks

with limited supervision. Nevertheless, fine-tuning all parameters of such models poses significant challenges in terms of memory consumption and data efficiency. This makes full fine-tuning impractical, particularly when adapting a model to multiple tasks concurrently (Houlsby et al., 2019; Jia et al., 2022a; Lian et al., 2022). To address these challenges, parameter-efficient fine-tuning (PEFT) methods have been proposed. These methods focus on optimizing only a small subset of parameters while keeping the majority of the pre-trained model fixed. This not only reduces memory and computational costs but also simplifies optimization and mitigates the risk of overfitting under low-resource conditions. Moreover, PEFT methods often achieve performance comparable to, or even better than, full fine-tuning, while maintaining high efficiency (Jia et al., 2022a).

Among existing PEFT approaches, one of the most prevalent strategies involves introducing additional parameters to the pre-trained backbone for fitting the downstream tasks. For instance, adapter-based methods introduce and update a residual module of the fully-connected layers (Bapna et al., 2019; Houlsby et al., 2019; Pfeiffer et al., 2020a,b; Rebuffi et al., 2017; Rücklé et al., 2020; Stickland and Murray, 2019; Sung et al., 2022; Wang et al., 2020c; Zhang et al., 2021b). Moreover, prompt-based and representation editing methods injecting learnable prompts into the input or intermediate representations (Ding et al., 2021; Gao et al., 2020; Hu et al., 2021b; Jia et al., 2022a; Ju et al., 2022; Li and Liang, 2021; Liu et al., 2023d, 2022b), and by learning transformation parameters to edit the internal representations of the model (Lian et al., 2022; Liu et al., 2022a; Wu et al., 2024a,c), respectively. In contrast to methods that introduce additional parameters, our research proposes a novel selection-based approach that refrains from adding any new parameters. Instead, it selectively updates a subset of the original model’s parameters while preserving the intrinsic architecture of the pre-trained model.

2.3.2 Multitask learning

Multitask learning aim to enable a single model to simultaneously learn multiple tasks, with the objective of improving overall performance by leveraging shared knowledge across tasks, respectively. A fundamental challenge of them is how to effectively share information while mitigating potential interference across different tasks. One straightforward strategy is to explicitly design the model architecture with two components: task-specific modules, which capture specific knowledge for individual tasks, and shared modules, which facilitate knowledge transfer across them (Kokkinos, 2017; Murugesan and Carbonell, 2017; Sachan and Neubig, 2018; Shi et al., 2023; Sun et al., 2020; Zhang et al., 2021a, 2025). Another line of research retains a unified architecture and instead focuses on resolving the problem of negative interference, where the learning of one task adversely impacts others, by exploring solutions from an optimization perspective (Chen et al., 2020e; Choenni et al., 2022; Liu et al., 2023a, 2021a,c; Navon et al., 2022; Sener and Koltun, 2018; Shaham et al., 2022; Wang et al., 2020d; Yu et al., 2020).

Negative interference is a prevalent challenge in joint learning of multiple tasks. Specifically, competition may arise during joint optimization, wherein performance improvements on certain tasks can lead to degradation on others, compared to training them independently (Parisotto et al., 2016; Rusu et al., 2015; Wang et al., 2020d). From an optimization point of view, a primary cause of this phenomenon is the existence of conflicting gradients among tasks (Yu et al., 2020) during training. In joint training, task-specific gradients are typically averaged to compute the final update direction. However, when the angle between some of these gradients exceeds 90 degrees, the resulting update may reduce the loss for one task while increasing it for another, thereby hindering overall convergence. In our research, we investigate whether the proposed sparse training method where only a subset of model parameters is updated while the overall model architecture remains unchanged, can effectively mitigate gradient conflicts, with the aim of improving overall multitask performance.

2.4 Transparency

2.4.1 Interpretability

With the rapid advancement of deep learning across various domains, including computer vision, natural language processing and multimodal learning, black-box models have become the dominant paradigm due to their remarkable performance (Xu and Yang, 2025). However, their internal decision-making processes often remain non-transparent, which poses significant challenges in critical application scenarios such as medical image analysis (Guan et al., 2022; Luo et al., 2024), autonomous driving (Atakishiyev et al., 2024; Guan et al., 2022), and cybersecurity, including both attack and defense mechanisms (Ahmadian et al., 2023; Szegedy et al., 2013). This opacity underscores the urgent need for transparency and interpretability, which aim to uncover the internal mechanisms of black-box models and clarify their decision-making processes (Xu and Yang, 2025). Interpretability methods can generally be classified into two primary categories: active and passive approaches (Xu and Yang, 2025; Zhang et al., 2021c).

Active interpretability refers to techniques that intervene in a model’s structure prior to training or involve the design of inherently transparent model architectures. These approaches aim to enhance interpretability while maintaining task performance, thereby making the model’s decision-making process more intelligible to human users (Xu and Yang, 2025; Zhang et al., 2021c). For instance, decision trees utilize a series of “if-then” rules arranged in a hierarchical structure, intuitively revealing the model’s decision-making logic and providing a clear basis for understanding predictions (Custode and Iacca, 2023; Ueno and Zhao, 2018; Ying et al., 2015). Similarly, knowledge graphs, which consist of heterogeneous triplets (head entity, relation, tail entity), also offer high interpretability (Mohamed et al., 2021; Neil et al., 2018; Wang et al., 2019d), especially in tasks like information retrieval (Gaur et al., 2022) and question answering (Tuan et al., 2022). Additionally, some methods employ additive model structures, where each

feature contributes independently to the final output through smooth functions, allowing for a clearer understanding of feature effects (Agarwal et al., 2021; Caruana et al., 2015; Hastie, 2017; Kraus et al., 2024). Other approaches incorporate optimization techniques “such as regularization (Srivastava et al., 2014), activation regularization (Wu et al., 2017), and explanation loss (Dong et al., 2017)” or integrate interpretable modules like capsule networks (Sabour et al., 2017), Memory Wrap (La Rosa et al., 2023), and Stack-NMN (Hu et al., 2018), to guide the model toward more transparent reasoning mechanisms during training.

Passive interpretability, in contrast to active interpretability, is applied post hoc “that is, after the model has been trained. These methods aim to uncover human-understandable rules or patterns by analyzing various components of the model, including its learned weights, output behaviors, and architecture (Xu and Yang, 2025; Zhang et al., 2021c). A wide range of representative approaches fall under this category, each focusing on different aspects of the model, such as behavior, attribution, concepts, and underlying mechanisms (Bereska and Gavves, 2024). Behavior-based methods are model-agnostic, treating the model as a black box and analyzing only the relationship between input and output. These techniques are widely used to assess model robustness and identify dependencies among input variables (Covert et al., 2021; Ribeiro et al., 2016; Shapley et al., 1953). Attribution-based methods seek to explain model outputs by tracing them back to individual input contributions, thus providing transparency without requiring access to internal structures. The most common technique in this category is based on the gradient, such as Integrated Gradients (Sundararajan et al., 2017), Grad-CAM (Selvaraju et al., 2017), SmoothGrad (Smilkov et al., 2017), and DeepLIFT (Shrikumar et al., 2017a). Concept-based methods explore the internal representations learned by the model with the goal of probing abstract concepts and behavioral patterns, such as probing latent knowledge (Burns et al., 2022), quantifying the similarity of the representation across different models (Bansal et al., 2021; Burns et al., 2022). Our work primarily focuses on mechanistic interpretability, which aims to dissect and understand the internal computational mechanisms of models, and will be introduced in detail in the next section.

2.4.2 Mechanistic interpretability

Mechanistic interpretability (Nanda et al., 2023; Olah, 2024) is an emerging subfield within artificial intelligence interpretability research that aims to elucidate the internal computational mechanisms of neural networks. Rather than treating models as black boxes, this approach emphasizes analyzing the internal structure of models by examining components such as weights, neurons, layers, activations, and circuits, in order to derive meaningful explanations for model behavior (Bereska and Gavves, 2024). In general, this approach adopts a reverse-engineering methodology to identify functional roles of specific network components (Conmy et al., 2023; Sharkey et al., 2025). Mechanistic interpretability plays a critical role not only in understanding model decisions but also

in facilitating a range of downstream applications (Lin et al., 2025). These include model editing and intervention (Basu et al., 2024), the enhancement of compositional generalization capabilities (Zarei et al., 2024), and the identification and mitigation of spurious correlations (Balasubramanian et al., 2024). While this approach has attracted substantial attention in the context of natural language processing, research on mechanistic interpretability in multimodal models remains relatively limited. Our research focuses on this type of interpretability in the multimodal domain, with the goal of investigating how large multimodal language models integrate and interact across different modalities.

2.4.3 Interpretability for different-modality models

Vision models have been extensively explored from the perspective of interpretability. A fundamental category of these methods is feature visualization, which aims to reconstruct input patterns that maximally activate specific neurons. Such techniques reveal the hierarchical feature organization in convolutional neural networks (CNNs), from low-level edges to high-level semantic concepts (Springenberg et al., 2015; Zeiler and Fergus, 2014). Another widely adopted class focuses on attribution-based visualization, which highlights input regions most responsible for the model’s predictions. Techniques such as CAM (Zhou et al., 2016), Grad-CAM (Selvaraju et al., 2017), Score-CAM (Wang et al., 2020a), and SmoothGrad (Smilkov et al., 2017) have demonstrated that CNNs often rely on sparse, localized, and discriminative regions, rather than the entire object. Beyond visualization, interpretability research also explores the construction of diagnostic datasets to probe model biases or attributions. For instance, works in (Kim et al., 2018, 2023; Lucieri et al., 2020) build counterfactual examples to interpret how a specific concept affects the outcome of the target image classifier, while Lapuschkin et al. (2019) reveal that models may exploit spurious correlations, such as background artifacts or watermarks, instead of target-relevant features. Relatedly, studies using adversarial examples (Goodfellow et al., 2015; Ilyas et al., 2019) highlight that vision models often rely on non-robust and imperceptible features that, while predictive, are misaligned with human perception. This challenges the assumption that the representations learned by these models are inherently meaningful or interpretable from a human perspective. Recent works have introduced auxiliary or synthetic datasets to systematically examine such biases; for example, Arias-Duart et al. (2023) employ Stable Diffusion (Rombach et al., 2022) to generate images with controlled contextual variations, enabling fine-grained analysis of model behavior across different scenarios (Gao et al., 2023a; Jain et al., 2022). Additionally, feature inversion techniques (Dosovitskiy and Brox, 2016; Mahendran and Vedaldi, 2015) have shown that deeper layers in CNNs retain substantial perceptual information, but show a bias toward texture. Efforts to quantify the semantic alignment of internal representations, such as Bau et al. (2017) reveal that only a limited subset of neurons corresponds to clearly defined, human-understandable concepts. With the advent of Vision Transformers (ViTs), attention-based interpretability has gained traction. For example, Chefer et al. (2021b) propose an attention attribution method

that illustrates how ViTs exhibit more global receptive behavior compared to CNNs, effectively capturing object boundaries, though deeper attention heads tend to specialize in class-specific semantics.

Language models have witnessed substantial progress, prompting a growing body of research focused on their interpretability (Mosbach et al., 2024). Some works focused on exploring the contribution of the modules in the model for the final prediction. For example, in Jain and Wallace (2019) questioned the explanatory power of attention mechanisms, demonstrating that attention weights do not reliably indicate causal decision pathways. Differently, several works focused on analyzing the internal structure and semantic representations within models. For instance, Clark et al. (2019b) revealed functional differentiation across attention heads and layers, where lower layers predominantly encode syntactic relations and higher layers capture semantic alignment. Similarly, probing studies by Liu et al. (2019a) demonstrated that contextual embeddings hierarchically encode linguistic knowledge, ranging from surface syntax to abstract semantics. To further enhance interpretability, causal and counterfactual analysis frameworks have been proposed, enabling the identification of decision-critical input features and the diagnosis of model vulnerabilities through controlled perturbations and influence-tracing techniques (Alvarez-Melis and Jaakkola, 2017; Moradi and Samwald, 2021). Additionally, researchers have exposed the social and cultural biases embedded in large language models by employing specially curated datasets, illustrating how these models frequently reflect cultural centrism and value-laden assumptions inherited from their training corpora (Nadeem et al., 2020; Tao et al., 2024). Another important line of inquiry explores factual knowledge within language models, such as investigating whether models can memorize such knowledge (Petroni et al., 2019a), where it is stored (Jiang et al., 2020), how it can be modified or updated (De Cao et al., 2021; Dhingra et al., 2022), and how it is extracted during inference (Geva et al., 2023).

Vision-language models have also attracted increasing attention in the field of interpretability. Several studies adopt a black-box perspective by analyzing the input-output behavior of models to understand the influence of various modalities. For example, Cao et al. (2020) compare the relative importance of visual and textual modalities, while Frank et al. (2021a) investigate how each modality contributes to performance across visual and textual tasks. In contrast, other works focus on instance-level explanations by attributing model predictions to specific input components. These include techniques such as aggregating attention scores across layers (Aflalo et al., 2022; Stan et al., 2024), gradient-based attribution methods (Chefer et al., 2021a), and model disentanglement strategies (Lyu et al., 2022). Furthermore, a top-down perspective has been employed in probing studies, which aim to uncover high-level semantic properties encoded in learned representations. These approaches reveal the extent to which models capture visual semantics (Dahlgren Lindström et al., 2020), verb understanding (Hendricks and Nematzadeh, 2021), and physical attributes such as shape and size (Salin et al., 2022).

2.4.4 Mechanistic Interpretability for different-modality models

Vision models have received comparatively less attention in the field of mechanistic interpretability, particularly in contrast to the language models. Nonetheless, recent efforts have begun to bridge this gap. Joseph (2023) proposed a framework for probing the internal mechanisms of Vision Transformers (ViTs), utilizing the logit lens technique to visualize the evolution of patch-level predictions across model layers. Moreover, Bahador (2025) conducted a fine-grained analysis of individual attention heads in ViTs fine-tuned on visually distorted 2D representations that include extraneous artifacts such as axis labels, titles, and color bars. Their results reveal a hierarchical specialization within the model: attention heads in early layers predominantly capture low-level and often task-irrelevant features (e.g., text fragments, edges, or corners), whereas deeper layers exhibit increased task relevance by attending to semantically meaningful regions. Zimmermann et al. (2023) investigated the effect of model scale on mechanistic interpretability using a psychophysical evaluation paradigm. Their results “based on a range of architectures including GoogLeNet (Szegedy et al., 2015), ConvNeXt (Liu et al., 2022d), ResNet-50 (He et al., 2016), and ViT (Dosovitskiy et al., 2020b)” demonstrate that scaling alone does not inherently improve interpretability.

Language models have been the focus of a substantial body of mechanistic interpretability. Several methods have been proposed to elucidate the internal mechanisms of models by examining how information is represented and processed across different layers. For instance, techniques such as the logit lens have been employed to reveal how prediction confidence evolves over computational stages, as well as to probe the nature of information encoded in intermediate hidden representations (Belrose et al., 2023a; Geva et al., 2022; Pal et al., 2023). In a similar vein, other studies focus on analyzing the functional roles of specific architectural components, including the encoder in encoder-decoder transformers (Langedijk et al., 2023) and individual attention heads in autoregressive models such as GPT-2 (Sakarvadia et al., 2023). Moreover, given that certain features can be interpreted as linear combinations of neuron activations (Elhage et al., 2022), a growing body of research has focused on disentangling such representations by decomposing neural network activations into interpretable component features through the use of sparse autoencoders (Cunningham et al., 2023; Marks et al., 2024; Sharkey et al., 2022). In addition, a body of work has adopted causal methodologies “such as causal tracing (Meng et al., 2022), causal mediation analysis (Vig et al., 2020), and causal ablation (Wang et al., 2022)” to investigate the decision-making processes within models by isolating and manipulating internal activations to assess their impact on output behavior. These methods have enabled key applications such as localizing model behavior by identifying critical activation pathways responsible for storing and retrieving factual knowledge (Geva et al., 2023; Goldowsky-Dill et al., 2023; Meng et al., 2022; Stolfo et al., 2023), as well as analyzing component interactions via circuit-level analysis to uncover sub-networks responsible for implementing specific functions within the model’s computation graph (Geva et al., 2023; Hanna et al., 2023; Hendel et al., 2023; Lieberum et al., 2023; Wang et al., 2022).

Vision–language models have received comparatively less attention in the domain of mechanistic interpretability, despite the growing interest in this area within natural language processing. One of the early efforts in this direction is the causal tracing tool proposed by [Palit et al.](#) for image-conditioned text generation on BLIP ([Li et al., 2022a](#)). Recent studies have begun to probe the internal behavior of multimodal large language models (MLLMs) by connecting specific external outputs to underlying mechanisms“such as the storage of information in model parameters ([Basu et al., 2024](#)), the emergence of toxic content in the logit distributions of initial tokens ([Zhao et al., 2024](#)), and the localization and transformation of object-related visual information across layers ([Neo et al., 2024](#); [Palit et al.](#); [Schwettmann et al., 2023](#)). Further investigations have explored the localization of safety mechanisms ([Xu et al., 2024](#)) and the mitigation of redundant visual tokens ([Zhang et al., 2024a](#)). Nevertheless, research offering a comprehensive understanding of the internal mechanisms behind multimodal information integration in MLLMs is still lacking. Our research makes an important first step towards filling this gap.

Chapter 3

Commonsense Knowledge Enhanced Multimodal Reasoning

Chapter Highlights One of the key challenges in multimodal systems is their limited ability to perform inferences that require commonsense knowledge, which is a more human-like form of reasoning compared to superficial pattern recognition. In this research, we aim to enhance multimodal reasoning by integrating object-level commonsense knowledge into multimodal systems. Specifically, we focus on a high-level multimodal reasoning task: referring expression comprehension (REC). We propose incorporating commonsense knowledge of individual objects directly into their visual representations, thereby improving the model’s reasoning capabilities. This approach seeks to bridge the gap between surface-level recognition and deeper, more nuanced inference, making the system better equipped for tasks that demand an understanding of the world’s underlying commonsense relationships.

3.1 Introduction

Referring expression comprehension (REC) aims at locating a target object/region in an image given a natural language expression as input. The nature of the task requires multi-modal reasoning and joint visual and language understanding. In the past few years, several REC tasks and datasets have been proposed, such as RefCOCO (Yu et al., 2016), RefCOCOg (Mao et al., 2016) and RefCOCO+ (Yu et al., 2016) (RefCOCOs). These ‘conventional’ REC tasks typically focus on identifying an object based on visual or spatial information of the object, such as its colour, shape, location, etc.; therefore primarily evaluating a model’s reasoning abilities over visual attributes and spatial relationships.

In practice, however, people often describe an object using non-visual or spatial information – consider, for example, the sentence (expression) “Give me something soft but rich in starch to eat” (Wang et al., 2020b). Such instances require reasoning beyond spatial and visual attributes, and need to be interpreted with respect to the common

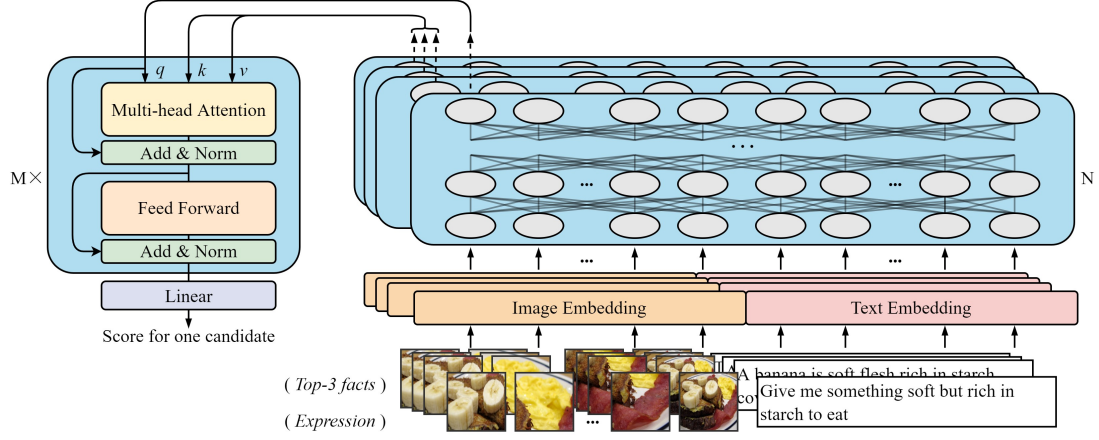


Figure 3.1: CK-Transformer. For each candidate (the first one in the figure), given an expression, a set of visual region candidates and top-K facts ($K=3$ in the figure), the model first encodes the expression and all top-K facts into corresponding multi-modal features, then fuses these features and maps them into a matching score for the candidate.

sense knowledge (fact) embedded in the expressions, such as knowledge about which kind of objects are edible, soft and rich in starch in the given image. Recently, Wang et al. (2020b) proposed a new dataset, KB-Ref, to evaluate the reasoning ability of a model over not only visual and spatial features but also commonsense knowledge. The dataset is devised such that at least one piece of fact from a knowledge base (KB) is required for a target object (referred to by an expression) to be identified.

Therefore, searching for appropriate facts from a KB is also crucial part in KB-Ref. In contrast to the only existing work (Wang et al., 2020b), in which for each object candidate, the top-K facts with the highest cosine similarity between the averaged Word2Vec (Mikolov et al., 2013) embedding of the fact and the given expression are maintained, our framework focuses on multi-modal embedding and reasoning simultaneously over both the expression and the image to identify the top-K facts. Multi-modal features encode richer information helping to improve reasoning over varying (semantic) contexts and identification of relevant facts; for example, the above example of expression can be answered with the object “banana” in an image (or, equivalently, with the object “mashed potato” in another image).

In this paper, we propose a novel multi-modal framework for KB-Ref – Commonsense Knowledge Enhanced Transformers (CK-Transformer, CK-T for short) – that integrates (top-K) facts into all object candidates in an image for better identification of the target object. Specifically, our contributions are four-fold: 1) We propose the CK-T (see Figure 3.1) that effectively integrates diverse input from different modalities: vision, referring expressions and facts; 2) To the best of our knowledge, our approach is the first that introduces visual information into the identification of (top-K) relevant facts; 3) Our approach achieves a new state of the art using only top-3 facts per (candidate)

object, which is furthermore substantially more efficient compared to existing work utilizing as much as top-50 facts; 4) We introduce facts into ‘conventional’ REC tasks, leading to improved performance.

3.2 Related Work

Referring expression comprehension with commonsense knowledge Different from conventional REC tasks (see Appendix A.1 for details), KB-Ref focuses on querying objects given an expression that requires commonsense knowledge reasoning. The authors benchmarked a baseline model, ECIFA, for integration of facts, expression and image, and selects the target object by comparing the match scores between the image features and corresponding top-K fact features for all object candidates in the image (Wang et al., 2020b). In our framework, we select top-K facts for each candidate by comparing the cosine similarity between the fact and expression embedding, where the embeddings are generated from a multi-modal encoder rather than a text encoder used in the ECIFA model.

Pre-trained vision–language encoders Several pre-trained multi-modal encoders (Chen et al., 2020c; Li et al., 2019; Su et al., 2019; Tan and Bansal, 2019a) have been proposed, achieving state-of-the-art results on vision–language tasks. Currently, UNITER (Chen et al., 2020c) as one of powerful pre-trained encoders achieves the best performance on REC tasks (RefCOCO). In this paper, we adapt UNITER such that it is used as a multi-modal encoder in fact search and as part of the CK-T.

3.3 Methodology

We formulate KB-Ref as a classification problem based on an image I consisting of a set of candidates (image regions) $I = \{c_j\}_{j=1}^n$ obtained from either ground-truth labels or predictions of a pre-trained object detector. Specifically, given an expression e , an image I and a KB, we first search for top-K facts $F_i^K = \{f_j\}_{j=1}^k$ from the KB for each candidate c_i , and then feed e , I , and F_I^K (the selected facts over I) into our CK-T simultaneously to predict the target object over all candidates.

3.3.1 Image-based fact search

For each candidate c_i in a given image, we retrieve all the facts from the KB (see Appendix A.4 for details on the KB used in our framework) according to its category (e.g., a candidate object may belong to category ‘car’). Then, we calculate the cosine similarity between the facts and the given expression, where the similarities are obtained from a similarity extractor which we train by adapting UNITER. Specifically, given image–expression and image–fact pairs as input, we extract expression and fact features respectively from the position of the cross-modality output of UNITER (corresponding

to the input of [CLS] token, see Appendix A.2 for details), and then calculate the cosine similarity between the two. During training, inspired by Devlin et al. (2018), we replace 50% of ground-truth facts with random facts from the KB (with a similarity of 0), to help the model better distinguish useful facts from non-useful ones. Finally, we maintain top-K facts F_i^K with higher similarities to the expression for each candidate c_i .

3.3.2 Commonsense Knowledge Enhanced Transformer

The CK-T consists of a bi-modal encoder (see 3.3.2) and a fact-aware classifier (see 3.3.2).

Bi-modal encoder

The bi-modal encoder (initialized by UNITER-base with N=12 layers (Chen et al., 2020c)) integrates two modalities: image and text (e or f_i). Specifically, after generating the input embedding E_{Inp} consisting of image and text embedding (same with UNITER, see Appendix A.3 for details), for each candidate c_i , we extract the expression-aware and fact-aware object features respectively (f_i^e and f_i^f) from the position of the visual output corresponding to c_i in the same encoder, based on the input of all candidates I , and e or f_i .

Fact-aware classifier

The fact-aware classifier is composed of multi-head attention layers and fully connected layers. For each candidate c_i , f_i^e and F_i^f (all K fact-aware object features for c_i) are fed into the integrator simultaneously (*Key* and *Value* are from F_i^f , and *Query* is from f_i^e), and fused into one three-source object features f_i^t (image, expression and top-K facts).

Finally, f_i^t is mapped into a match score s_i for c_i by a linear layer, and the optimization objective is to minimize the cross-entropy loss over all scores $\{s_j\}_{j=1}^n$ corresponding to all candidates I .

3.4 Results

We compare our CK-T to existing approaches on KB-Ref task without and with facts. Then we explore the importance of introducing visual information for fact search. Furthermore, we introduce facts into the traditional RefCOCO dataset, which was collected from MSCOCO (Lin et al., 2014) but differs in the types of expressions and object candidate settings. We extract image region features using an off-the-shelf detector, Faster R-CNN with ResNet-101 (Ren et al., 2015), based on bounding boxes (bbxes) (ground-truth labels or predicted results from the detector). See Appendix A.4 and A.5 for details about these datasets and experiment setting¹. Through parameter

¹Including the efficiency discussion about our model

Model	Accuracy (%)	
	Val	Test
CMN (Hu et al., 2017)	41.28	40.03
SLR (Yu et al., 2017)	44.03	42.92
VC (Niu et al., 2019)	44.63	43.59
LGARNs (Wang et al., 2019c)	45.11	44.27
MAttNet (Yu et al., 2018)	46.86	46.03
ECIFA-nf (Wang et al., 2020b)	37.95	35.16
CK-T-nf (Ours)	58.02	57.53
ECIFA (Wang et al., 2020b)	59.45	58.97
MAtt+E (Wang et al., 2020b)	64.08	63.57
CK-T-Word2Vec	60.40	61.39
CK-T-Uw/oImage	64.44	64.78
CK-T (Ours)	65.62	66.71
Human	—	90.31

Table 3.1: Accuracy on KB-Ref dataset without and with facts (top and bottom part, respectively) using ground-truth bounding boxes and object categories.

search on K and M (see Figure 1 and 2 in Appendix A.6), we keep $M = 2$ Fact-aware classifier blocks and top-3 facts for each candidate.

Ground-truth bounding boxes and categories By following Wang et al. (2020b), we report our results on KB-Ref without and with facts. As can be seen in Table 3.1 (top), CK-T-nf, a version of CK-T without facts², achieves an accuracy of 57.53% on the test set, outperforming existing approaches that do not utilize facts by approximately 11% – 22%. At the bottom part of the table we can see that our fact-enhanced CK-T model achieves the highest accuracy (66.71%) on the test set, which is 7.74% higher than that of ECIFA (a baseline model proposed by Wang et al. (2020b)), and 3.14% higher than MAtt+E³. It is worth noting that both ECIFA and MAtt+E incorporate the top-50 facts for each candidate, which is considerably higher compared to top-3 facts in our CK-T. We surmise this is due to the fact that our fact search approach utilizes multi-modal fact and expression embeddings.

Predicted bounding boxes and categories To facilitate a fair comparison with ECIFA-d (Wang et al., 2020b), we also use the maximum 10 detected bbxs for each image (CK-T-m10). As can be seen in Table 3.2, CK-T-m10 achieves an accuracy which is $\approx 5\%$ higher than that of ECIFA-d on the test set. CK-T-m100, a variant using at most 100 detected bbxs achieves a substantial improvement with $\approx 7\%$, compared with

²all word tokens in fact sentences are replaced with only one [MASK] token.

³Wang et al. (2020b) introduces their facts fusion module –Episodic Memory Module (E)–into MAttNet model (Matt) (Yu et al., 2018) widely used for conventional REC.

Model	Accuracy (%)	
	Val	Test
ECIFA-d (Wang et al., 2020b)	24.11	23.82
CK-T-m10 (Ours)	28.33	28.71
CK-T-m100 (Ours)	35.66	35.96

Table 3.2: Accuracy on KB-Ref using predicted bbxs and object categories.

CK-T-m10. This difference is primarily due to the increase in the number of correctly detected bbxs and predicted categories. Specifically, we find that with the top-100 bbxs, the number of samples containing the target bbxs rises from 18,901 to 31,653, while among these target bbxs, the number of correctly predicted categories grows from 11,324 to 15,928, out of a total of 43,284 samples in the KB-Ref dataset. This can also explain the dramatic decline on the accuracy between CK-T and CK-T-m10.

Incorporating image features into fact search We experiment with various approaches to fact search and evaluate their effectiveness on KB-Ref (Table 3.1). We first utilize top-k facts searched in (Wang et al., 2020b), where they use a pre-trained Word2Vec (Skip-Gram) (Mikolov et al., 2013) for searching facts (CK-T-Word2Vec). Then, we also selected facts from similarity predictors based on only text as input (CK-T-Uw/oImage)⁴, instead of image-text pairs in CK-T. As shown in Table 3.1, both CK-T-Uw/oImage and CK-T achieve better accuracy on the test set compared to CK-T-Word2Vec. Compared to CK-T-Uw/oImage, CK-T achieves around 2% higher accuracy. This is primarily due to the additional visual information used during fact search (see Appendix A.9 for the examples of the selected facts by these fact search methods).

Introducing facts in traditional REC tasks We incorporate facts from the KB used in KB-Ref into the tasks of RefCOCOs using CK-T. Table 3.3 shows the results based on the ground-truth bbxs and categories (discussion about the detected results can be seen in Appendix A.7). Compared with U_{REC} ⁵, the model introducing facts achieves better or equal accuracy on all RefCOCOs tasks, where RefCOCOg is improved more than RefCOCO and RefCOCO+. This is because RefCOCOg has less same-category object candidates in an image compared to RefCOCO and RefCOCO+ (an average of 1.63 and 3.9 per image, respectively) (Yu et al., 2016), and thus the retrieved facts integrated into different candidates are diversified (we first retrieve facts using the category), which contributes to distinguishing between candidates. This difference can also be proved in McNemar Test, where we find the change in the proportion of errors is statistically significant after introducing facts as compared to before on RefCOCOg (p -value = $1.19e-08 < \alpha = 0.05$), while the similar proportions are found on RefCOCO

⁴Inspired by Frank et al. (2021b), we replace all object candidate feature with the average of all image region features.

⁵Chen et al. (2020c) achieve state-of-the-art results on RefCOCOs by finetuning UNITER. (We re-finetune the model for fair comparison and conduct McNemar Test)

Task		Accuracy (%)	
		U_{REC}	Intro Facts
Ref-COCO	Val	90.98	91.43
	Test A	91.50	92.09
	Test B	90.89	90.95
Ref-COCO+	Val	83.23	83.45
	Test A	85.09	85.49
	Test B	79.08	79.08
Ref-COCOg	Val	86.23	87.21
	Test	85.79	87.59

Table 3.3: Introducing facts into RefCOCO, RefCOCO+ and RefCOCOg. RefCOCO and RefCOCO+ have two different test sets, Test A and Test B, containing multiple persons and multiple objects in images respectively.

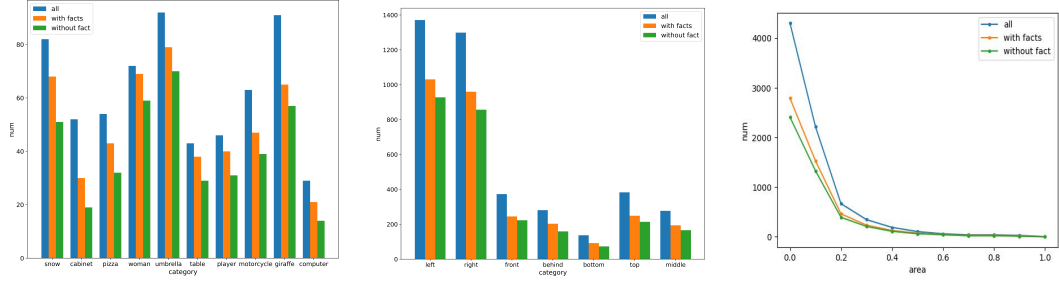
and RefCOCO+ (see Appendix A.8 for details about McNemar Test). The overall impact of commonsense knowledge in traditional REC is, however, not substantial. This is primarily due to much smaller number (78) of categories among the candidates in RefCOCOs, compared to 1805 in the KB-Ref (Wang et al., 2020b). This limits the variety of selected facts, therefore impacting the extent to which commonsense knowledge is useful.

3.5 Analysis

To investigate in what cases commonsense knowledge helps, we conduct a fine-grained analysis of model performance on the test set of KB-Ref. Specifically, we compare the samples predicted by model with and without facts (CK-T and CK-T-nf) on three aspects: object categories, spatial relationships and the size of the bounding box.

Object categories The test set contains 1502 categories and CK-T outperforms CK-T-nf on 1347 categories. Top 10 categories for which most improvement is observed are shown in Figure 3.2a. In case of the 155 categories that do not show improvement, we find that the average number of samples per category is 6.68, making the results less reliable.

Spatial relationships We then investigate to what extent solving the REC task with and without facts relies on spatial reasoning, and whether there are particular spatial relationships between objects for which the use of facts is most crucial. Similar to the works of (Johnson et al., 2017; Kazemzadeh et al., 2014), we focus on the following spatial relationships: *left*, *right*, *front*, *behind*, *bottom*, *top*, *middle*. As shown in Figure 3.2b, the model with facts (CK-T) outperforms that without facts (CK-T-nf) on all spatial relationships.



(a) Top 10 categories showing most improvement after introducing facts. (b) The analysis of spatial relationships. (c) The analysis of different bounding box sizes.

Figure 3.2: Fine-grained analysis. *all*: the total number of samples in the test set; *with fact*: the number of test samples that CK-T predicts correctly; *without fact*: the number of test samples that CK-T-nf predicts correctly.

The size of the bounding box We then investigate the role of facts when identifying objects of different sizes, using the size of their bounding box as a proxy. We use the normalized area of the bounding box as the metric of bboxes size. As shown in Figure 3.2c, the facts improve model performance on all bounding box sizes.

3.6 Conclusion

In this paper, we proposed CK-Transformer, which effectively integrates commonsense knowledge and the expression into the representations of the corresponding visual objects for multi-modal reasoning on KB-Ref. Our CK-Transformer achieves a new state-of-the-art performance on KB-Ref using only top-3 most relevant facts. We also demonstrated that visual information is beneficial for fact search. Finally, we show that commonsense knowledge improves conventional REC tasks across three different datasets.

3.7 Limitations

The computational requirements of our model are affected by the number of facts. Specifically, we train our CK-Transformer for 10000 steps with a batch size of 64 on one Titan RTX GPU, which takes 2.5, 3, 3.5, 7 days with the number of facts: 3, 5, 10, 20 respectively. The CK-Transformer processes 3.8, 2.8, 2.1, 0.7 samples on average per second at training time and 8.3, 7.3, 6.6, 1.1 samples per second at test time, with these amounts of facts. The computational requirements of our models are thus substantial, and future work should consider improving computational efficiency and thus reducing environmental impact.

Chapter 4

Gradient-based Parameter Selection for Efficient Fine-Tuning

Chapter Highlights With the growing size of pre-trained models, full fine-tuning and storing all the parameters for various downstream tasks is costly and infeasible. Parameter-efficient fine-tuning (PEFT) arise as an promising method for this challenge. In this chapter, we will investigate the parameter-efficient fine-tuning (PEFT) through the perspective of the sparse training as sparse training can provide a more precise and fine-grained tuning. Specifically, we propose a new parameter-efficient fine-tuning method, **Gradient-based Parameter Selection (GPS)**, demonstrating that only tuning a few selected parameters from the pre-trained model while keeping the remainder of the model frozen can generate similar or better performance compared with the full model fine-tuning method. Different from the existing popular and state-of-the-art parameter-efficient fine-tuning approaches, our method does not introduce any additional parameters and computational costs during both the training and inference stages. Another advantage is the model-agnostic and non-destructive property, which eliminates the need for any other design specific to a particular model.

4.1 Introduction

The pre-training and fine-tuning pipeline has become a common paradigm for adapting large models pre-trained on substantial amounts of data to downstream tasks with fewer training samples. However, fine-tuning all the parameters in the model is memory-intensive and data-inefficient, which is costly and infeasible for multiple downstream tasks given a large-scale model (Houlsby et al., 2019; Jia et al., 2022a; Lian et al., 2022). To tackle this issue, parameter-efficient fine-tuning (PEFT) methods have been proposed with the aim of tuning a minimal number of parameters to fit downstream tasks while keeping most of the parameters frozen. Another benefit of PEFT is that tuning a smaller set of parameters reduces the complexity of optimization and alleviates overfitting concerns when adapting large pre-trained models to downstream tasks with

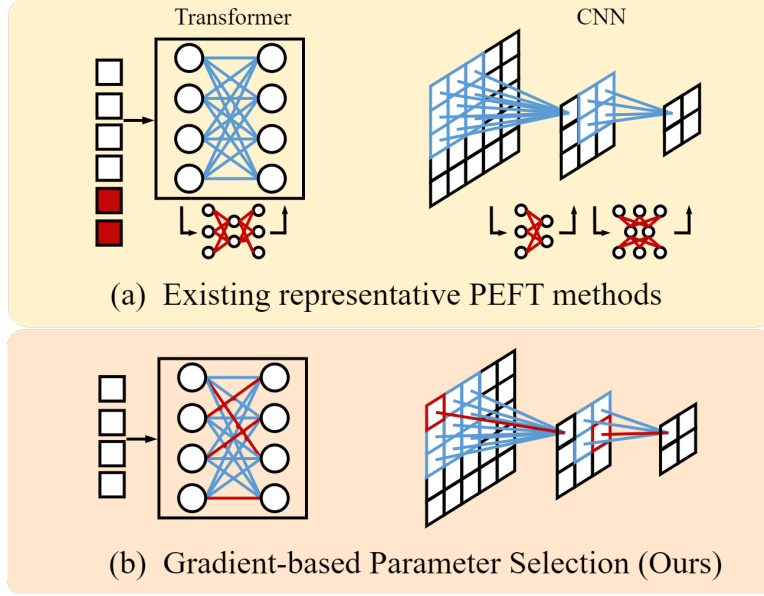


Figure 4.1: Comparison between our GPS and other PEFT methods. (a) Existing popular methods introduce extra parameters for tuning downstream tasks, which might need a special design for diverse architectures, such as appending prompt into the input token in Transformer or inserting different modules into different layers (b) Our approach avoids the introduction of additional parameters and solely fine-tunes the selected parameters from the model, employing a unified gradient-based parameter selection method across diverse architectural variations, e.g. Transformer and CNN.

limited data, resulting in comparable or even superior performance compared to full fine-tuning (Jia et al., 2022a). Inspired by the success of PEFT in NLP (Ding et al., 2023; He et al., 2021a; Hu et al., 2021a, 2022c; Li and Liang, 2021; Su et al., 2021), several methods have been introduced to vision tasks, such as Adapter (Houlsby et al., 2019) and Visual Prompt Tuning (VPT) (Jia et al., 2022a) introducing extra learnable parameters into the backbone and the input space of the pre-trained model respectively. SSF, another representative method, transforms features across layers of the pre-trained model using extra learnable layers (Lian et al., 2022).

However, these methods introduce additional parameters into the pre-trained model and disrupt its original architecture, leading to increased computational costs during training and/or inference stages. Furthermore, these approaches lack generalizability across various model architectures. Specifically, different models are equipped with distinct components (layers), such as MLPs, activation functions, and self-attention layers. These methods need to determine the optimal locations for inserting extra parameters between different layers; moreover, certain transformer-based techniques cannot be directly applied to convolution-based methods like VPT. Therefore, these methods exhibit limited compatibility with diverse architectures.

To tackle the above issues, we propose a non-destructive network architecture and

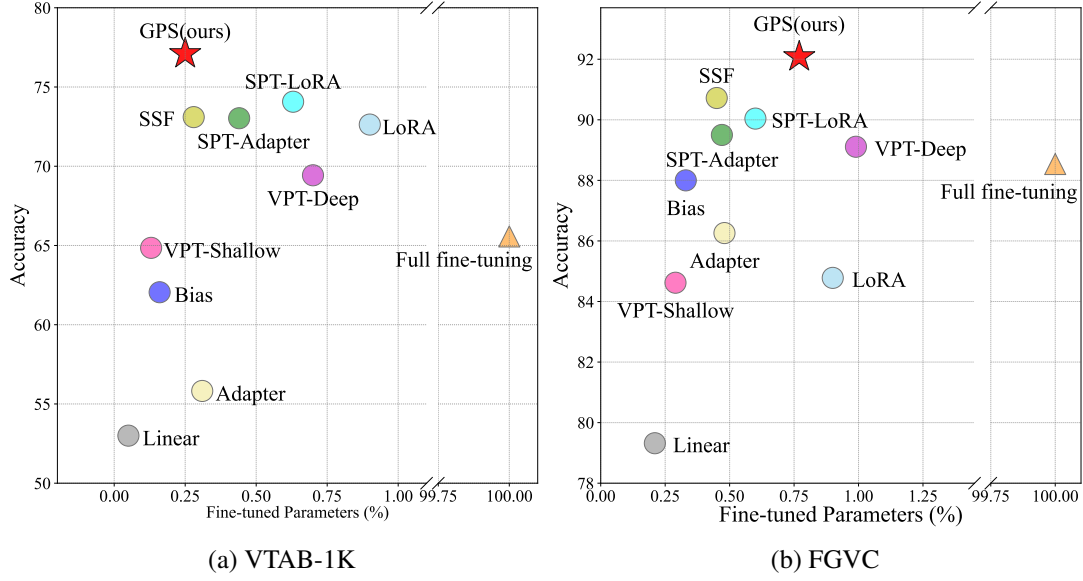


Figure 4.2: Performance comparisons of 11 fine-tuning methods with a pre-trained ViT-B/16 model on the VTAB-1k (a) and FGVC (b) benchmarks. Our GPS (red stars) achieves state-of-the-art performance on both benchmarks with only 0.25% and 0.77% average trainable parameters respectively.

model-agnostic PEFT approach, which introduces no extra parameters during both training and test stages and provides a unified solution for various architectures. We select a small number of essential parameters from the pre-trained model and only fine-tune these parameters for the downstream tasks. To select them, we propose a fine-grained Gradient-based Parameter Selection (GPS) method. For each neuron in the network, we choose top-K of its input connections (weights or parameters) with the highest gradient value, resulting in a small proportion of the parameters in the model being selected.

Such design offers five-fold benefits: i) The pre-trained model can efficiently tackle downstream tasks because the gradient direction indicates the fastest loss function changes and highest change rate, facilitating efficient gradient descent during model fine-tuning. We also provide a sparse regularized equivalent form for GPS, which indicates better generalization than full fine-tuning; ii) Each neuron within the network possesses the potential to adjust its activation state by fine-tuning selected input connections. Consequently, the pre-trained model exhibits flexibility in modifying features of varying granularities to suit diverse downstream tasks. For instance, when adapting a model pre-trained on ImageNet (Deng et al., 2009) for CIFAR-100 (Rebuffi et al., 2017), it is necessary to refine high-level features; whereas for ImageNet-Sketch (Wang et al., 2019b) adaptation, more detailed feature fine-tuning is required. iii) Our approach avoids introducing extra parameters and computational costs and keeps the architecture of the model intact; iv) The selection procedure enables its application across diverse models

Method	Mean Acc.	Params. (%)	Model Agnostic	No extra Train param.	No extra Infer params.	Task Adaptive
Full (Jia et al., 2022a)	70.36	100	✓	✓	✓	✗
Linear (Jia et al., 2022a)	58.48	0.08	✓	✓	✓	✗
Bias (Zaken et al., 2021)	67.54	0.20	✓	✓	✓	✗
Adapter (Houlsby et al., 2019)	60.04	0.35	✗	✗	✗	✗
VPT (Jia et al., 2022a)	73.53	0.76	✗	✗	✗	✗
LoRA (Hu et al., 2021a)	75.16	0.90	✗	✗	✓	✗
SSF (Lian et al., 2022)	76.77	0.32	✗	✗	✓	✗
GPS (ours)	78.64	0.36	✓	✓	✓	✓

Table 4.1: Comparison between different fine-tuning methods. The ViT-B/16 model accuracy over all 24 tasks in FGVA and VTAB fine-tuned on ViT-B/16 model and the number of tunable parameters are shown in columns Acc. and Params. (%).

by adopting a neuron-based rather than a layer-based method, thereby eliminating the necessity for distinct designs for different layers in various models. v) Different from other methods using a pre-defined and consistent strategy for different tasks, our method adaptively selects parameters for each task by our proposed gradient strategy to better fit the domain-specific semantics of different downstream tasks. Please see the difference between our method with others in Figure 4.1 and Table 4.1.

We evaluate our approach on a total of 27 visual tasks (including image classification and semantic segmentation) over 4 different model architectures. Our GPS achieves state-of-the-art performance compared to other PEFT methods and has a good balance between performance and the number of trainable parameters, as illustrated in Figure 4.2. Compared with the full fine-tuning, GPS achieves 3.33% (FGVC) and 9.61% (VTAB) improvement of the accuracy while tuning only 0.36% parameters of the pre-trained model on average over 24 tasks; it also demonstrates a significant improvement of 17% and 16.8% in mDice and mIoU, respectively, on medical image segmentation task. Moreover, we verify the effectiveness of our approach on different network architectures, such as Transformer and Convolutional Neural Networks. Furthermore, we compare GPS with various parameter selection methods and demonstrate its superior properties. GPS provides a new paradigm for PEFT and inspires deeper insights into this field.

4.2 Related work

Visual parameter efficient fine-tuning In general, there are typically two primary categories of PEFT. Addition-based methods introduce additional parameters to the pre-trained backbone. Adapters (Gao* et al., 2021; Gao et al., 2023b; Houlsby et al., 2019; Pfeiffer et al., 2020a,b; Rebuffi et al., 2017; Stickland and Murray, 2019; Sung et al., 2022; Wang et al., 2020c; Zhang et al., 2021b, 2022a, 2023b) adopt a residual pathway and learn down and up projection with a nonlinear activation. Others (Mahabadi et al.,

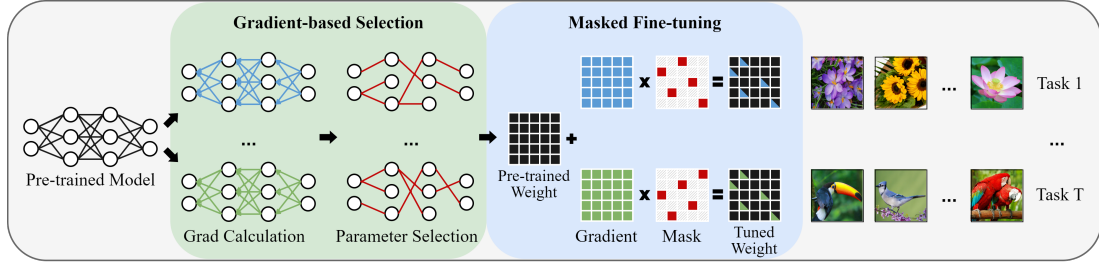


Figure 4.3: The overall pipeline of GPS. We first select a small portion of important parameters (sub-network) for each task from the original pre-trained model using a gradient-based method. Then only fine-tune the sub-network while keeping other parameters frozen.

2021) propose a hyper-network to generate model weights or decompose the dense weighted matrix into the low-rank matrix (Karimi Mahabadi et al., 2021). Prompt methods (Ding et al., 2021; Gao et al., 2020; Hu et al., 2021b; Ju et al., 2022; Li and Liang, 2021; Liu et al., 2023d, 2022b; Zhang et al., 2023c; Zhu et al., 2023) wrap the input with context. VPT (Jia et al., 2022a) prepend learnable prompts to the input tokens. SSF (Lian et al., 2022) achieves promising results by scaling and shifting the feature between layers. Selection-based methods select a subset of the parameters for tuning, such as only fine-tuning bias (Zaken et al., 2021), last K layers (Houlsby et al., 2019; Jia et al., 2022a). While traditionally considered less effective than addition-based methods, our approach of adaptively selecting parameters for each task yielded surprisingly strong results.

Sub-network training Pruning technique (Gale et al., 2019; Han et al., 2015a,b; Kruschke and Movellan, 1991; Li et al., 2016a; Wen et al., 2016) uncovers the importance of subnetworks. The lottery ticket hypothesis (Frankle and Carbin, 2018) articulates that subnetworks can reach the accuracy of the original model. Fine-tuning sub-networks are widely studied. SpotTune (Guo et al., 2019) designs a policy network to make routing decisions for subset networks. Child-tuning (Xu et al., 2021) iteratively updates a subset of parameters by masking out some gradients during the backward process. However, these methods are not aligned with the PEFT setting. We fix a small number of parameters and only tune them for fitting downstream to achieve PEFT.

4.3 Approach

Different from the currently popular methods introducing additional parameters to fine-tune the pre-trained model for downstream tasks (Houlsby et al., 2019; Jia et al., 2022a; Lian et al., 2022), we select only a small number of parameters from the pre-trained model and then only update these parameters during the fine-tuning stage. Specifically, our method has two stages: parameter selection and masked fine-tuning. For each

downstream task t , we first select a small portion of important parameters (task-specific parameters) from the original pre-trained model using a gradient-based method. We then fine-tune the pre-trained model for the task t , keeping all other unimportant parameters frozen and updating only selected parameters using a sparse binary mask to set the gradient of unimportant parameters to zero (see Figure 4.3).

4.3.1 Gradient-based parameter selection

Relevant studies have indicated that the pre-trained backbone exhibits diverse feature patterns at distinct parameter positions, and the same positions make varying contributions to fine-tuning various tasks (Chatterji et al., 2019; Kumar et al., 2022; Naseer et al., 2021; Raghu et al., 2021; Yosinski et al., 2014). Therefore, we posit that there exists an optimal subset of parameters for fine-tuning a pre-trained model to a downstream task. This subset is essential and necessary for fine-tuning the task, and the different tasks require a distinct subset. Formally, given a downstream task t with the dataset \mathcal{D}_t and a pre-trained model $\Theta = \{w_1, w_2, \dots, w_N\}$, we aim to find a subset of w , i.e. $w = \{w_1, w_2, \dots, w_n\}$ ($n \ll N$). we select parameters following two principles: 1) Important for downstream tasks; 2) Distributed over the whole network.

Importance for downstream tasks We identify the importance of parameters in a pre-trained model for a specific task by selecting those with the highest gradient value, which is obtained by calculating the gradient of a loss function with respect to its parameters. The intuition behind this is that the parameters with the largest gradient value indicate the loss function changes fastest along the gradient direction and has the greatest change rate, which facilitates efficient gradient descent during fine-tuning. Specifically, the gradient of the parameters is calculated by

$$\nabla \mathcal{L}_{\mathcal{D}_t}(\Theta) = \left[\frac{\partial \mathcal{L}}{\partial w_1} \dots \frac{\partial \mathcal{L}}{\partial w_N} \right]^\top \quad (4.1)$$

where $\mathcal{L}(w)$ is the loss function. Normally, when we fine-tune a pre-trained model on a downstream task, we need a new classification head (*i.e.* MLP) with random initialization. In order to avoid the adverse effects of these randomly initialized parameters on gradient calculation using the cross-entropy loss function, we use Supervised Contrastive Loss (SCL) (Khosla et al., 2020) as the loss function for calculating the gradient during parameter selection, since it does not need to involve the head (We still use cross-entropy loss during fine-tuning stage). SCL is a variant of Contrastive Loss (CL) that aims to bring different augmented samples of the same image closer together in embedding space. In contrast, SCL tries to cluster samples from the same class together, which coincides with our target of the downstream classification tasks. Specifically, given a task with the dataset $\mathcal{D}_t = \{\mathbf{x}_i, y_i\}_{i=1 \dots K}$, SCL is calculated by

$$\mathcal{L}^{\text{scl}} = \sum_{i \in \mathcal{D}_t} \mathcal{L}_i^{\text{scl}} = \sum_{i \in \mathcal{D}_t} \frac{-1}{|P(i)|} \sum_{p \in P(i)} \log \frac{\exp(\mathbf{z}_i \odot \mathbf{z}_p / \tau)}{\sum_{a \in A(i)} \exp(\mathbf{z}_i \odot \mathbf{z}_a / \tau)} \quad (4.2)$$

where i represents i^{th} sample in \mathcal{D}_t ; $P(i) \equiv \{p \in A(i) : \tilde{y}_p = \tilde{y}_i\}$ is subset of \mathcal{D}_t , in which all samples have the same class with i ; $A(i) \equiv \mathcal{D}_t \setminus \{i\}$; z is the feature extracted from the pre-trained encoder and $\tau \in \mathcal{R}^+$ is a scalar temperature parameter.

Equivalent with sparse regularization In the above, we implicitly assume that the order of $\left(\left\|\frac{\partial \mathcal{L}}{\partial w_1}\right\| \cdots \left\|\frac{\partial \mathcal{L}}{\partial w_N}\right\|\right)$ is the same as $(\|w'_1 - w_1\| \cdots \|w'_N - w_N\|)$, which means selecting parameters with top-n gradient norm is the same as selecting top-n of the fine-tuning changes. Therefore, GPS captures the top-n important parameters for downstream tasks. The optimization objective can be rewritten as

$$\Theta' = \min \mathcal{L}(\Theta') \quad s.t. \quad \|\Theta' - \Theta\|_0 \leq n \quad (4.3)$$

where $\|\cdot\|_0$ is the l_0 norm and Θ' is the fine-tuned model. By Lagrangian duality, solving the above problem is equivalent to solving the following problem:

$$\mathcal{L}(\Theta') + \lambda \|\Theta' - \Theta\|_0 \quad (4.4)$$

with appropriate λ . Hence, GPS can be reviewed as a sparse regularized fine-tuning, which may lead to better generalization. [Fu et al. \(2023\)](#) demonstrate that Equation (4.4) has smaller generalization bound than pure optimization toward \mathcal{L} with full fine-tuning, resulting in better performance.

Distribution over the whole network A simple idea for parameter selection is to select a certain percentage of parameters with the highest gradient from the entire network. Our experiments have shown that with this idea, the majority of the selected parameters are located in the top layers of the network (see Supplementary for details), which is consistent with the findings reported in ([Houlsby et al., 2019](#); [Howard and Ruder, 2018](#)). However, solely fine-tuning these top-layer parameters is insufficient to mitigate the impact of the pre-trained model’s own inductive bias, particularly when there exist substantial disparities in data distributions between upstream and downstream tasks, which need to fine-tune more detailed features from shallower layers. Motivated by various studies indicating the distinct roles played by different components of neural networks ([Cao et al., 2022](#); [Fan et al., 2020](#); [Proakis and Manolakis, 1992](#); [Shrikumar et al., 2017b](#); [Wang et al., 2021b](#)), we posit that when fine-tuning a pre-trained model for downstream tasks, the adjusted parameters should be distributed throughout the entire network. The reason behind this lies in the ability of the model to adapt the information stored in parameters at different levels of granularity to fit downstream tasks. Therefore, our strategy is that for each neuron in the network, we select top-K (at least one) connections (weights) among all the input connections of the neuron, as shown in Figure 4.3. By doing so, every neuron within the network possesses the potential to fine-tune its activation state rather than solely adjust high-level information in the top layers. In other words, our approach fine-tunes the detailed information stored in each neuron of the network, which better fits the downstream task during the fine-tuning stage. Our exploratory experiment further substantiates this assertion, as shown in Table 4.6(a).

Combining the two points above, we first calculate the gradient of the loss with respect to all the weights in the models for a specific task. Then for each neuron in the network, we select top-K connections with the highest gradient value (the modulus of gradient) among all input connections to the neuron. Doing so can not only ensure that important parameters for downstream tasks are chosen and allow the model to tune the activation state of all neurons for better fitting of downstream tasks. Another benefit of this selection procedure is its ease of application across various model architectures, such as Transformer and CNN, avoiding any model-specific design. Our experiments also demonstrate the effectiveness of our approach across diverse architectures, as shown in Table 4.2 and Table 4.4.

4.3.2 Masked fine-tuning

After parameter selection for a specific task, we fine-tune the pre-trained model on the task. During fine-tuning, we only update the selected parameters while keeping the remaining parameters of the pre-trained model frozen. As our selected parameters are distributed across all neurons in every layer, only a few parameters within a specific weight matrix of the network are chosen, resulting in the updated matrix being sparse. Therefore, we utilize a mask to help with the sparse training. Specifically, for j^{th} weight matrix $\mathbf{W}_j \in \mathbb{R}^{d_{in} \times d_{out}}$ in the network, we build a same size of binary mask $\mathbf{M}_j \in \mathbb{R}^{d_{in} \times d_{out}}$:

$$\mathbf{M}_j = \begin{cases} 1, & w_j^k \in \mathbf{w} \\ 0, & w_j^k \notin \mathbf{w} \end{cases} \quad (4.5)$$

where w_j^k represents k^{th} element in j^{th} weight matrix. For each element in \mathbf{M}_j , its value is set to 1 if the corresponding parameter in \mathbf{W}_j is selected, and 0 otherwise. Then the weight matrix is updated by

$$\mathbf{W}_j \leftarrow \mathbf{W}_j - \epsilon \nabla \mathcal{L}(\mathbf{W}_j) \odot \mathbf{M}_j \quad (4.6)$$

where $\nabla \mathcal{L}(\mathbf{W}_j)$ is the gradient of the cross-entropy loss with respect to \mathbf{W}_j . As a result, the gradients of unselected parameters are zeroed out and excluded from updates, while only a small number of our selected parameters are updated during fine-tuning for downstream tasks. Please see Figure 4.3 for a visualization of our method.

4.4 Experiments

We evaluate GPS on various downstream tasks, including image classification tasks and semantic segmentation tasks with different architectures. First, we briefly introduce our experimental settings, including datasets, backbones, and baselines. Then we demonstrate the effectiveness and universality of GPS. Moreover, we systematically study the impacts of different selection schemes and conduct comprehensive ablation experiments.

Dataset	CUB -2011	NA- Brids	Oxford Flowers	Stan. Dogs	Stan. Cars	Mean Acc.	Params. (%)
Full (Jia et al., 2022a)	87.3	82.7	98.8	89.4	84.5	88.54	100.00
Linear (Jia et al., 2022a)	85.3	75.9	97.9	86.2	51.3	79.32	0.21
Bias (Zaken et al., 2021)	88.4	84.2	98.8	91.2	79.4	88.40	0.33
Adapter (Houlsby et al., 2019)	87.1	84.3	98.5	89.8	68.6	85.66	0.48
LoRA (Hu et al., 2021a)	85.6	79.8	98.9	87.6	72.0	84.78	0.90
VPT-Shallow (Jia et al., 2022a)	86.7	78.8	98.4	90.7	68.7	84.62	0.29
VPT-Deep (Jia et al., 2022a)	88.5	84.2	99.0	90.2	83.6	89.11	0.99
SSF (Lian et al., 2022)	<u>89.5</u>	<u>85.7</u>	<u>99.6</u>	89.6	<u>89.2</u>	<u>90.72</u>	0.45
SPT-Adapter (He et al., 2023a)	89.1	83.3	99.2	91.1	86.2	89.78	0.47
SPT-LoRA (He et al., 2023a)	88.6	83.4	99.5	<u>91.4</u>	87.3	90.04	0.60
GPS (Ours)	89.9	86.7	99.7	92.2	90.4	91.78	0.77

Table 4.2: Performance comparisons on FGVC with ViT-B/16 models pre-trained on ImageNet-21K.

4.4.1 Experimental settings

Datasets Following VPT (Jia et al., 2022a) and SSF (Lian et al., 2022), we evaluate our GPS method on a series of datasets categorized into three groups: i) **FGVC**: Fine-Grained Visual Classification (FGVC) benchmark includes 5 downstream tasks, which are CUB-200-2011 (Wah et al.), NABirds (Van Horn et al., 2015), Oxford Flowers (Nilsback and Zisserman, 2008), Stanford Dogs (Khosla et al., 2011) and Stanford Cars (Gebru et al., 2017). ii) **VTAB-1k**: Visual Task Adaptation Benchmark (Zhai et al., 2019) (VTAB) contains 19 visual classification tasks which are grouped into three sets: Natural, Specialized, and Structured. iii) **CIFAR-100** (Krizhevsky et al., 2009) and **ImageNet-1k** (Deng et al., 2009): widely use for general image classification task.

Backbones For a fair comparison, we follow VPT and SSF by using ViT-B/16 (Dosovitskiy et al., 2020b) pre-trained on ImageNet-21K (Deng et al., 2009) for the main image classification experiments. Moreover, to demonstrate the universality of our GPS, we also explore other backbones, including Swin Transformer (Liu et al., 2021d) and ConvNeXt-B (Liu et al., 2022c) for another variant of Transformer-based and CNN-based architecture, respectively. In addition, we finetune semantic segmentation tasks on SAM (Kirillov et al., 2023a), a strong segmentation foundation model.

Baselines We compare our GPS with a variety of fine-tuning protocols that can be mainly categorized into three types: i) **Full**: Full fine-tuning is the most commonly used protocol updating all parameters of the whole model during tuning. ii) **Selection-based**: This kind of method selects a subset of parameters in the original model for fine-tuning, including linear probing and Bias (Zaken et al., 2021). Such methods are easy to implement and require no extra computations but have not worked well. Our method

Method \ Dataset		Natural							Specialized				Structured							VTAB		
		CIFAR-100	Caltech101	DTD	Flowers102	Pets	SVHN	Sun397	Patch Camelyon	EuroSAT	Resisc45	Retinopathy	Clevr/count	Clevr/distance	DMLab	KITTI/distance	dSprites/loc	dSprites/ori	SmallNORB/azi	SmallNORB/ele	Mean Acc.	Mean Params. (%)
Full (Jia et al., 2022a)		68.9	87.7	64.3	97.2	86.9	87.4	38.8	79.7	95.7	84.2	73.9	56.3	58.6	41.7	65.5	57.5	46.7	25.7	29.1	65.57	100.00
Linear (Jia et al., 2022a)		63.4	85.0	64.3	97.0	86.3	36.6	51.0	78.5	87.5	68.6	74.0	34.3	30.6	33.2	55.4	12.5	20.0	9.6	19.2	53.00	0.05
Bias (Zaken et al., 2021)		72.8	87.0	59.2	97.5	85.3	59.9	51.4	78.7	91.6	72.9	69.8	61.5	55.6	32.4	55.9	66.6	40.0	15.7	25.1	62.05	0.16
Adapter (Houlsby et al., 2019)		74.1	86.1	63.2	97.7	87.0	34.6	50.8	76.3	88.0	73.1	70.5	45.7	37.4	31.2	53.2	30.3	25.4	13.8	22.1	55.82	0.31
LoRA (Hu et al., 2021a)		68.1	91.4	69.8	99.0	90.5	86.4	53.1	85.1	95.8	84.7	74.2	83.0	66.9	50.4	81.4	80.2	46.6	32.2	41.1	72.63	0.90
VPT-Shallow (Jia et al., 2022a)		77.7	86.9	62.6	97.5	87.3	74.5	51.2	78.2	92.0	75.6	72.9	50.5	58.6	40.5	67.1	68.7	36.1	20.2	34.1	64.85	0.13
VPT-Deep (Jia et al., 2022a)		78.8	90.8	65.8	98.0	88.3	78.1	49.6	81.8	96.1	83.4	68.4	68.5	60.0	46.5	72.8	73.6	47.9	32.9	37.8	69.43	0.70
SSF (Lian et al., 2022)		69.0	92.6	75.1	99.4	91.8	90.2	52.9	87.4	95.9	87.4	75.5	75.9	62.3	53.3	80.6	77.3	54.9	29.5	37.9	73.10	0.28
SPT-ADAPTER (He et al., 2023a)		72.9	93.2	72.5	99.3	91.4	88.8	55.8	86.2	96.1	85.5	75.5	83.0	68.0	51.9	81.2	51.9	31.7	41.2	61.4	73.03	0.44
SPT-LoRA (He et al., 2023a)		73.5	93.3	72.5	99.3	91.5	87.9	55.5	85.7	96.2	85.9	75.9	84.4	67.6	52.5	82.0	81.0	51.1	30.2	41.3	74.07	0.63
GPS (Ours)		81.1	94.2	75.8	99.4	91.7	91.6	52.4	87.9	96.2	86.5	76.5	79.9	62.6	55.0	82.4	84.0	55.4	29.7	46.1	75.18	0.25

Table 4.3: Performance comparisons on VTAB-1k with ViT-B/16 models pre-trained on ImageNet-21K.

belongs to this group and achieves the best performance while ensuring convenience and universality. iii) **Addition-based**: This kind of method adds new trainable parameters to the backbone, including Adapter (Houlsby et al., 2019), VPT (Jia et al., 2022a) and SPT-Adapter (He et al., 2023a). Such methods require extra computations in both the training and inference stages. Other methods like LoRA (Hu et al., 2021a), SSF (Lian et al., 2022), and SPT-LoRA (He et al., 2023a) also add new tunable parameters during the training stage, but these parameters can be reparameterized into the backbone during testing.

Implementation details We follow SSF to process the images in all the FGVC, VTAB-1k and CIFAR-100 datasets. We employ the Adam (Kingma and Ba, 2014) optimizer with cosine learning rate decay to fine-tune models for 100 epochs, and the linear warm-up is used in the first 10 epochs. All experiments are conducted on the NVIDIA A100 GPU.

4.4.2 Performance on image classification

We present a comprehensive evaluation of the effectiveness of our GPS by comparing it against multiple baselines on 3 benchmarks, comprising a total of 26 datasets. In addition to common benchmarks (FGVC and VTAB-1k), we also compare our method with others on different architectures. We evaluate the performance and effectiveness by Top-1 accuracy (%) and the number of fine-tuned parameters.

Image classification performance As shown in Table 4.2 and Table 4.3, our GPS outperforms all other fine-tuning methods by a large margin on both FGVC and VTAB benchmarks, sufficiently demonstrating that our method of parameter selection is a simple yet effective way for model tuning. On FGVC, GPS outperforms all other fine-tuning methods, including full fine-tuning, on all 5 tasks. It obtains 1.02% and

Architecture	Swin-B		ConvNeXt-B	
	Ave. Acc.	Params.(%)	Ave. Acc.	Params.(%)
Full (Jia et al., 2022a)	92.42	100.00	93.04	100.00
Linear (Jia et al., 2022a)	87.90	0.28	88.00	0.28
SSF (Lian et al., 2022)	91.54	0.56	92.48	0.56
GPS (Ours)	92.56	0.95	93.32	0.90

Table 4.4: Performance comparisons on FGVC benchmark (Average accuracy over 5 tasks) with different model architectures.

3.24% accuracy improvement of the mean accuracy compared to the previous SOAT method SSF (Lian et al., 2022) and full fine-tuning, while it only uses 0.77% of trainable parameters. On VTAB, GPS also outperforms all other fine-tuning methods. Specifically, it obtains 1.11% and 9.61% improvement of the mean accuracy on 19 VTAB tasks compared to the previous SOAT method SPT-LoRA (He et al., 2023a) and full fine-tuning. GPS beats the previous SOTA by 1.75%, 0.23%, and 0.63% in the Natural, Specialized and Structured subsets, respectively. Meanwhile, GPS also uses fewer trainable parameters compared to VPT-Deep, SSF, and SPT-LoRA (0.25% vs. 0.70%, 0.28% and 0.63%), which further illustrates the high efficiency of our approach. For most tasks, we exclusively select the top 1 input connection for each neuron; however, for more challenging tasks, multiple connections are chosen (see Supplement for details). The percentage of learnable parameters in our GPS can be explicitly controlled by adjusting the number of connections selected, allowing for a balance between parameter count and performance in tasks.

Generalization on different architectures Since our method only selects a subset of parameters from the pre-trained model for fine-tuning, it is naturally model-agnostic. We compare GPS with other representative methods across ViT-B/16 (Table 4.2), Swin-B and ConvNeXt-B architectures on the FGVC dataset (Table 4.4), CIFAR-100 and ImageNet-1k (Please see full results in Supplementary). Among all three architectures, GPS consistently outperforms all other baselines, demonstrating its model-agnostic advantage. The Swin and Convnext have more complex designs than ViT, enabling them to acquire comprehensive and high-quality features during pre-training. Consequently, even the simplest linear probing method yields commendable results on these two architectures, reducing the effectiveness of the PEFT method and causing the previous SOTA SSF to underperform Full. However, in this scenario, our GPS still maintains a lead over Full with gains of 0.12% and 0.28%, respectively, further showing the effectiveness of our method.

Computational cost In Figure 4.4, we compare the computational cost of GPS with other fine-tuning methods to demonstrate the efficiency of our approach. Following SSF (Lian et al., 2022), we reimplement VPT (Jia et al., 2022a) with 200 and 50 prompts

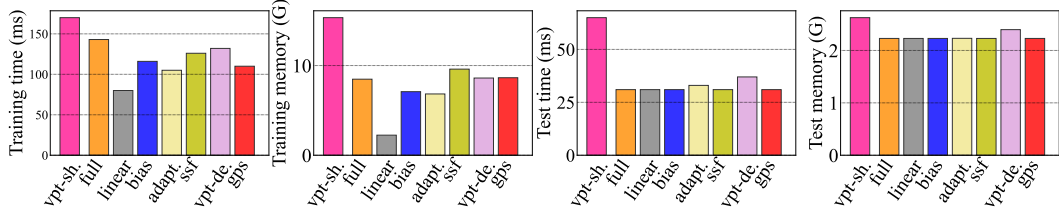


Figure 4.4: Computational cost of different tuning methods. From left to right: training time, training memory, test time, and test memory. Training/Test time is the time consumed by a mini-batch.

Method	mDice (\uparrow)	mIoU (\uparrow)	Params. (M)
Full (Jia et al., 2022a)	71.1	55.7	93.8
Linear (Jia et al., 2022a)	71.6	46.6	4.06
Bias (Zaken et al., 2021)	86.5	69.1	4.16
Adapter (Chen et al., 2023)	84.8	66.7	4.12
SSF (Lian et al., 2022)	87.3	71.7	4.26
GPS (Ours)	88.1	72.5	4.22

Table 4.5: Quantitative Result for Polyp Segmentation

for the shallow and deep versions, respectively. A batch size of 32 is used in both the training and inference stages. For a fair comparison, for all experiments, we do not use mixed precision training, which was used in SSF. All metrics are measured on a single NVIDIA A100 GPU. In the training stage, GPS has less time and memory consumption than both VPT and SSF. Compared with full fine-tuning, GPS has a much lower time overhead and a similar memory overhead, but it leads to an increased performance by a large margin. Since GPS is a selection-based method, it does not introduce any additional parameters, so it can achieve the same minimum time and memory overhead as full fine-tuning during inference without any reparameterization operation, which is much lower than the addition-based Adapter and VPT.

4.4.3 Semantic segmentation

In addition to visual classification tasks, we also explore our method for the task of semantic segmentation. Segment Anything Model (SAM) (Kirillov et al., 2023a) is a strong foundation model for segmentation. It is pre-trained on a large-scale dataset enabling powerful generalization. However, several studies, e.g. (Chen et al., 2023), have reported poor performance of SAM on medical segmentation tasks such as polyp segmentation (Jha et al., 2020a). To address this limitation, they proposed employing Adapter to effectively fine-tune SAM for downstream medical segmentation tasks.

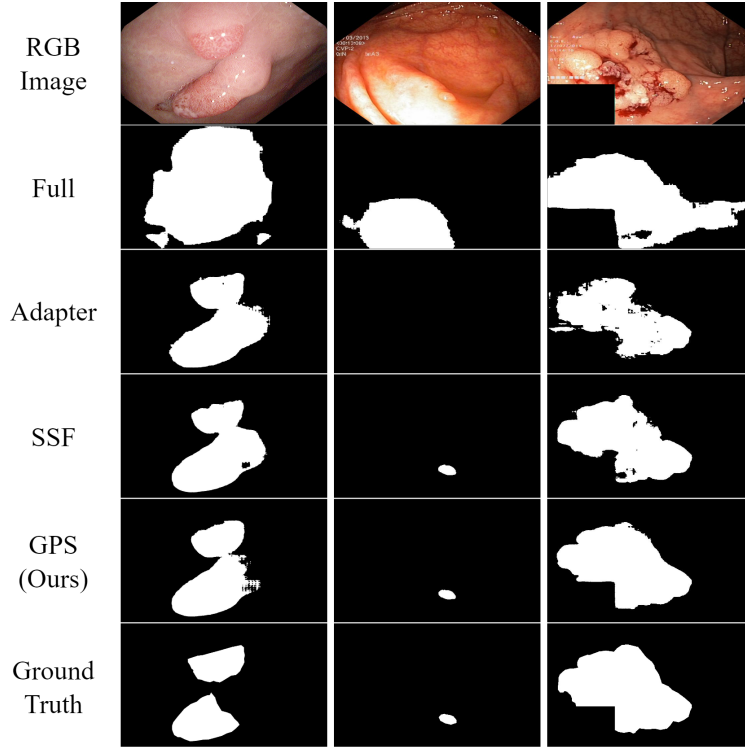


Figure 4.5: The Visualization of Polyp segmentation task. Our GPS can still handle difficult segmentation cases compared with others.

Following their experimental setup, we applied our method to SAM and conducted a comparative analysis with other PEFT approaches. Our GPS yielded exceptional results, as shown in Table 4.5 and visually depicted in Figure 4.5 (See Supplementary for more case visualization).

4.4.4 Impacts of different selection schemes

Different selection levels Our GPS selects trainable parameters at the neuron level, i.e. selecting top-k input connection per neuron. We also investigate parameter selection methods at different levels. As shown in Table 4.6 (a), *Net* and *Layer* represent selecting a certain proportion of the parameters with the highest gradient based on the entire network and each layer, respectively. For a fair comparison, we keep the same number of parameters selected over these levels. We can see that the finer the granularity of selection, the better the performance. For example, the accuracy on CUB increases by 0.44% and 0.77% when selection level changes from network to layer, and from layer to neuron.

Different selection criteria We further study the effectiveness of our gradient-based selection method by comparing different selection criteria. As shown in Table 4.6 (b), *Net Random* and *Neuron Random* means randomly selecting top-K the input connection

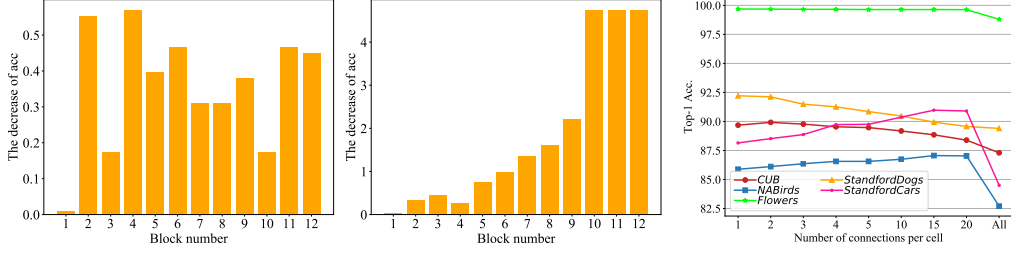


Figure 4.6: Impacts of different selection locations and quantities. From left to right: (a) Performance drop caused by not selecting parameters from k -th blocks. (b) And by not selecting from the top k blocks. (c) Impacts of different numbers of selected connections on performance.

Dataset		CUB	NABirds	Flowers	Cars	Dogs
(a)	Net	86.86	86.55	99.62	89.65	91.32
	Layer	87.30	86.79	99.64	90.03	91.90
(b)	Net Random	86.60	85.98	99.61	89.10	91.34
	Neuron Random	87.17	86.02	99.62	89.52	91.82
	Magnitude	87.29	85.99	99.62	89.29	91.30
(c)	Head+CE	87.05	86.20	99.64	89.25	91.29
GPS		88.07	86.64	99.69	90.10	92.30
		± 0.11	± 0.03	± 0.01	± 0.10	± 0.10

Table 4.6: The result on FGVC for investigating impacts of different selection schemes and ablations. (a) Different selection levels. (b) Different selection criteria. (c) Gradient calculating method.

for each neuron and selecting the same number of parameters based on the whole network respectively. *Magnitude* means selecting top-K input connections with the largest weight per neuron. As we can see, the increase in the randomness of parameter selection causes a decrease in performance (*Net Random* < *Neuron Random*). The result of *Magnitude* is similar to *Neuron Random*, demonstrating neuron-level selection is crucial.

Different selection location To investigate the impact of selected parameters located at different layers within the network, we conducted experiments using the ViT-B/16 model fine-tuning on CUB and evaluated accuracy degradation when applying our GPS method to select parameters from the entire network except for a specific transformer block or previous several transformer blocks. As shown in Figure 4.6a, it is surprising to note that when we do not select parameters from a specific block, the biggest drop in the accuracy comes from the shallow layers (block 2 and block 4). This finding supports

our GPS approach that selects parameters from the entire network rather than just the last few layers. When we do not select the parameters from the first specific number of blocks, it is observed that the accuracy drop increases with more blocks removed (Figure 4.6b).

4.4.5 Ablation study

Head-free contrastive loss To obtain more accurate gradients for selecting parameters, inspired by the representation learning pre-training methods, Our GPS adopts the supervised contrastive loss to calculate gradient (without random initialization of the classification head). As shown in Table 4.6 (c), when we use the cross-entropy loss (with the head) to calculate the gradient, the average accuracy on FGVC is dropped by 0.67%, illustrating the importance of obtaining accurate gradients.

Selected connection number As shown in Figure 4.6c we select top-K input connections per neuron as trainable parameters, ranging from 1 to 15, and conduct experiments on the 5 tasks. We can observe that more trainable parameters do not necessarily lead to better performance, but each data set has a performance peak. In addition, on the dataset with sufficient training data, the addition of trainable parameters can greatly improve the accuracy. Our GPS can easily control the number of trainable parameters and achieve optimal results on each dataset.

Robustness to seeds Addition-based fine-tuning methods like VPT are sensitive to the initialization of additional parameters as well as random seeds, whereas select-based methods are not. All results in Table 4.6 are the average accuracy of three seeds on FGVC datasets (Only shows the std of GPS here. Please see details in supplementary). The results show random seeds have little influence on our method.

4.5 Conclusion

In this paper, we propose a new paradigm for PEFT, *i.e.* Gradient-based Parameter Selection (GPS). Our approach does not introduce any additional parameters and only fine-tunes a small subset of the pre-trained model’s parameters for downstream tasks, resulting in robust generalization across diverse models and adaptively selecting a subset of parameters for each task. Remarkably, GPS achieves significant improvement on a range of tasks (including image classification and semantic segmentation), compared to the full fine-tuning method. GPS also attains SOTA performance compared to other PEFT methods.

Chapter 5

GPU-Efficient Sparse Training for Parameter-Efficient Fine-Tuning

Chapter Highlights In this chapter, we further explore parameter-efficient fine-tuning through sparse training on larger models, specifically addressing the challenge of GPU memory consumption, which is a common issue encountered with sparse training. Existing parameter-efficient fine-tuning (PEFT) methods can generally be categorized into two types: addition-based methods and selective in-situ adaptation. The former, such as LoRA, introduce additional modules to adapt the model to downstream tasks, providing strong memory efficiency. However, these methods often have limited representational capacity, making them less suitable for fine-grained adaptation. On the other hand, selective in-situ adaptation directly fine-tunes a carefully selected subset of the original model parameters, enabling more precise and effective adaptation, but at the cost of significantly increased memory consumption. To address this trade-off, we propose a novel PEFT method that combines the benefits of sparse training (fine-grained model fine-tuning) with the memory efficiency characteristic of addition-based methods, thereby achieving both high adaptation precision and memory efficiency.

5.1 Introduction

Large language models (LLMs) demonstrate remarkable generalization capabilities, yet achieving optimal performance on downstream tasks often still requires fine-tuning. As model sizes grow, full-parameter fine-tuning becomes increasingly impractical due to substantial computational and memory demands. For example, fine-tuning LLaMA 2-13B without CPU offloading requires 26 GB for trainable parameters in FP16, 52 GB for Adam optimizer states (two FP32 moments per parameter), 26 GB for gradients, and an additional 2–4 GB for activations depending on batch size and sequence length. This results in a memory footprint of approximately 106–108 GB in total, far exceeding the capacity of commodity GPUs and necessitating premium hardware (e.g., A100 80G). This highlights the pressing need for more efficient and scalable adaptation strategies.

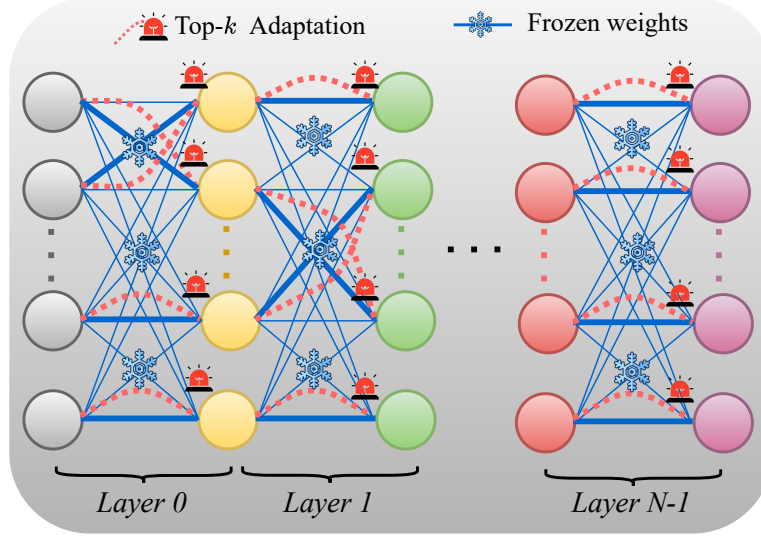


Figure 5.1: Overview of NeuroAda. For each neuron, only the top- k weights are adapted, while the rest remain frozen. Bold blue indicates selected pretrained weights; red dashed edges represent newly introduced trainable parameters.

A growing body of work on parameter-efficient fine-tuning (PEFT) addresses the computational and memory overhead of full-model adaptation by introducing a set of trainable parameters while keeping the backbone frozen. One major class of these methods is known as *addition-based adaptation*, which augments the pretrained model with additional modules designed to inject task-specific flexibility. These additions vary in form and location, including adapter layers inserted into projection blocks (Pfeiffer et al., 2020a; Sung et al., 2022), nonlinear activation reparameterizations (Zhang et al., 2021b), prompt tuning applied to input embeddings (Liao et al., 2023a,b; Lin et al., 2020), latent representation perturbations (Wu et al., 2024b), and low-rank matrix decompositions applied directly to weight spaces, such as in LoRA (Hu et al., 2022a) and its variants (Kopiczko et al., 2023; Zhang et al., 2023a). These methods typically offer improved memory efficiency by restricting gradient computation and optimizer state updates to the newly introduced modules, as opposed to the entire model in full fine-tuning. However, their scalability is constrained: as model size increases, the limited representational capacity of the added modules often leads to diminishing returns (He et al., 2024).

Another prominent line of research is *selective in-situ adaptation*, which fine-tunes a carefully selected subset of a pretrained model’s original parameters, without introducing any additional parameters, modules, or layers. Structure-based approaches, such as BitFit (Ben Zaken et al., 2022) and Partial- k (Jia et al., 2022b) updating only the bias terms and the last k layers, respectively, exemplify early efforts in this direction. More recently, fine-grained and unstructured parameter selection methods have attracted increasing attention. These approaches aim to identify task-relevant parameters at a more granular level, such as GPS (Zhang et al., 2024b) and SPT (He et al., 2023b), which

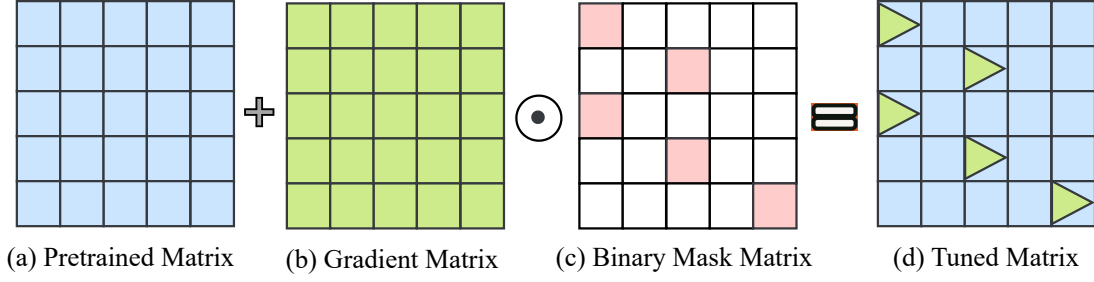


Figure 5.2: Mask-based sparse tuning employs a binary mask matrix to suppress gradient updates for unselected parameters. However, this approach incurs significant memory overhead, as gradients for the entire original parameter matrix must still be computed and retained by the optimizer.

demonstrate strong performance on vision tasks by selectively fine-tuning subsets of parameters that are most critical for the target task. Compared to structured approaches and *addition-based adaptation* methods, these unstructured strategies offer greater flexibility in parameter selection, enabling more precise and targeted model adaptation, and thereby substantially improving downstream performance (Fu et al., 2023; Shen et al., 2024). However, this sparse tuning paradigm leads to higher memory usage due to mask-based implementations. As shown in Figure 5.2, although only a small portion of the parameters is selected for updating, memory consumption remains comparable to that of full fine-tuning. This limitation becomes particularly problematic with the increase of model size (Shen et al., 2024; Zhai et al., 2022).

These limitations motivate us to explore whether a unified approach can be designed that achieves the fine-grained parameter tuning characteristic of selective in-situ adaptation, while retaining the memory efficiency advantages of addition-based methods. To this end, we propose **NeuroAda**, an additive, overlay-style adaptation method that utilizes a carefully designed approach to introduce new parameters to enable fine-grained adjustments while maintaining memory efficiency. Specifically, as shown in Figure 5.1, NeuroAda first selects the top- k highest-magnitude input connections (weights/parameters) for each neuron in the network prior to finetuning and then, for each selected parameter, a bypass connection (initialized to zero) is introduced. During finetuning, only the newly introduced parameters are fine-tuned, while the original parameters remain frozen. This approach inherits both the performance benefits of *selective in-situ adaptation* and the memory efficiency of *addition-based methods*. Crucially, for each neuron, at least one of its input connections is selected for update, ensuring that all neurons have the potential to modify their activation states and thus change the state of the entire network for effective adaptation. Consequently, NeuroAda presents a scalable and practical solution for large language models.

NeuroAda offers four key advantages that make it both practical and effective in real-world scenarios: (1) **Highly efficient computation:** NeuroAda eliminates the need of the mask for sparse finetuning, which typically requires full gradient computation.

(2) **Highly efficient GPU memory usage:** Only the newly added parameters are updated, significantly lowering memory usage by avoiding optimizer state tracking for the full model. (3) **Task-agnostic and generalizable:** Parameter selection is based on weight magnitudes from the pretrained model, enabling consistent selection across tasks, making the method broadly applicable and easy to deploy. (4) **Fine-grained, neuron-level adaptation:** NeuroAda ensures every neuron has the potential to change the activation state of each neuron during finetuning, maximizing the representational expressiveness of individual neurons. Empirically, NeuroAda achieves state-of-the-art performance on 23+ tasks compared with other PEFT methods, including both natural language generation and understanding, highlighting its practical utility and effectiveness.

5.2 Related Work

Addition-based Adaptation. Includes adapter-based methods (He et al., 2021b; Liao et al., 2023a,b; Lin et al., 2020; Pfeiffer et al., 2020a) and low-rank reparameterization techniques such as LoRA (Hu et al., 2022a) and its variants AdaLoRA (Zhang et al., 2023a), VeRA (Kopiczko et al., 2023), QLoRA (Dettmers et al., 2024), and DoRA (Liu et al., 2024c), which introduce trainable low-rank matrices into projection layers. While LoRA avoids inference-time overhead by merging updates into base weights, it often suffers from scalability issues and diminishing returns when applied to large models or complex tasks (Liu et al., 2024c). A parallel direction modifies hidden states instead of weights: activation steering (Li et al., 2023b; Liu et al., 2023e), concept erasure (Avitan et al., 2024; Belrose et al., 2023b; Singh et al., 2024), and block-level editing (Wu et al., 2024b) offer instance-specific control but require task-specific adaptation.

Selective In-Situ Adaptation. This class of work fine-tunes a subset of the model’s original parameters without introducing any additional modules or weights, often achieving strong performance with minimal architectural changes. However, its practical memory and compute benefits frequently fall short in large-scale settings. Methods such as SIFT (Song et al., 2023), SHiRA (Bhardwaj et al., 2024), and SpIEL (Ansell et al., 2024) enforce sparsity constraints, yet still require full backward passes to compute gradients for the entire weight space. More targeted approaches, including SMT (He et al., 2024) and GPS (Zhang et al., 2024b), improve efficiency by selecting submatrices or top- k gradients per neuron, but rely on gradient-based warm-up and dynamic masking. These mechanisms introduce additional overhead from binary mask storage, dense optimizer states, and full-gradient tracking, making them difficult to scale to large language models. In contrast, our method, NeuroAda, inherits the advantages of both paradigms by avoiding gradient-based selection and selecting the top- k weights per neuron.

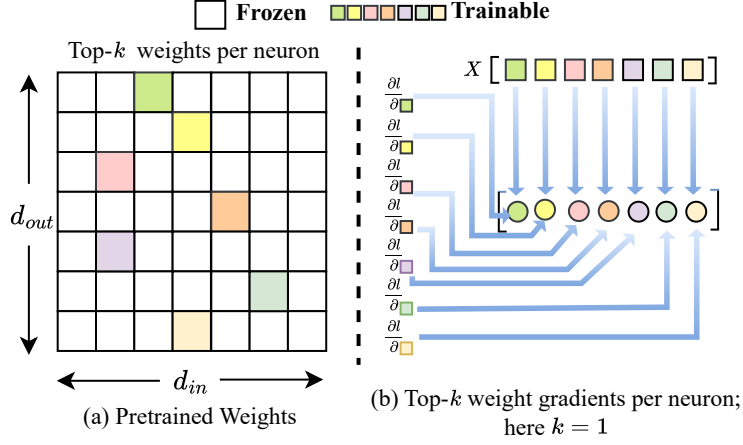


Figure 5.3: **Neuron-wise Top- k Weight Selection and Gradient Computation.** (a) Pre-trained weight matrix of size $d_{out} \times d_{in}$, where for each neuron (row), only the top- k weights (i.e., highest-magnitude) are selected for adaptation (colored), and the rest remain frozen (white). (b) Corresponding gradient matrix restricted to the top- k weights per neuron (here $k = 1$), showing gradients only for trainable entries. This strategy enables fine-grained, neuron-level adaptation while preserving most of the pretrained model, effectively activating each neuron’s potential through less-invasive tuning.

5.3 Methodology

In this section, we first present the necessary preliminaries, and then introduce NeuroAda, a new adaptation framework that activates each neuron’s potential by selectively updating a small subset of its weights. Specifically, we freeze all pretrained model weights and introduce sparse, additive overlay-style adaptation method in which top- k bypasses of input connections (weights/parameters) are introduced into each neuron in the neural network for adaptation. This neuron-wise adaption preserves the original parameters intact while enabling targeted learning signals at fine granularity. As illustrated in Figure 5.3, NeuroAda ensures that every neuron participates in adaptation, supporting both efficiency and generalization. During inference, the small number of learned deltas can be merged into the base weights, resulting in no additional overhead at runtime.

5.3.1 Preliminaries

Let \mathcal{M}_{Φ} be an L -layer pretrained language model with parameters $\Phi = \{\mathbf{W}^{(\ell)}, \mathbf{b}^{(\ell)}\}_{\ell=1}^L$. For any linear sub-layer we write $\mathbf{h}_{out} = \mathbf{W}\mathbf{h}_{in} + \mathbf{b}$, where $\mathbf{W} \in \mathbb{R}^{d_{out} \times d_{in}}$ and each *row* of \mathbf{W} corresponds to a **neuron**. During standard fine-tuning all entries of \mathbf{W} are updated, yielding heavy computational and memory costs (‘5.1).

Table 5.1: **Memory comparison per projection.** Mask-based sparse tuning methods require 1 bit per weight¹. NeuroAda with $k = 1$ only stores one BF16 value (2 bytes) and one index (2 bytes) per row, totaling 4 bytes per neuron. This yields over $100\times$ memory savings for a single linear layer.

Model	d_{model}	Mask [MB]	NeuroAda[MB]	Saving Ratio
LLaMA-1 7B	4096	$\frac{4096^2}{8 \times 2^{20}} \approx 2.00$	$\frac{4096 \times 4}{2^{20}} \approx 0.016$	$\approx 125\times$
LLaMA-2 7B	4096	2.00	0.016	$125\times$
LLaMA-1 13B	5120	$\frac{5120^2}{8 \times 2^{20}} \approx 3.13$	$\frac{5120 \times 4}{2^{20}} \approx 0.020$	$\approx 156\times$
LLaMA-2 13B	5120	3.13	0.020	$156\times$

Sparse additive updates. NeuroAda freezes Φ and introduces a *delta* “parameter tensor Δ with the same shape as Φ but sparsity constrained:

$$\Phi' = \Phi + \Delta, \quad \|\Delta\|_0 \ll \|\Phi\|_0, \quad (5.1)$$

where $\|\cdot\|_0$ counts non-zero elements. Only Δ is trainable; the base model remains intact, so Δ can be *merged in-place* after training, incurring zero inference overhead.

5.3.2 Top- k selection

A core design goal is that *every neuron receives at least a small learning signal*. For each neuron—that is, for each row $\mathbf{w} \in \mathbb{R}^{d_{\text{in}}}$ of a weight matrix, we identify the indices of its k largest-magnitude components:

$$\mathcal{I}(\mathbf{w}) = \arg \top k |w_j|_{j \in \{1, \dots, d_{\text{in}}\}}. \quad (5.2)$$

We then allocate trainable delta weights only at these positions:

$$[\Delta]_{i,j} = \begin{cases} \theta_{i,j} & \text{if } j \in \mathcal{I}(\mathbf{w}_i) \\ 0 & \text{otherwise,} \end{cases} \quad (5.3)$$

where $\theta_{i,j}$ is a trainable parameter defined only for $j \in \mathcal{I}(\mathbf{w}_i)$ and initialized to 0. For all other positions, $\Delta_{i,j}$ is fixed to zero and excluded from both optimization and memory storage. While NeuroAda uses weight magnitude for top- k selection, the framework is flexible: task-guided criteria such as gradient magnitude or random ticketing can be substituted into $\mathcal{I}(\cdot)$. We employ magnitude due to its task-agnostic stability and the advantage of requiring no warm-up or additional computation. This design choice is empirically validated in our ablation study, where magnitude-based selection achieves strong performance without relying on task-specific signals, as shown in Figure 5.7.

¹While 1-bit-per-weight is a theoretical lower bound for binary mask storage, actual implementations in PyTorch and other frameworks use byte-addressable storage (e.g., `BoolTensor`), leading to significantly higher memory overhead.

Mask-free implementation. Since the top- k sparsity pattern is determined *a priori*, Eq. equation 5.3 can be implemented without maintaining a full binary mask over the weight matrix. Instead, we store a compact list of *indices* and corresponding *BF16 values*“only k entries per row“eliminating the need for dense masking or indexing during training. This design leads to substantial memory savings and indexing efficiency. As shown in Table 5.1, for a single projection layer in LLaMA-2 13B, a 1-bit-per-weight binary mask requires over 3 MB of memory, while NeuroAda with $k=1$ uses only 0.02 MB“over **156** \times smaller. These savings scale across layers and are especially beneficial for high-throughput training on limited-memory devices.

5.3.3 Featherlight adaptation

During fine-tuning we optimize only $\{\theta_{i,j}\}$ while re-using the forward path of the frozen backbone. For a linear layer the forward pass becomes

$$\mathbf{h}_{\text{out}} = \underbrace{\mathbf{W}\mathbf{h}_{\text{in}}}_{\text{frozen}} + \underbrace{(\mathbf{P} \odot \Theta)}_{\text{trainable } \Delta} \mathbf{h}_{\text{in}}, \quad (5.4)$$

where \mathbf{P} is an index matrix with zeros everywhere except $[P]_{i,j} = 1$ when $(i, j) \in \mathcal{I}(\mathbf{w}_i)$, \odot denotes element-wise product, and Θ is the dense tensor of learnable $\theta_{i,j}$.²

Lightweight backward pass and optimizer states. During back-propagation, NeuroAda updates only the k selected coordinates per neuron. As a result, the dominant memory contributors in full-model training“BF16/FP32 gradients and the two FP32 moment estimates in the AdamW optimizer“are reduced proportionally by a factor of $\frac{k}{d_{\text{in}}}$. Because all delta parameters are stored directly in BF16 and no FP32 master weights are needed, the optimizer maintains only $2 \times k$ FP32 values per row instead of $2 \times d_{\text{in}}$. This yields a substantial memory reduction in the optimizer state maintained by AdamW. In standard dense fine-tuning, AdamW stores two FP32 moment estimates (first and second moments) for each trainable parameter, resulting in:

$$\text{AdamW Mem. (Masked): } 2 \times d_{\text{out}} \times d_{\text{in}} \times 4(\text{bytes}), \quad (5.5)$$

where 4 bytes denotes the size of a 32-bit float. In contrast, NeuroAda updates only k weights per row, so the optimizer state becomes:

$$\text{AdamW Mem. (NeuroAda): } 2 \times d_{\text{out}} \times k \times 4(\text{bytes}). \quad (5.6)$$

This reduces memory usage by a factor of $\frac{d_{\text{in}}}{k}$ per linear layer. For example, with $d_{\text{in}} = 5120$ and $k = 1$, this corresponds to a $5120 \times$ reduction in AdamW state memory.

²We implement this with fused scatter-add so the additional multiply is executed only on the k selected positions; no dense mask is materialised.

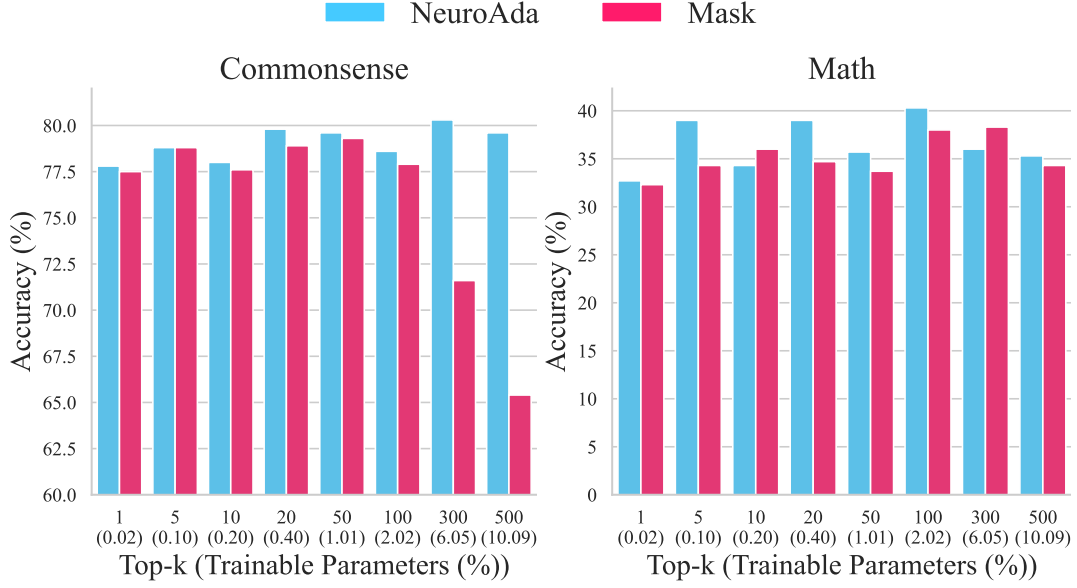


Figure 5.4: Performance comparison between our NeuroAda and mask-based methods on LLaMA-7B. Top- k means selecting top- k input connections per neuron in the neural network.

5.4 Neuron-wise Sparse Adaptation: Comparative Analysis

In this section, we first compare the mask-based method, which applies binary masks to zero out the gradients of unselected (frozen) parameters (see Figure 5.2), with our NeuroAda, which introduces new trainable parameters to bypass the selected ones, rather than directly tuning them, for sparse fine-tuning. The comparison is conducted in terms of effectiveness, GPU memory usage, and training efficiency. We then further investigate the effectiveness of the proposed method NeuroAda, which aims to ensure that all neurons in the network have the potential to update their activation states during fine-tuning. This is done by analyzing the proportion of neurons involved in fine-tuning and examining different parameter selection strategies for activating them.

Experiment setup To ensure a fair comparison between our NeuroAda and mask-based methods, as well as across different parameter selection strategies, we conduct a hyperparameter search over different the learning rates for each experiment using the training set. The best-performing configuration is then selected based on validation set performance. This is necessary because PEFT methods are generally sensitive to the choice of learning rate (Wu et al., 2024c). The hyperparameter search space is presented in Table 16 in Appendix. The details of used datasets: COMMONSENSE15K and GSM8K are provided in Appendix C.4.

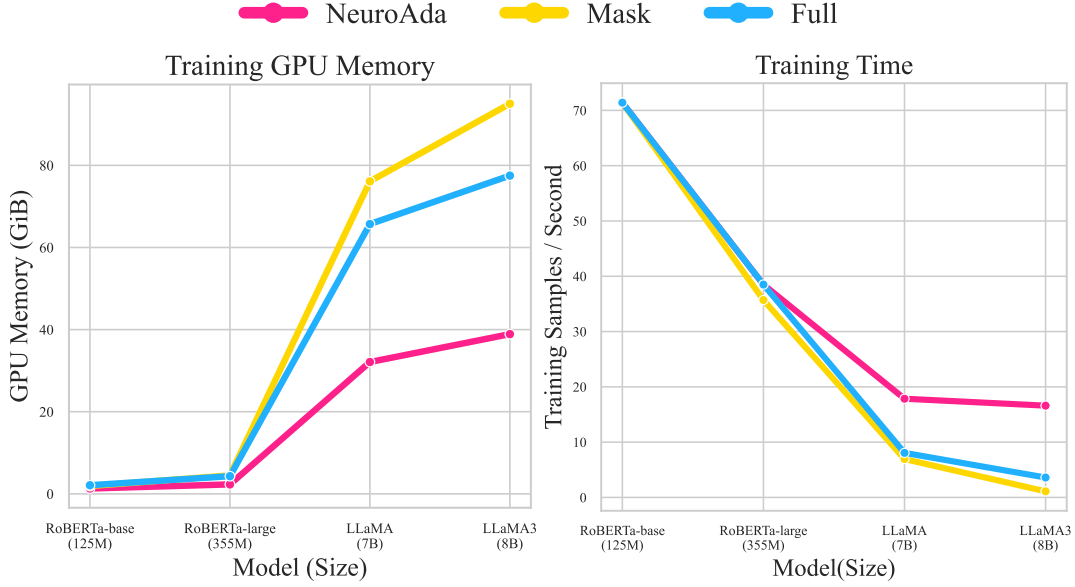


Figure 5.5: Training GPU memory and training efficiency on different models (RoBERTa-base, RoBERTa-large, LLaMA-7B, LLaMA3-8B) with NeuroAda, mask-based and full fine-tuning method.

Question 1: *Can our method NeuroAda be a competitive or even superior alternative to the mask-based sparse tuning approach?* We address this by comparing their task performance, GPU memory usage, and training efficiency.

Performance To comprehensively and fairly evaluate the effectiveness of the two methods, we compare them under the same proportion of trainable parameters, ranging from 0.02% to 10%, on the COMMONSENSE15K and GSM8K tasks. As shown in Figure 5.4, our proposed method NeuroAda performs comparably to, or even better than, the mask-based method across most parameter budgets on both datasets. In particular, the NeuroAda outperforms the mask-based method by approximately 9% and 14% in accuracy when using 6.05% and 10.09% of trainable parameters, respectively, on the commonsense reasoning task.

Training memory and time We evaluate models of varying sizes, including RoBERTa-base, RoBERTa-large (Liu et al., 2019c), LLaMA-7B (Touvron, 2023), and LLaMA3-8B (Vavekanand and Sam, 2024). Specifically, we sample 500 examples from the MNLI task in the GLUE natural language understanding benchmark (Wang et al., 2019a), and another 500 examples from the natural language reasoning task GSM8K. We use these samples to train the RoBERTa and LLaMA models, respectively. All experiments are conducted on a single NVIDIA H100 GPU with a batch size of 2. We report the GPU memory usage and training time for each model. Figure 5.5 shows

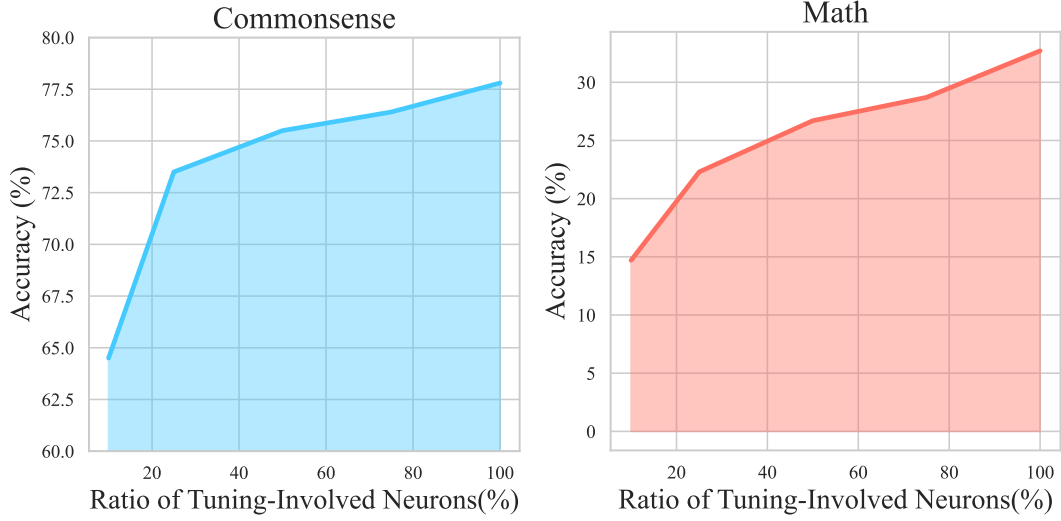


Figure 5.6: Comparison across different proportions of neurons involved in the fine-tuning process.

our proposed addition-based sparse training method NeuroAda consumes less GPU memory compared to the mask-based counterpart, especially as the model size increases. For example, the NeuroAda achieves up to 60% memory savings on LLaMA3-8B. In addition, the NeuroAda enables significantly faster training, particularly for larger models. It processes 16.6 samples per second, compared to only 1.1 samples per second with the mask-based method.

Question 2: *How effective is the proposed method NeuroAda in enabling all neurons to update their activation states during fine-tuning for downstream task adaptation?* To answer this question, we first investigate how different parameter selection strategies can be used to ensure that all neurons have the potential to update their activation states during fine-tuning. We further analyze how task performance on COMMONSENSE15K and GSM8K varies with the proportion of neurons allowed to update their activation states during training.

Involved number of neurons. Our proposed method selects the top- k input connections for each neuron in the network, ensuring that at least one input connection per neuron is selected for tuning. This design enables all neurons to update their activation states during fine-tuning, allowing better adaptation to downstream tasks. To demonstrate its effectiveness, we select parameters from various proportion of neurons per layer for tuning and evaluate the resulting performance on the COMMONSENSE15K and GSM8K tasks. As shown in Figure 5.6, increasing the number of neurons involved during training leads to consistent performance improvements on both tasks. This suggests that enabling all neurons to update their activation states is beneficial “and likely necessary” for effective downstream task adaptation.

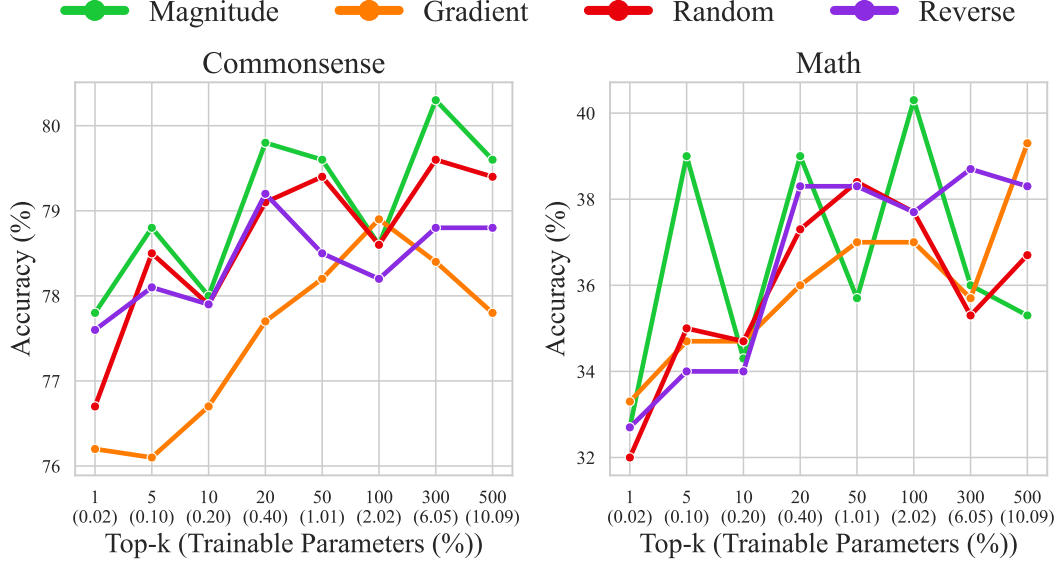


Figure 5.7: Comparison of different parameter selection strategies for involving neurons for the fine-tuning process. Among all input connections for each neuron in the network, Top- k connection with highest magnitude (*Magnitude*), highest gradient absolute value (*Gradient*), lowest magnitude (*Reverse*) are selected for training using addition-based method. *Random* means randomly selecting Top- k input connections per neuron.

Different selection strategies To explore the effectiveness of enabling all neurons in the network to update their activation states during training, we experiment with different parameter selection strategies for each neuron under different trainable parameter budget. Specifically, for each neuron, we select the top- k input connections based on four criteria: highest weight magnitude, highest gradient absolute value, lowest weight magnitude, and random selection from all input connections. As shown in Figure 5.7, all selection methods yield comparable performance on both the COMMONSENSE15K and GSM8K tasks across different trainable parameter budgets. The average accuracies across all budgets for all selection methods are closely aligned, ranging from 77.69% to 79.24% on COMMONSENSE15K, and from 35.89% to 36.54% on GSM8K. These results again highlight the importance of involving all neurons in the adaptation process, regardless of the specific selection method used. Additionally, across both tasks, the *Magnitude* selection method achieves the highest win rate across different parameter budgets compared to the other strategies. Therefore, we adopt the *Magnitude* selection strategy as the default in NeuroAda.

5.5 Experiments

We evaluate NeuroAda on 23+ datasets spanning commonsense reasoning (Section 5.5.1), arithmetic reasoning (Section 5.5.2), and natural language understanding (Section 5.5.3). Experiments cover both encoder-only (RoBERTa-base (Liu et al.,

Table 5.2: Performance comparison with existing PEFT methods on eight commonsense reasoning datasets and seven arithmetic reasoning datasets across four models: LLaMA-7B/13B, LLaMA2-7B, and LLaMA3-8B. *Most baseline results are taken from [Hu et al. \(2023a\)](#). †Results are from [Wu et al. \(2024c\)](#), ‡ results are taken from [He et al. \(2024\)](#) and *results are from [Liu et al. \(2024c\)](#), as they share the same experimental setting with [Hu et al. \(2023a\)](#). For a fair comparison, our NeuroAda is also trained for 3 epochs to align with these baselines. +When 3-epoch results are not available in the original paper, we re-trained the baselines using the official code and their reported best hyperparameters. All results for our method are averaged over three runs with different random seeds. Our method selects the top-20 and top-1 input connections per neuron for high-budget and low-budget parameter groups, respectively. The results of LLaMA3-8B are shown in Table 13 in Appendix due to space limitation.

Model	PEFT	Params (%)	Accuracy (↑)								
			Commonsense Reasoning								
			BoolQ	PIQA	SIQA	HellaS.	WinoG.	ARC-e	ARC-c	OBQA	Avg.
ChatGPT*	—	—	73.1	85.4	68.5	78.5	66.1	89.8	79.9	74.8	77.0
LLaMA (7B)	Series*	1.953%	63.0	79.2	76.3	67.9	75.7	74.5	57.1	72.4	70.8
	Parallel*	3.542%	67.9	76.4	78.8	69.8	78.9	73.7	57.3	75.2	72.3
	LoRA*	0.826%	68.9	80.7	77.4	78.1	78.8	77.8	61.3	74.8	74.7
	DoRA _{half} *	0.427%	70.0	82.6	79.7	83.2	80.6	80.6	65.4	77.6	77.5
	DoRA*	0.838%	68.5	82.9	79.6	84.8	80.8	81.4	65.8	81.0	78.1
	SMT†	0.840%	68.7	81.7	78.3	91.6	78.8	84.1	67.7	77.4	78.7
	NeuroAda	0.404%	73.1	85.4	80.9	94.3	84.3	87.8	71.7	84.2	82.7
	PrefT*	0.039%	64.3	76.8	73.9	42.1	72.1	72.9	54.0	60.6	64.6
	DiReFT*	0.031%	66.1	82.5	78.8	92.6	81.9	83.2	67.1	79.8	79.0
	LoReFT†	0.031%	68.3	83.5	79.7	92.7	82.6	83.2	67.4	78.5	79.5
	NeuroAda	0.020%	69.6	83.6	80.5	92.3	81.1	84.0	68.1	80.4	80.0
	NeuroAda	0.020%	69.6	83.6	80.5	92.3	81.1	84.0	68.1	80.4	80.0
LLaMA (13B)	Series*	1.586%	71.8	83.0	79.2	88.1	82.4	82.5	67.3	81.8	79.5
	Parallel*	2.894%	72.5	84.9	79.8	92.1	84.7	84.2	71.2	82.4	81.5
	LoRA*	0.670%	72.1	83.5	80.5	90.5	83.7	82.8	68.3	82.4	80.5
	DoRA _{half} *	0.347%	72.5	85.3	79.9	90.1	82.9	82.7	69.7	83.6	80.8
	DoRA*	0.681%	72.4	84.9	81.5	92.4	84.2	84.2	69.6	82.8	81.5
	SMT†	0.680%	71.1	84.4	81.7	93.7	83.2	86.7	73.7	85.2	82.4
	NeuroAda	0.327%	73.3	87.9	82.7	96.0	86.9	90.2	77.1	88.6	85.3
	PrefT*	0.031%	65.3	75.4	72.1	55.2	68.6	79.5	62.9	68.0	68.4
	DiReFT*	0.025%	70.2	86.6	82.5	95.0	85.2	86.3	73.5	84.4	83.0
	LoReFT†	0.025%	72.0	85.6	82.1	94.8	85.3	86.9	73.0	85.0	83.1
	NeuroAda	0.016%	73.0	86.4	82.2	94.5	84.0	87.6	74.5	86.0	83.5
	NeuroAda	0.016%	73.0	86.4	82.2	94.5	84.0	87.6	74.5	86.0	83.5
Llama2 (7B)	LoRA*	0.826%	69.8	79.9	79.5	83.6	82.6	79.8	64.7	81.0	77.6
	DoRA _{half} *	0.427%	72.0	83.1	79.9	89.1	83.0	84.5	71.0	81.2	80.5
	DoRA*	0.838%	71.8	83.7	76.0	89.1	82.6	83.7	68.2	82.4	79.7
	SMT†	0.840%	72.0	83.8	80.8	93.3	82.8	86.7	74.0	81.0	81.8
	NeuroAda	0.404%	73.6	86.5	81.1	94.8	87.8	89.1	75.9	85.6	84.3
	DiReFT*	0.031%	68.2	83.4	79.8	93.4	83.1	84.6	70.3	79.4	80.3
	LoReFT*	0.031%	66.6	81.8	79.3	93.4	82.6	83.0	70.2	80.8	79.7
	NeuroAda	0.020%	71.4	82.8	79.8	93.3	84.0	85.4	70.1	81.2	81.0
	NeuroAda	0.020%	71.4	82.8	79.8	93.3	84.0	85.4	70.1	81.2	81.0
	NeuroAda	0.020%	71.4	82.8	79.8	93.3	84.0	85.4	70.1	81.2	81.0
	NeuroAda	0.020%	71.4	82.8	79.8	93.3	84.0	85.4	70.1	81.2	81.0
	NeuroAda	0.020%	71.4	82.8	79.8	93.3	84.0	85.4	70.1	81.2	81.0
			Arithmetic Reasoning								
			MulAri	GSM8K	AddSub	AQuA	SinEq	SVAMP	MAWPS	Avg.	
GPT-3.5 _{175B} *	—	—	83.8	56.4	85.3	38.9	88.1	69.9	—	70.4	
LLaMA (7B)	Series*	1.953%	92.8	33.3	80.0	15.0	83.5	52.3	—	59.5	
	Parallel*	3.542%	94.5	35.3	86.6	18.1	86.0	49.6	—	61.7	
	LoRA*	0.826%	95.0	37.5	83.3	18.9	84.4	52.1	—	61.9	
	SMT†	0.860%	91.5	34.2	85.8	23.6	84.6	53.6	—	62.2	
	SMT†	1.260%	93.4	35.6	86.8	24.2	85.3	54.8	—	63.4	
	NeuroAda	0.404%	96.0	36.5	92.4	22.0	94.1	53.2	—	68.4	
	PrefT*	0.039%	—	24.4	—	14.2	—	38.1	63.4	35.0	
	DiReFT*	0.031%	—	20.5	—	21.3	—	39.9	68.1	37.5	
	LoReFT†	0.031%	—	21.6	—	22.4	—	43.6	69.5	39.3	
	NeuroAda	0.020%	—	30.3	—	22.8	—	48.9	77.7	44.9	
	NeuroAda	0.020%	—	30.3	—	22.8	—	48.9	77.7	44.9	
	NeuroAda	0.020%	—	30.3	—	22.8	—	48.9	77.7	44.9	
LLaMA (13B)	Series*	1.586%	93.0	44.0	80.5	22.0	87.6	50.8	—	63.0	
	Parallel*	2.894%	94.3	43.3	83.0	20.5	89.6	55.7	—	64.4	
	LoRA*	0.670%	94.8	47.5	87.3	18.5	89.8	54.6	—	65.4	
	NeuroAda	0.327%	97.5	43.9	92.2	21.7	93.9	61.4	—	71.4	
	PrefT*	0.031%	—	31.1	—	15.7	—	41.4	66.8	38.8	
	DiReFT*	0.025%	—	32.1	—	23.2	—	51.2	76.1	46.7	
	LoReFT†	0.025%	—	35.5	—	23.4	—	54.6	81.8	48.8	
	NeuroAda	0.016%	—	43.0	—	25.6	—	56.7	83.6	52.2	
	DiReFT*	0.031%	—	26.4	—	23.6	—	48.4	71.8	42.6	
	LoReFT*	0.031%	—	26.2	—	18.5	—	46.7	76.9	42.1	
	NeuroAda	0.020%	—	36.1	—	22.8	—	52.1	82.4	48.4	
	NeuroAda	0.020%	—	36.1	—	22.8	—	52.1	82.4	48.4	
	NeuroAda	0.020%	—	36.1	—	22.8	—	52.1	82.4	48.4	

2019c)) and decoder-only (LLaMA (Touvron et al., 2023b; Touvron, 2023)) models up to 13B parameters. We benchmark against strong PEFT baselines, including Bit-Fit (Zaken et al., 2021), prefix-tuning (Li and Liang, 2021), adapters (He et al., 2021b), LoRA (Hu et al., 2022b), DoRA (Liu et al., 2024c), SMT (He et al., 2024), RED (Wu et al., 2024b), DiReFT, and LoReFT (Wu et al., 2024c). Following LoReFT, all models use `torch.bfloat16` and run on a single NVIDIA A100 or H100 GPU.

Our comparison considers not only benchmark performance but also parameter efficiency. To demonstrate the robustness of our method under varying parameter budget constraints, we categorize all baseline methods into two groups based on the proportion of trainable parameters: those with $\geq 0.1\%$ and those with $< 0.1\%$. Note that these two groups differ by orders of magnitude in the number of trainable parameters, allowing us to assess performance across both relatively high and extremely low parameter budgets. We then compare our NeuroAda against these baselines under comparable levels of parameter efficiency to ensure a fair and meaningful evaluation. Specifically, we select the top-1 and top-20 input connections per neuron for matching the budget of the two groups, resulting in 0.016% and 0.327% trainable parameters on LLaMA-13B, respectively.

5.5.1 Commonsense reasoning

Following the experimental protocol of (Hu et al., 2023a), we fine-tune LLaMA-7B/13B, LLaMA2-7B, and LLaMA3-8B on COMMONSENSE170K, a composite dataset consisting of eight commonsense reasoning tasks, as described in Appendix C.1. Then, We evaluate each task on its test set and compare our results with baselines from Hu et al. (2023a), as well as DoRA (Liu et al., 2024c), DiReFT, LoReFT (Wu et al., 2024c), and SMT (He et al., 2024) under the same setting.

Hyperparameter tuning Inspired by Wu et al. (2024c), we use COMMONSENSE15K, a subset of COMMONSENSE170K, to perform hyperparameter search. The search space is detailed in Table 15. Specifically, we split COMMONSENSE15K into training and validation sets, as described in Section 5.4. Our hyperparameter search is conducted only on LLaMA-7B, and the best-performing configuration on the validation set is subsequently applied to all other models, including LLaMA-7B/13B, LLaMA2-7B, and LLaMA3-8B, for training on COMMONSENSE170K.

Results As shown in Table 5.2, our NeuroAda achieves state-of-the-art performance under both parameter budget regimes ($\geq 0.1\%$ and $< 0.1\%$). Notably, under the higher parameter budget setting, NeuroAda outperforms all baselines by a considerable margin. For example, its average accuracy surpasses the second-best baseline, SMT, by 4%. In addition, NeuroAda remains effective even under the lower parameter budget setting, consistently outperforming other baselines in this regime.

5.5.2 Arithmetic reasoning

Following [Hu et al. \(2023a\)](#) and [Wu et al. \(2024c\)](#), we fine-tune LLaMA-7B/13B, LLaMA2-7B, and LLaMA3-8B on MATH10K, a composite dataset comprising seven arithmetic reasoning tasks, and evaluate each task separately. The dataset details are provided in Appendix C.4.

Hyperparameter tuning Following [Wu et al. \(2024c\)](#), we perform hyperparameter search on the LLaMA-7B model using the GSM8K dataset, which is split into training and validation sets as in [Wu et al. \(2024c\)](#). The best-performing configuration on the validation set is then applied to all models, including LLaMA-7B/13B, LLaMA2-7B, and LLaMA3-8B for training on the MATH10K dataset. The full hyperparameter search space is provided in Table 14 in Appendix.

Results As shown in Table 5.2, NeuroAda consistently achieves the highest average accuracy across all model sizes and parameter budgets. Under a higher parameter budget (e.g., 0.327% on LLaMA-13B), it outperforms all baselines by a clear margin. For example, NeuroAda outperforms the second-best baseline, LoRA, by 6%, while using even fewer trainable parameters. Even under extremely low budgets (e.g., 0.020% on LLaMA2-7B), NeuroAda remains competitive and surpassing other low-budget baselines by up to 6% with even fewer trainable parameters.

5.5.3 Natural language understanding

We evaluate the effectiveness of our method on the GLUE benchmark ([Wang et al., 2019a](#)), a widely used suite of sequence classification tasks for evaluating natural language understanding (NLU), using the RoBERTa-base model. To ensure fair comparison, we follow the training, evaluation, and hyperparameter tuning procedures in [Wu et al. \(2024c\)](#).

Results We report the result in table 12 in the Appendix due to the space limitation. It shows NeuroAda achieves the highest average score across both moderate (0.2674%) and extreme low-budget (0.0297%) regimes. Compared to LoRA (0.239%), it improves the average GLUE score by +0.7. Under the extreme budget, it surpasses LoReFT by +0.8, RED by +0.7, and DiReFT by +1.8, despite using fewer parameters. Notably, NeuroAda achieves the best score on 6 out of 8 tasks in the low-budget setting, demonstrating its strong generalization even with minimal adaptation capacity.

5.6 Conclusion

This paper introduced NeuroAda, a featherlight and scalable fine-tuning framework that activates each neuron’s potential through top- k magnitude-based weight selection.

By inheriting the performance benefits of sparse tuning and the memory efficiency of addition-based methods, NeuroAda avoids structural modifications, runtime masking, and full-gradient computation. Its static selection of high-magnitude weights per neuron enables task-agnostic, fine-grained adaptation with significantly reduced memory and computational overhead. Empirical results across diverse reasoning and language understanding tasks show that NeuroAda surpasses strong adaptation baselines, achieving robust generalization under an extremely few number of trainable parameters and tight memory budgets.

5.7 Limitations

While NeuroAda demonstrates strong empirical performance across diverse tasks and architectures, our current evaluation is limited to models up to 13 billion parameters (LLaMA-13B). We anticipate that the benefits of our method may further amplify at larger scales, but assessing its efficacy on models beyond 13B remains an important direction for future work. Evaluating scalability and stability under extreme model sizes is critical for deployment in real-world, high-capacity systems.

Chapter 6

Sparse Training for Gradient Conflict Mitigation in MTL

Chapter Highlights In addition to single-task adaptation, we explore joint multi-task learning (MTL) from the perspective of sparse training, which presents a greater challenge compared to single-task learning. One of the common issues in multi-task learning is the occurrence of gradient conflicts, which can lead to competition among different tasks during joint training. This competition often results in improvements in one task at the expense of deterioration in another. In this chapter, we systematically investigate the occurrence of gradient conflicts across various methods and propose a strategy to reduce such conflicts through sparse training. Our extensive experiments demonstrate that sparse training effectively mitigates gradient conflicts and results in superior performance. Furthermore, sparse training can be seamlessly integrated with gradient manipulation techniques, thereby enhancing their effectiveness.

6.1 Introduction

Attaining the status of a generalist agent necessitates addressing multiple tasks within a unified architecture, thereby emphasizing the significance of multi-task learning (MTL) (Zhang and Yang, 2021), which involves concurrently acquiring proficiency in multiple tasks and striving for superior overall performance compared to learning these tasks separately.

The primary concern for MTL lies in the phenomenon of task competition when the model is jointly trained by optimizing the average loss across all tasks. As a result, a subset of tasks demonstrates superior performance while others remain sub-optimized compared to their individual learning counterparts. One of the reasons behind it, from an optimization perspective, is gradient conflict (GC) (Yu et al., 2020), wherein the direction and magnitude of gradients between tasks differ significantly. This can result in the average gradient biasing towards optimizing one task while providing relatively smaller and sometimes even negative optimization for other tasks when updating the

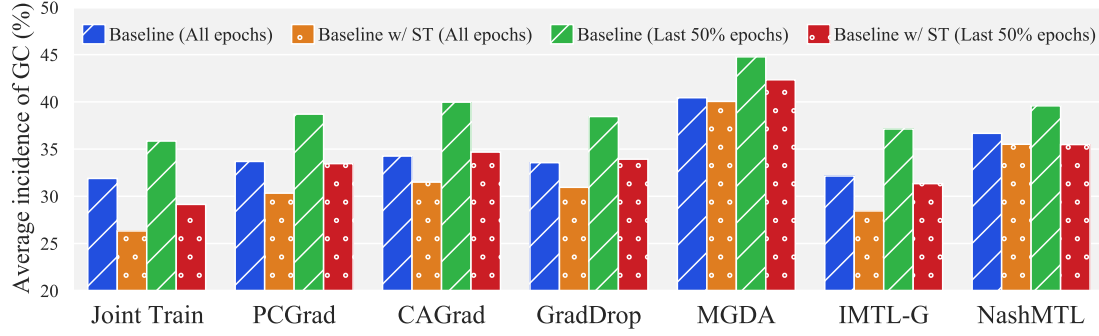


Figure 6.1: The average occurrence percentage of gradient conflict over epochs (all epochs/last 50% epochs) during training on the SAM model with NYUv2 datasets is evaluated using various methods, including joint training and gradient manipulation techniques.

network (Liu et al., 2023a; Yu et al., 2020).

Numerous works have employed the gradient manipulation method to directly or indirectly adjust the gradients of tasks to mitigate the issue of gradient conflict in tasks. The former involves direct alteration of task gradients through manually designed criteria when conflicts arise (Chen et al., 2020e; Liu et al., 2021a; Yu et al., 2020), while the latter modifies task gradients by adjusting weights of loss for each task (Liu et al., 2023a, 2021c; Navon et al., 2022; Sener and Koltun, 2018). Although these methods effectively modify the gradients conflicting with each other, they do not decrease the occurrence of conflicting gradients during training (Shi et al., 2023).

A simple approach to mitigate the occurrence of conflicting gradients is to convert those layers in which gradient conflict frequently arises into task-specific layers, thereby reducing the likelihood of gradient conflicts within the remaining shared layers (Shi et al., 2023). However, this strategy introduces additional modules and disrupts the internal structure of the original model, resulting in increased computational costs. Furthermore, identifying frequently conflicting layers adds extra computational costs. This becomes prohibitively expensive as the model size continues to expand, and thus prompting our fundamental inquiry:

(Q) Is there a universally applicable approach to proactively mitigate the occurrence of gradient conflicts as well as preserve architectural integrity for MTL?

To tackle this issue, we propose a novel perspective on mitigating gradient conflict in MTL, termed Sparse Training (ST), wherein a subset of parameters from the original model are selected to learn multiple tasks simultaneously while keeping the remaining parameters frozen. The intuition behind this lies in the reduction of a high-dimensional optimization problem to a low-dimensional one, which effectively alleviates the optimization complexity. Moreover, restricting the gradient updates of individual tasks to influence only a subset of parameters, rather than all parameters, effectively reduces potential interference between tasks.

Our key findings demonstrate that ST can effectively reduce the incidence of gradi-

ent conflict, particularly during the later stages of training, as illustrated in Figure 6.1. A summary of our contributions is as follows: i) We provide a novel perspective, sparse training, for proactively reducing the incidence of gradient conflict during training while keeping the architecture intact; ii) Sparse training can be easily applied to improve various gradient manipulation methods by reducing the occurrence the gradient conflict over different datasets and architectures; iii) In addition to conventional research that primarily focuses on smaller models (MTAN (Liu et al., 2019b) and SegNet (Badrinarayanan et al., 2017)), we provide a comprehensive assessment of larger pre-trained models, including SAM (Chen et al., 2023), ViT (Dosovitskiy et al., 2020b), Swin Transformer (Liu et al., 2021d), using various gradient manipulation techniques, such as PCGrad (Yu et al., 2020), CAGrad (Liu et al., 2021a), GradDrop (Chen et al., 2020e), MGDA (Sener and Koltun, 2018), IMTL-G (Liu et al., 2021c) and NashMTL (Navon et al., 2022), to stimulate research in the field of sparse training for MTL. Our findings demonstrate that as the model size increases, the issue of gradient conflict becomes more exacerbated, as shown in Figure 6.5a, underscoring the significance of investigating the gradient conflict in large-scale models.

6.2 Related work

Multi-task optimization for MTL The recent works (Chen et al., 2020e; Liu et al., 2023a, 2021a,c; Navon et al., 2022; Sener and Koltun, 2018; Yu et al., 2020) have achieved impressive results in addressing task imbalance issues in MTL by directly or indirectly modifying conflicting task gradients. Specifically, some works (Chen et al., 2020e; Liu et al., 2021a; Yu et al., 2020) propose to form a new update gradient at each training step by directly altering gradients based on certain criteria. Other works (Kendall et al., 2018; Liu et al., 2023a, 2021c; Navon et al., 2022; Sener and Koltun, 2018) learn dynamic loss scale to balance different tasks during training, and thus indirectly altering the gradient of tasks. However, these methods only address GC when it occurs and do not proactively prevent it. In this paper, we sparsely train an MTL model, effectively reducing the incidence of GC.

Training with subset of parameters Several methods have already been proposed in single-task learning. Some of them select a subset of parameters based on a certain pre-defined rule, such as gradient (Fu et al., 2023; Zhang et al., 2023d) and magnitude of parameters (Lagunas et al., 2021). In addition to selecting parameters by hand design, the works in (Mostafa and Wang, 2019; Sanh et al., 2020; Xu et al., 2021) automatically select the subset of parameters through optimization. Although sparse training has been extensively investigated in single-task learning, its application in MTL remains relatively unexplored. Sun et al. (2020) and Calandriello et al. (2014) learn to share information between tasks using a sparse model instead of sparse training. Differently, we research the gradient conflict via the sparse training perspective.

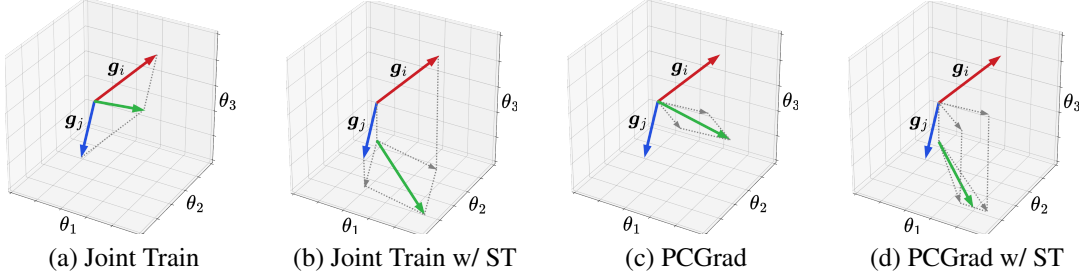


Figure 6.2: Visualization of gradients change for different methods. g_i and g_j are two conflicting gradients, and the green arrow is the actual update vector. The process of sparse training can be interpreted as performing an orthographic/coordinate projection of conflicting gradients onto the subspace defined by the selected parameters, resulting in better alignment of the projected gradients.

6.3 Approach

6.3.1 Background

Multi-task learning (MTL) aims to learn multiple tasks simultaneously within a single model. Formally, given $\{\mathcal{T}_t\}_{t=1}^T$ tasks (≥ 2) and a model Θ with parameters $\Theta = (\theta_{\text{sha}}, \theta_{\text{sep}})$, where θ_{sha} and θ_{sep} are shared parameter with all tasks and task-specific parameters $\theta_{\text{sep}} = \{\theta_{\text{sep}}^t\}_{t=1}^T$ respectively, the commonly used optimization method for MTL (referred to as *Joint Train*) is based on computing the average loss across all tasks with equal weights:

$$\Theta^* = \arg \min_{\Theta} \mathcal{L}(\Theta), \quad (6.1)$$

$$\mathcal{L}(\Theta) = \mathcal{L}(\theta_{\text{sha}}, \theta_{\text{sep}}) = \frac{1}{T} \sum_{t=1}^T \mathcal{L}_t(\theta_{\text{sha}}, \theta_{\text{sep}}^t) \quad (6.2)$$

where each task t is associated with a corresponding loss function $\mathcal{L}_t(\theta_{\text{sha}}, \theta_{\text{sep}}^t)$.

Gradient conflict (GC) However, optimizing all tasks by aggregating their losses indiscriminately (Equation (6.2)) may lead to task competition, wherein certain tasks demonstrate improvement while others exhibit a decline compared to training them separately. From an optimization perspective, one of the reasons stems from conflicts in gradients. Formally, the update of task \mathcal{T}_i may potentially exert a detrimental impact on another task \mathcal{T}_j , namely:

$$\Delta \mathcal{L}_j = \mathcal{L}_j(\hat{\theta}_{\text{sha}}, \theta_{\text{sep}}^j) - \mathcal{L}_j(\theta_{\text{sha}}, \theta_{\text{sep}}^j), \quad (6.3)$$

$$\hat{\theta}_{\text{sha}} = \theta_{\text{sha}} - \alpha \mathbf{g}_i \quad (6.4)$$

where $\mathbf{g}_i = \nabla_{\theta_{\text{sha}}} \mathcal{L}_i(\theta_{\text{sha}}, \theta_{\text{sep}}^i)$ is the gradient of loss on task \mathcal{T}_i with respect to θ_{sha} and α is the learning rate. After the first-order Taylor approximation, Equation (6.3) can be expressed as $-\alpha \mathbf{g}_i \cdot \mathbf{g}_j + o(\alpha)$. Gradient conflict arises when $\mathbf{g}_i \cdot \mathbf{g}_j < 0$, leading to $\Delta \mathcal{L}_j > 0$, indicating that task \mathcal{T}_i has a detrimental impact on task \mathcal{T}_j . Following (Yu et al., 2020), we provide the definition of gradient conflict:

6.3.1. DEFINITION (Gradient Conflict). If $\cos \phi_{ij} < 0$, where ϕ_{ij} is the angle between gradients of two tasks \mathbf{g}_i and \mathbf{g}_j ($i \neq j$), then \mathbf{g}_i and \mathbf{g}_j are deemed to exhibit gradient conflict.

Gradient manipulation To alleviate the issue of gradient conflict, gradient manipulation methods adjust conflicting gradients based on specific criteria and utilize these modified gradients for model updating. Instead of updating the model on the average gradient in Equation (6.1) and Equation (6.2):

$$\nabla_{\theta_{\text{sha}}} \mathcal{L}(\Theta) = \frac{1}{T} \sum_{t=1}^T \nabla_{\theta_{\text{sha}}} \mathcal{L}_t(\theta_{\text{sha}}, \theta_{\text{sep}}^t), \quad (6.5)$$

the gradients of all tasks in gradient manipulation methods are modified as follows:

$$\nabla_{\theta_{\text{sha}}} \mathcal{L}_{\text{gm}}(\Theta) = \frac{1}{T} \sum_{t=1}^T \mathbf{w}_t \nabla_{\theta_{\text{sha}}} \mathcal{L}_t(\theta_{\text{sha}}, \theta_{\text{sep}}^t), \quad (6.6)$$

$$\mathbf{w}_t = f(\nabla_{\theta_{\text{sha}}} \mathcal{L}_1, \dots, \nabla_{\theta_{\text{sha}}} \mathcal{L}_T) \quad (6.7)$$

where \mathbf{w}_t can be either pre-defined or dynamically computed for tasks via f and thus achieve the aim of adjusting the task gradient (Chen et al., 2020e; Liu et al., 2023a, 2021a,c; Navon et al., 2022; Sener and Koltun, 2018; Yu et al., 2020). However, the results of our experiment suggest that these methods can only modify gradients when conflicts occur, rather than proactively reducing the occurrence of GC during training, compared with *Joint Train*, as shown in Figure 6.1.

6.3.2 Sparse training for multi-task learning

In this study, we investigate the gradient conflict commonly observed in multi-task learning from a novel perspective: sparse training, which selectively trains only a subset of the model parameters as opposed to full parameter training. This perspective is based on the intuition that by converting a high-dimensional space optimization problem into a lower-dimensional one, the complexity of optimization can be effectively reduced. Additionally, by limiting the impact of gradient updates to only a subset of parameters for each task instead of all parameters, potential interference between tasks can be mitigated.

Sparse training (ST) entails the initial parameter selection from the original model, and then updating only these parameters while keeping other parameters fixed during model training. To clarify potential misunderstandings regarding ST—often confused with sparse networks, where parameters are abandoned for model compression—we provide the following definition to ensure consistency and ease of understanding throughout this paper.

6.3.2. DEFINITION (Sparse Training). Given a model Θ and a binary mask matrix M indicating whether parameters in Θ are selected, where $M \in \mathbb{R}^{|\Theta| \times |\Theta|}$, $M_{ii} \in \{0, 1\}$ and $M_{ij} = 0$ ($\forall i \neq j$), the model is updated by $\hat{\Theta} = \Theta - \alpha M \nabla_{\Theta} \mathcal{L}(\Theta)$. We define this training strategy as sparse training.

Typically, the model architecture in multi-task learning includes a shared encoder as a feature extractor with task-specific decoders for multiple tasks. Therefore, sparse training is used in the encoder, and full parameters training for the decoders. We detail how the mask is computed in section Section 6.3.4. We now apply sparse training for multi-task learning (*Joint Train*). The visualization of the gradient change can be viewed in Figure 6.2 and the update with the reformulated gradient from Equation (6.5) is as follows

$$\hat{\theta}_{\text{sha}} = \theta_{\text{sha}} - \nabla_{\theta_{\text{sha}}} \mathcal{L}(\Theta) = \theta_{\text{sha}} - M \frac{1}{T} \sum_{t=1}^T \nabla_{\theta_{\text{sha}}} \mathcal{L}_t(\theta_{\text{sha}}, \theta_{\text{sep}}^t). \quad (6.8)$$

Combination with gradient manipulation methods The application of sparse training can be seamlessly and effectively extended to improve various gradient manipulation methods in MTL. The update with the reformulated gradient from Equation (6.6) is as follows

$$\hat{\theta}_{\text{sha}} = \theta_{\text{sha}} - \nabla_{\theta_{\text{sha}}} \mathcal{L}_{\text{gm}}(\Theta) = \theta_{\text{sha}} - M \frac{1}{T} \sum_{t=1}^T \mathbf{w}_t \nabla_{\theta_{\text{sha}}} \mathcal{L}_t(\theta_{\text{sha}}, \theta_{\text{sep}}^t). \quad (6.9)$$

6.3.3 Theoretical analysis for sparse training

After introducing sparse training into MTL, the optimization objective in Equation (6.1) can be formed:

$$\Theta^* = \arg \min_{\Theta} \mathcal{L}(\Theta), \text{ s.t. } \|(I - M)(\theta_{\text{sha}} - \theta_{\text{sha}}^{\text{in}})\|^2 = 0, \quad (6.10)$$

where $\theta_{\text{sha}}^{\text{in}}$ is the initialized original model for θ and I is identity matrix. According to Lagrangian duality, Equation (6.10) can be reformulated as:

$$L = \min_{\Theta} \max_{\lambda} \mathcal{L}(\Theta) + \lambda \|(I - M)(\theta_{\text{sha}} - \theta_{\text{sha}}^{\text{in}})\|^2. \quad (6.11)$$

This can be transformed to optimize the upper bound L of regularized problem:

$$L_r = \min_{\Theta} \mathcal{L}(\Theta) + \|(I - M)(\theta_{\text{sha}} - \theta_{\text{sha}}^{\text{in}})\|^2 \leq L. \quad (6.12)$$

Please see the supplemental material for proof. [Fu et al. \(2023\)](#) demonstrates that Equation (6.12) has better stability and smaller generalization bound than only optimizing Equation (6.1), resulting in better performance.

6.3.4 Parameter selection per neuron (PSN)

Several promising sparse training methods exist for single-task learning, but they are either time-consuming, requiring mask updates at each iteration ([Mostafa and Wang, 2019](#); [Sanh et al., 2020](#); [Xu et al., 2021](#)), or memory-intensive due to gradient calculations for all parameters ([Fu et al., 2023](#); [Zhang et al., 2023d](#)). In MTL, where multiple tasks are trained simultaneously, time efficiency is crucial. Thus, we adopt a one-time selection method, choosing parameters before training and keeping the mask fixed throughout. We consider the following two aspects for selection, magnitude of the parameter and involvement of all neurons in the network.

The magnitude of parameters Several studies have focused on model compression through the elimination of parameters with lower magnitudes ([Frankle and Carbin, 2018](#); [Han et al., 2015b](#)). This highlights the significance of parameters with larger magnitudes in neural networks, which is consistent with our experimental findings (See Figure 6.5c). The intuition behind this phenomenon lies in the fact that parameters with larger magnitudes exert a greater influence on altering neuron activation states through the activation function, wherein a neuron becomes active once the input surpasses a predefined threshold. Therefore, we exclusively select parameters with the highest magnitude for training multiple tasks.

The involvement of all neurons A simple idea is to select a certain proportion of parameters with the highest magnitude from the neural network (NN), but this may prevent some neurons from being engaged during training and hinder effective model training due to the dependence of the NN state on neuron activation. Motivated by studies highlighting distinct roles for different components in NN ([Fan et al., 2020](#); [Wang et al., 2021a](#); [Zhang et al., 2023d](#)), we posit that engaging all neurons is crucial for effective model training. The rationale is that each neuron within the network possesses the inherent capability to finely adjust its activation state, thereby effectively adapting the overall NN state to the tasks, especially for learning multiple tasks simultaneously. Our experiments further substantiate this assertion, as shown in Figure 6.5c.

PSN By integrating the two aspects, we select the top-K connections (weight/parameters) with the highest magnitude among all input connections for each neuron in the network (Please see Figure 6.3 for top-1 example). This approach facilitates the training process for fitting tasks by ensuring that every neuron possesses activation potential, while parameters with higher magnitudes facilitate easier activation

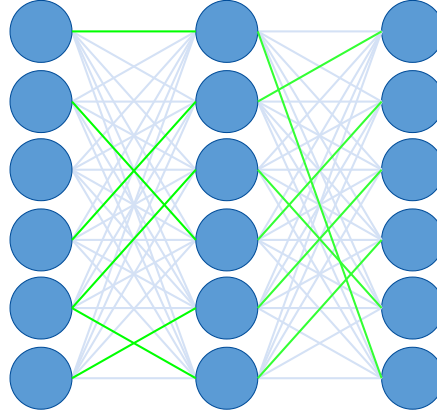


Figure 6.3: PSN. Top-1 highest-magnitude parameter among all input connections of each neuron is selected.

of neurons. In this paper, sparse training refers to using this method to select parameters and training the selected parameter, unless otherwise specified.

6.4 Experiments

Our experiments are conducted on comprehensive MTL benchmarks to evaluate the effectiveness of sparse training. First, we investigate if sparse training reduces gradient conflict. Subsequently, we examine its impact on performance across various MTL setups. The more details of the experiment are provided in Appendix D.4.

6.4.1 EXPERIMENTAL SETUP

Dateset Our MTL datasets are categorized into three groups: i) *Dense prediction tasks*: **NYUv2** (Couprie et al., 2013): An indoor scene understanding dataset containing 1449 RGBD images with per-pixel labels across 13 classes, including semantic segmentation, depth estimation, and surface normal prediction. **CityScapes** (Cordts et al., 2016): 5000 street-view RGBD images with per-pixel annotations for 7-class semantic segmentation and depth estimation. ii) *Multiple binary-classification tasks*: **CelebA** (Liu et al., 2015): 200,000 facial images of 10,000 celebrities, each with 40 binary attributes for facial features. We use the first 10 attributes for 10 binary classification tasks due to limited computation. iii) *Multiple multi-class classification tasks*: **VTAB** (Zhai et al., 2019): Containing 24 image understanding tasks with 1000 training examples per task. We use four tasks from it to create two multi-task benchmarks: **Clevr**: Simple 3D shapes with counting and depth prediction tasks. **SmallNORB**: Artificial objects with object azimuth and camera elevation prediction tasks.

Baseline We evaluate our approach using various baselines including i) single-task learning (*STL*): Each task is trained independently; ii) *Joint Train*: Training all tasks with average task loss; and 6 gradient manipulation methods including 3 direct and 3 indirect modification techniques. The former includes: iii) *PCGrad*: Projecting each task gradient onto the normal plane of other tasks (Yu et al., 2020); iv) *CAGrad*: Enhancing the optimization of average loss by explicitly regulating the minimum decrease across tasks (Liu et al., 2021a); and v) *GradDrop*: Stochastically dropping specific dimensions of the gradients based on their level of conflict. The latter includes vi) *MGDA*: Identifying the same descent direction for each task (Sener and Koltun, 2018); vii) *IMTL-G*: Determining the update direction by ensuring equal projections on gradients (Liu et al., 2021c); viii) *NashMTL*: Treating MTL as a bargaining game to optimize all tasks (Navon et al., 2022).

Model We experiment with several architectures including: i) *CNN-based*: *MTAN* (Liu et al., 2019b) incorporates an attention mechanism into the SegNet (Badrinarayanan et al., 2017). ii) *Transformer-based*. *SAM* (Kirillov et al., 2023b) is a strong visual foundation model for segmentation. *ViT-B/16* (Dosovitskiy et al., 2020b) and *Swin Transformer* (Liu et al., 2021d) are vision classification models pre-trained on ImageNet21K (Deng et al., 2009). All experiments were conducted on pre-trained SAM, ViT and Swin (except for randomly initialized MTAN), unless otherwise specified.

Evaluation i) *Relative task drop* ($\Delta m\%$). Following (Maninis et al., 2019), we evaluate the MTL overall performance for a baseline b by computing the average performance drop against STL s over $\{\mathcal{T}_t\}_{t=1}^T$ tasks and $K_{\mathcal{T}_t}$ metrics for each \mathcal{T}_t : $\Delta m\% = (\frac{1}{T} \sum_{t=1}^T \frac{1}{K_{\mathcal{T}_t}} \sum_{k=1}^{K_{\mathcal{T}_t}} (-1)^{\delta_k} (M_b^k - M_s^k) / M_s^k) \times 100$ where M_b^k, M_s^k are the value of metrics k evaluated with b and s respectively. $\delta_k = 1$ if the M^k is higher the better and 0 otherwise. ii) *Average incidence of GC* ($p\%$). We evaluate the extent of gradient conflict for a baseline by calculating the average incidence of GC over epochs during training. Given T tasks, E epochs, and I iterations per epoch, $p\% = \frac{1}{EI} \sum_{e=1}^E \sum_{i=1}^I (N_{gc} / N_{all}) \times 100$, where N_{gc} and N_{all} represent the number of occurrence of gradient conflicts between two tasks for all task combinations $\binom{T}{2}$ and the number of the combinations in each iteration during training, respectively.

6.4.2 Incidence of gradient conflict

We train a MTL model using the *Joint Train* and 6 state-of-the-art gradient manipulation techniques including *PCGrad*, *CAGrad*, *GradDrop*, *MGDA*, *IMTL-G* and *NashMTL* and then introduce our sparse training strategy to these methods. Throughout the training process, we record instances of GC between any two tasks among all tasks for each training iteration and then calculate the average incidence of GC both over all epochs and the last 50% epochs. The observations of the SAM model on the NYU-v2

dataset are provided below. Similar results on other datasets and models are shown in Appendix D.6, Appendix D.6, Appendix D.6 and Appendix D.6.

Gradient manipulation methods cannot effectively reduce the incidence of gradient conflict The gradient manipulation methods (Chen et al., 2020e; Liu et al., 2023a, 2021a,c; Navon et al., 2022; Sener and Koltun, 2018; Yu et al., 2020) aim to modify conflicting gradients that are prevalent during the joint training of MTL. As shown in Table 6.1, the average incidence of GC using *Joint train* is 31.89% across all training epochs and 35.85% over the last 50% epochs. The incidence of GC cannot be effectively reduced by any gradient magnitude methods compared with the *Joint train*, as shown in Figure 6.1 and Table 6.1. The reason is that these methods can only make the conflicting gradients not conflict when the GC occurs, rather than proactively prevent the occurrence of GC. The incidence of GC is even exacerbated by these methods, particularly MGDA showing a significant increase of 8.55% compared to *Joint Train*. Notably, these findings are consistent with (Shi et al., 2023), where they provide the distribution of the angles between the two task gradients.

Sparse training effectively decreases the occurrence of gradient conflict As shown in Table 6.1, after combining sparse training with all methods, including *Joint Train* and gradient manipulation methods, the average incidence of gradient conflict is effectively reduced over all epochs. For example, ST in *Joint Train* reduced the incidence over all epochs by 5.56%. The phenomenon of gradient conflict reduction is consistently observed in nearly every training epoch, as illustrated in Figure 6.4, which further demonstrates the effectiveness of ST for decreasing gradient conflict. In addition, all methods with ST exhibit a greater improvement in the average incidence of gradient conflict during the last 50% epochs compared to all epochs, which implies a greater level of prevention of gradient conflict with the progress of sparse training. For instance of NashMTL, there is a threefold improvement in the average incidence of gradient conflict during the last 50% epochs compared to all epochs.

6.4.3 Performance on diverse benchmarks

It is natural to investigate whether reducing gradient conflict during training through sparsity can enhance performance on common benchmarks. In this section, we present diverse benchmarks to demonstrate the effectiveness of ST.

Sparse training improves the performance for all state-of-the-art methods The performance of *Joint Train* and all gradient manipulation methods is consistently improved by sparse training, as demonstrated in Table 6.2 for NYU-v2 benchmarks. Specifically, sparse training not only enhances overall task performance but also improves individual task performance for the majority of methods. For example, in Table 6.2,

Methods	Average incidence of GC (%)	
	All epochs	Last 50% epochs
Joint Train	31.89	35.85
w/ ST	26.33 (5.56)	29.14 (6.71)
PCGrad	33.69	38.70
w/ ST	30.33 (3.36)	33.46 (5.24)
CAGrad	34.26	39.97
w/ ST	31.50 (2.76)	34.68 (5.29)
GradDrop	33.56	38.45
w/ ST	30.95 (2.61)	33.93 (4.52)
MGDA	40.44	44.77
w/ ST	40.05 (0.39)	42.34 (2.43)
IMTL-G	32.15	37.13
w/ ST	28.45 (3.70)	31.34 (5.79)
NashMTL	36.67	39.58
w/ ST	35.51 (1.16)	35.48 (4.10)

Table 6.1: Average incidence of GC between tasks for different methods. We compute the average incidence of GC over all epochs and the last 50% epochs during training SAM on NYUv2. The improvement by sparse training is provided in (•).

Joint Train demonstrates improvements across all individual tasks through sparse training. Similarly, as shown in Table 6.3, all methods exhibit notable improvements by sparse training on CelebA, Clevr, SmallNORB and CityScapes benchmarks.

Effectiveness on both pre-trained and randomly initialized models Our study primarily focuses on the sparse training for large pre-trained models, because leveraging prior knowledge from these models can be beneficial for MTL and our experimental results demonstrate that larger models exhibit a more severe gradient conflict, as shown in Figure 6.5a. However, in order to ensure a fair comparison with related works that manipulate gradients in small and randomly initialized models, we also conduct experiments under the same setting as theirs to further demonstrate the effectiveness of sparse training. As shown in Table 6.3, we observe that even for the small randomly initialized models, the performance of joint training and all gradient manipulation methods is improved by sparse training. Please see Table 20 and Table 25 for the detailed results in the Appendix.

Generalization on different architectures and MTL tasks To evaluate the generalization across diverse architectures and MTL tasks, we conducted experiments

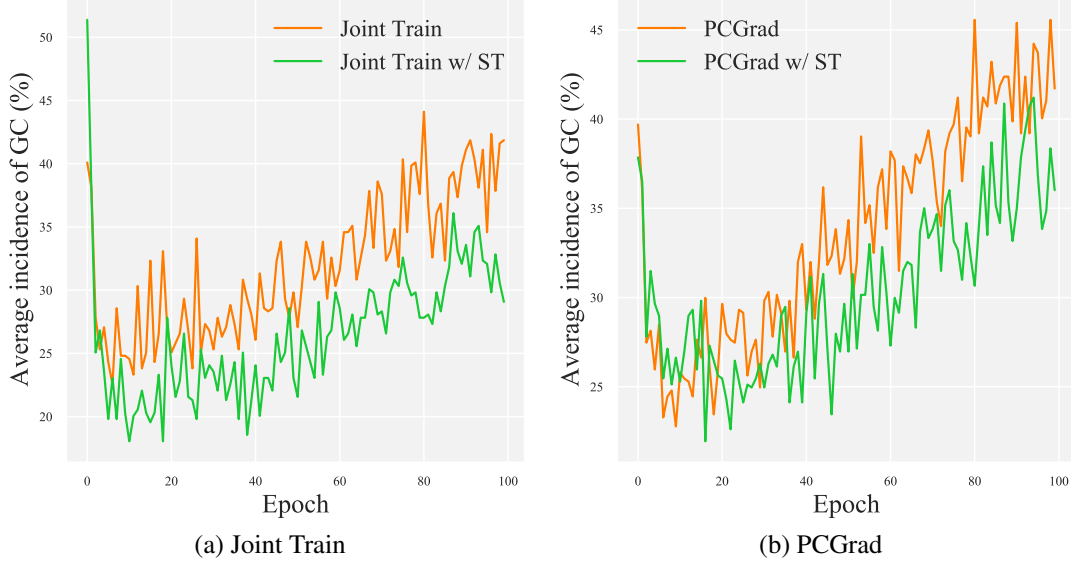


Figure 6.4: The incidence of GC between tasks during training SAM on NYUv2 dataset. The top and bottom figures are *Joint Train* and *PCGrad* respectively. Please see Figure 14 in Appendix D.6 for more results on other gradient manipulation methods.

on both CNN-based models and transformer-based models with varying visual MTL capabilities. Specifically, our MTL tasks encompassed visual classification (CelebA, Clevr and SmallNORB) and visual dense prediction (NYU-v2 and CityScapes). For the former, we utilized Swin Transformer and ViT as backbones for multiple binary classification tasks (Table 6.3) and two multi-class classification tasks (Table 6.3, and Table 24 in Appendix), respectively. The latter involved predicting dense masks for each task, necessitating an encoder-decoder structure to generate corresponding masks. We explored two types of structures: a symmetrical encoder-decoder structure with a CNN-based model, e.g. MTAN (Table 6.3, and Tables 20 and 25 in Appendix) and an asymmetric structure with a heavy-weight encoder and a light-weight decoder using a transformer-based model, e.g. SAM (Table 6.2 in Appendix). As shown in these tables, the efficacy of sparse training in improving all baselines across various architectures and MTL tasks underscores its robust generalization capability.

6.4.4 Ablation study

The larger the model, the more severe gradient conflicts. In this paper, we focus more on investigating the gradient conflict in the pre-trained large models as larger models demonstrated a more severe phenomenon of gradient conflict. This can be observed in Figure 6.5a, where *Swin/Tiny* demonstrates significantly less gradient conflict compared to *Swin/Base* and *Swin/Large*. It is worth noting that although larger models tend to experience more severe gradient conflicts, this does not necessarily lead

Methods	Segmentation		Depth		Surface Normal					$\Delta m\%$ ↓
	mIoU ↑	Pix Acc ↑	Abs Err ↓	Rel Err ↓	Angle Distance ↓		Within t° ↑			
					Mean	Median	11.25	22.5	30	
STL	58.62	79.20	0.3810	0.1553	19.29	12.64	46.37	72.19	80.73	—
Joint Train w/ ST	59.09	79.61	0.3348	0.1360	22.34	16.33	35.46	64.02	75.20	6.763
	60.03	79.96	0.3320	0.1353	21.98	15.92	36.69	64.92	75.82	5.314
PCGrad w/ ST	59.18	80.12	0.3258	0.1323	21.81	15.72	36.92	65.49	76.26	4.584
	59.37	80.33	0.3272	0.1330	21.53	15.39	38.02	66.09	76.71	3.741
CAGrad w/ ST	59.78	80.16	0.3215	0.1305	19.92	13.40	43.87	70.47	79.59	−1.816
	60.33	80.20	0.3232	0.1306	19.74	13.20	44.42	71.04	80.02	−2.423
GradDrop w/ ST	59.02	79.80	0.3283	0.1321	22.03	15.95	36.42	64.90	75.79	5.323
	59.74	80.32	0.3278	0.1322	21.81	15.63	37.40	65.50	76.15	4.329
MGDA w/ ST	37.43	67.58	0.4427	0.1810	19.23	12.61	46.43	72.35	80.87	9.162
	41.60	69.96	0.4414	0.1778	19.22	12.61	46.44	72.29	80.80	7.791
IMTL-G w/ ST	60.64	80.29	0.3324	0.1348	19.85	13.37	43.92	70.78	79.9	−1.537
	60.35	80.18	0.3347	0.1350	19.65	13.15	44.66	71.25	80.20	−1.955
NashMTL w/ ST	59.42	80.20	0.3303	0.1341	19.90	13.39	43.86	70.65	79.72	−1.295
	59.36	79.98	0.3278	0.1323	19.63	13.02	45.06	71.31	80.15	−2.384

Table 6.2: The test performance on NYU-v2 dataset training on SAM model. The green cell color indicates that sparse training improves the performance of joint training or gradient manipulation methods. The best result is highlighted in bold.

to inferior performance compared to smaller models with milder gradient conflicts. This discrepancy can be attributed to differences in model capacity and the prior knowledge embedded through pre-training. Nevertheless, this observation underscores the importance of exploring methods to mitigate gradient conflicts in larger models. Within the same model architecture and size, reducing gradient conflicts has been shown to improve performance, as evidenced by works such as (Liu et al., 2021a; Yu et al., 2020). Addressing severe gradient conflicts in larger models may thus unlock their full potential, enabling better utilization of their capacity and capabilities.

Effortless search for the number of trainable parameters. We explore the effect of trainable parameter numbers for ST. The results in Figure 6.5b show that the pre-trained model (SAM) and the randomly initialized model (MTAN) have different optimal trainable parameter numbers. MTAN requires $\sim 60\%$ of the parameters, while SAM needs only $\sim 30\%$, leveraging information from the pre-trained model. In our paper, most of the experiments use these proportions for ST and achieve better results (please see Table 17 in Appendix D.4 for the detailed number). Additionally, ST offers a wide range of trainable parameter options that outperform *Joint Train*, which implies that hyperparameter search for the number of trainable parameters becomes effortless. Specifically, both models have a $\sim 40\%$ probability of yielding superior outcomes.

Methods	CelebA	Clevr		SmallNORB	NYU-v2	CityScapes
	$\Delta m\% \downarrow$	Counting	Depth	$\Delta m\% \downarrow$	$\Delta m\% \downarrow$	$\Delta m\% \downarrow$
	(F1)	(Top 1 \uparrow)	(Top 1 \uparrow)			
STL	—	58.64	57.68	—	—	—
Joint Train w/ ST	3.12	54.86	54.68	5.84	10.70	5.59
	2.03	61.80	54.81	-0.21	10.11	2.49
PCGrad w/ ST	1.70	49.01	53.39	11.93	9.99	3.97
	1.42	59.01	55.29	1.75	9.71	1.98
CAGrad w/ ST	1.96	49.33	53.67	11.41	10.50	0.20
	1.23	58.51	55.27	2.19	10.22	-2.76
GradDrop w/ ST	1.18	49.02	52.88	12.36	11.73	3.58
	0.83	58.87	54.07	2.94	10.76	1.38
MGDA w/ ST	-0.41	49.56	55.97	9.22	10.15	1.38
	-1.08	58.23	56.91	1.02	9.79	-3.18
IMTL-G w/ ST	0.97	54.99	54.51	5.87	10.19	-0.76
	0.19	61.05	56.73	-1.24	10.15	-3.18
NashMTL w/ ST	3.59	47.04	53.07	13.89	10.84	-4.04
	3.22	58.61	54.97	2.37	9.57	-5.11

Table 6.3: The test performance on CelebA, Clevr, SmallNORB, NYU-v2 and CityScapes dataset. CelebA is trained on Swin Transformer. Clevr and SmallNORB are trained on ViT. NYU-v2 and CityScapes are trained on MTAN. We only present $\Delta m\%$ for limited space. Please see Table 20, Table 25 and Table 24 for detailed results in supplemental materials. The green cell color indicates that sparse training improves the performance of joint training or gradient manipulation methods. The best result is highlighted in bold.

Effectiveness for both higher magnitude and neural-level selection. We investigate various parameter selection approaches: *Random*: Randomly selecting parameters from the network; *Global*: Choosing parameters with the highest magnitude from the whole network instead of the input connections of each neuron in the network (*Ours*); *Reverse*: Selecting parameters with the lowest magnitude among input connections of each neuron. For a fair comparison, we maintain the same selected number. The results in Figure 6.5c indicate that higher magnitude values are superior to lower ones (*Ours* > *Reverse*). Furthermore, it is crucial to evenly select parameters from the entire network (*Ours* > *Random* > *Global*), as *Ours* ensure that the parameters of input connection for each neuron are selected, and *Random* guarantees an equal proportion of parameters is selected in each block of the network, whereas this is not the case for *Global* (see Figure 13 for detailed statistics in Appendix).

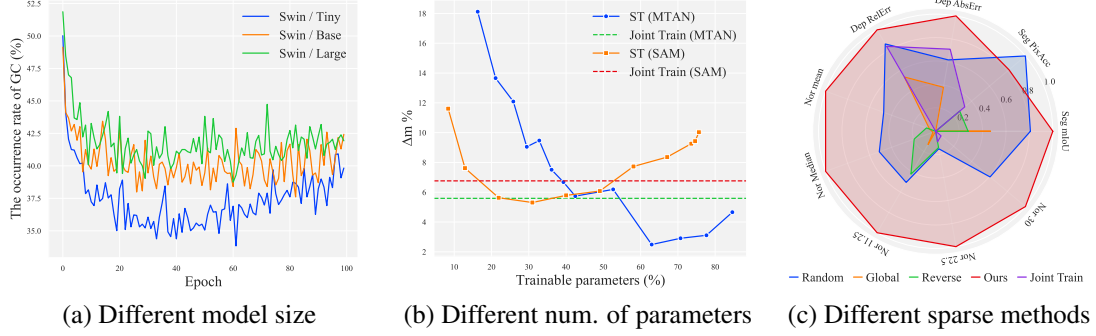


Figure 6.5: Ablation study for *Joint Train* with NYU-v2 dataset. (a) The average incidence of GC during joint training on different sizes of Swin transformers. Please see the numerical statics for all epochs in Table 21 in Appendix D.6. (b) The different number of trainable parameters for MTAN and SAM models. (c) Different sparse methods training on SAM. Metrics for all tasks are min-max normalized. Please see Table 18 for detailed results in Appendix D.6.

6.5 Conclusion

In this paper, the occurrence of gradient conflict in multi-task learning is extensively investigated from a novel perspective: sparse training. Extensive experiments demonstrate that sparse training transferring high-dimensional space into low-dimensional space effectively reduces the incidence of gradient conflict during training while preserving the integrity of the original model. Furthermore, combining sparse training with other gradient manipulation methods significantly improves performance for multi-task learning.

Chapter 7

Cross-modal Information Flow in Multimodal LLMs

Chapter Highlights The recent advancements in auto-regressive multimodal large language models (MLLMs) have demonstrated promising progress for vision-language tasks. However, little is currently known about the inner working mechanism of MLLMs and how linguistic and visual information interact within these models. In this chapter, we aim to fill this gap by examining the information flow between different modalities—language and vision—in MLLMs. Specifically, given an image-question pair as input, we investigate where in the model and how the visual and linguistic information are combined to generate the final prediction. Through experiments conducted with a series of models from the LLaVA series, we gain a new and comprehensive understanding of the spatial and functional aspects of image and language processing within MLLMs. This perspective not only facilitates future research into multimodal information localization and editing but also provides valuable insights for the development of more general and robust multimodal models in the future.

7.1 Introduction

Multimodal large language models (MLLMs) (Bai et al., 2023a; Dai et al., 2023; Li et al., 2023a; Liu et al., 2024a,b) have demonstrated notable performance across a wide range of vision-language tasks, which is largely attributed to the combination of powerful auto-regressive large language models (Touvron et al., 2023a; Zhang et al., 2022b; Zheng et al., 2023) and visual encoders (Dosovitskiy et al., 2020a; Fang et al., 2023; Radford et al., 2021b). Specifically, LLMs generate responses based on both visual and linguistic inputs where visual representations extracted from an image encoder precede the word embeddings in the input sequence. Despite the successful performance and wide applicability of MLLMs, there is still a lack of understanding of their internal working mechanisms at play when solving multimodal tasks. Acquiring deeper insights into these mechanisms could not only enhance the interpretability and transparency

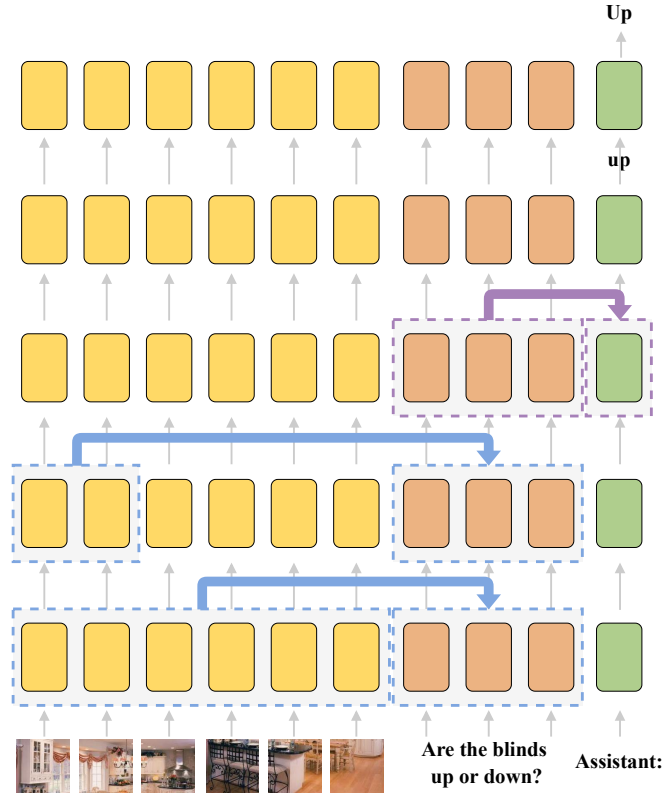


Figure 7.1: Illustration of the internal mechanism of MLLMs when solving multimodal tasks. From bottom to top layers, the model first propagates general visual information from the whole image into the linguistic hidden representation; next, selected visual information relevant to answering the question is transferred to the linguistic representation; finally, the integrated multimodal information within the hidden representation of the question flows to *last position* facilitating the final prediction. In addition, the answers are initially generated in lowercase form and then converted to uppercase for the first letter.

(Nanda et al., 2023; Olah, 2024) of these models but also pave the way for developing more efficient and robust models for multimodal interactions.

Some initial studies have begun to explore the internal states corresponding to external behaviors of MLLMs, focusing on specific aspects such as information storage in the model’s parameters (Basu et al., 2024), reflecting undesirable content generation through logit distributions of the generated tokens (Zhao et al., 2024), the localization and evolution of object-related visual information (Neo et al., 2024; Palit et al.; Schwettmann et al., 2023), the localization of safety mechanism (Xu et al., 2024) and the reduction of redundant visual tokens (Zhang et al., 2024a). However, the information flow between the two modalities within MLLMs remains poorly-understood, thus prompting our main question: *Where in the model and how is visual and linguistic in-*

formation integrated within the auto-regressive MLLMs to generate the final prediction in vision-language tasks?

To address this question, we investigate the interaction of different modalities by locating and analyzing the information flow (Elhage et al., 2021) between them, across different layers. Our focus is on the task of visual question answering (VQA), a popular multimodal task, where the *answer* is generated by MLLMs based on the input *image* and the corresponding *question*. Specifically, we aim to reverse engineer the information flow between the two modalities at inference time, by selectively inhibiting specific attention patterns between tokens corresponding to visual and linguistic inputs and by observing the resulting changes in the performance of the *answer* prediction.

In modern auto-regressive MLLMs, which employ Transformer decoder-only architecture (Vaswani et al., 2017b), the attention layer is the sole module enabling communication between hidden representations corresponding to different positions of the input. To inhibit cross-modal information flow, we therefore adopt an *attention knockout* approach, proposed by Geva et al. (2023). We use it to block attention edges connecting different types of hidden representations (*e.g.* *image* and *question*) at specific transformer layers.

We apply this method to a range of MLLMs from the LLaVA series, including LLaVA-1.5-7b, LLaVA-1.5-13b (Liu et al., 2024a), LLaVA-v1.6-Vicuna-7b (Liu et al., 2024b) and Llama3-LLaVA-NEXT-8b (lmm, 2024) and a number of diverse question types in VQA, as shown in Table 7.1. Our experiments focus on the following research questions: (1) How is the (more general) visual information from the whole image fused with the linguistic information in the question? (2) How is the more targeted visual information (*i.e.* specific image regions directly relevant to answering the question) integrated with linguistic information from the question? and (3) In what ways do the linguistic and visual components of the input contribute to the final *answer* prediction? To answer these questions we conduct a series of experiments, blocking information flow between (1) the input positions corresponding to the whole image to the different parts of the question; (2) the input positions corresponding to image regions containing objects relevant to answering the question, to the question; (3) the input positions corresponding to the image and the question to the final prediction, across different layers of the MLLM.

Our results reveal that in MLLMs, visual information undergoes a two-stage integration into the language representation within the lower-to-middle layers: first in a comprehensive manner, and subsequently in a more targeted fashion. This integrated multimodal representation is then propagated to the hidden representations in the subsequent layers, ultimately reaching the last position for generating an accurate response. The visualization of this mechanism is shown in Figure 7.1. To the best of our knowledge, ours is the first paper to elucidate the information flow between the two modalities in auto-regressive MLLMs. It thus contributes to enhancing the transparency of these models and provides novel and valuable insights for their development.

7.2 Related work

MLLMs multimodal large language models have demonstrated remarkable performance across a wide range of vision-language tasks, which is largely attributed to the development of the auto-regressive large language models. The representative MLLMs (Bai et al., 2023a; Dai et al., 2023; Li et al., 2023a, 2024; Liu et al., 2024a,b, 2023b) consist of an image encoder (Dosovitskiy et al., 2020a; Fang et al., 2023; Radford et al., 2021b) and a powerful decoder-only large language model (Touvron et al., 2023a; Zhang et al., 2022b; Zheng et al., 2023). The visual and linguistic information are integrated in original LLM. In this paper, we will investigate this inner working mechanism of multimodal information processing into these models.

Interpretability of multimodal models The interpretability of multimodal models has attracted a great deal of attention in the research community. Works in (Cao et al., 2020; Frank et al., 2021a) treat the model as a black box, analyzing input—output relationships to interpret the behavior of models, such as comparing the importance of different modalities (Cao et al., 2020) and the different modalities’ contribution to visual or textual tasks (Frank et al., 2021a). The works from (Aflalo et al., 2022; Chefer et al., 2021a; Lyu et al., 2022; Stan et al., 2024) aim to explain predictions by tracing outputs to specific input contributions for a single sample, including through merging the attention scores (Aflalo et al., 2022; Stan et al., 2024), using gradient-based methods (Chefer et al., 2021a) or model disentanglement (Lyu et al., 2022). Additionally, some works (Dahlgren Lindström et al., 2020; Hendricks and Nematzadeh, 2021; Salin et al., 2022) adopt a top-down approach, probing learned representations to uncover high-level concepts, such as visual-semantics (Dahlgren Lindström et al., 2020), verb understanding (Hendricks and Nematzadeh, 2021), shape and size (Salin et al., 2022). In contrast, our work focuses on the model’s internal processing mechanisms when solving multimodal tasks.

Mechanistic interpretability of MLLMs Mechanistic interpretability (Nanda et al., 2023; Olah, 2024) is an emerging research area in NLP, aiming to reverse-engineer detailed computations within neural networks. While it has gained attraction in NLP, research in the multimodal domain remains limited. Palit et al. introduced a causal tracing tool for image-conditioned text generation on BLIP (Li et al.), marking one of the few early efforts in this area. Several initial studies have started to explore the internal states of MLLMs by linking external behaviours to specific mechanisms, such as information storage in model parameters (Basu et al., 2024), undesirable content generation reflected in the logit distributions of the first generated token (Zhao et al., 2024), localization and evolution of object-related visual information (Neo et al., 2024; Palit et al.; Schwettmann et al., 2023), safety mechanism localization (Xu et al., 2024), and reducing redundant visual tokens (Zhang et al., 2024a). However, research offering a comprehensive understanding of the internal mechanisms behind multimodal information integration in MLLMs is still lacking. This paper makes an important first step towards filling this gap.

7.3 Tracing information flow in MLLMs

The focus of this paper is on auto-regressive multimodal large language models, which consist of an image encoder and a decoder-only language model, as shown in Figure 7.2. The image encoder transforms images into representations that the language model can take as input, while the language model integrates these visual cues with any provided text, generating responses one word at a time. Often, these components are initialized from a pre-trained image encoder (*e.g.* CLIP-ViT-L-336px (Radford et al., 2021b)) and a large language model (*e.g.* Llama 2 (Touvron et al., 2023a)) respectively. Since the interaction between modalities only occurs in the decoder-only transformer, our analysis centers around it and we refer to it as MLLM for brevity unless otherwise specified.

7.3.1 Background: MLLMs

Input The input to an MLLM typically comprises image and text features, with the image features being initially extracted from an image encoder and the text being encoded through word embeddings. Formally, an image x is evenly split into fixed-size patches and encoded by an image encoder to obtain N_V visual patch features $\mathbf{V} = [\mathbf{v}_i]_{i=1}^{N_V}$, $\mathbf{v}_i \in \mathbb{R}^d$. Similarly, the text t , consisting of N_T tokens, is embedded into representations through a lookup table of word embeddings, resulting in the text input $\mathbf{T} = [\mathbf{t}_i]_{i=1}^{N_T}$, $\mathbf{t}_i \in \mathbb{R}^d$. By concatenation of \mathbf{V} and \mathbf{T} , the multimodal input sequence $\mathbf{I} = [\mathbf{v}_1 \dots \mathbf{v}_{N_V}, \mathbf{t}_1 \dots \mathbf{t}_{N_T}] \in \mathbb{R}^{N \times d}$, where $N = N_V + N_T$, is fed into MLLM.

Hidden representation The input sequence is fed into the MLLM, where the hidden representation at each token position is encoded across L transformer layers. Each layer primarily consists of two modules: a masked multi-head attention (MHAT) followed by a fully connected feed-forward network (FFN) (Vaswani et al., 2017b). For conciseness, we have excluded the bias terms and layer normalization, as they are not crucial for our analysis. Formally, the hidden representation $\mathbf{h}_i^\ell \in \mathbb{R}^d$ in the position i of the input sequence at layer ℓ can be expressed as

$$\mathbf{h}_i^\ell = \mathbf{h}_i^{\ell-1} + \mathbf{a}_i^\ell + \mathbf{f}_i^\ell, \quad (7.1)$$

where $\mathbf{a}_i^\ell \in \mathbb{R}^d$ and $\mathbf{f}_i^\ell \in \mathbb{R}^d$ are the outputs of MHAT and FFN modules at layer ℓ , respectively. \mathbf{h}_i^0 represents a vector in the input \mathbf{I} with position of i . All hidden representations at layer ℓ corresponding to the whole input \mathbf{I} can be denoted by $\mathbf{H}^\ell = [\mathbf{h}_i^\ell]_{i=1}^N \in \mathbb{R}^{N \times d}$.

MHAT The masked multi-head attention (MHAT) module in each transformer layer ℓ contains four projection matrixes: $\mathbf{W}_Q^\ell, \mathbf{W}_K^\ell, \mathbf{W}_V^\ell, \mathbf{W}_O^\ell \in \mathbb{R}^{d \times d}$. For the multi-head attention, the input $\mathbf{H}^{\ell-1}$ is first projected to query, key and value: $\mathbf{Q}^\ell = \mathbf{H}^{\ell-1} \mathbf{W}_Q^\ell$, $\mathbf{K}^\ell = \mathbf{H}^{\ell-1} \mathbf{W}_K^\ell$, $\mathbf{V}^\ell = \mathbf{H}^{\ell-1} \mathbf{W}_V^\ell$. Then the projected query, key and value matrices are evenly split along the columns to H different heads: $\{\mathbf{Q}^{\ell,j}\}_{j=1}^H, \{\mathbf{K}^{\ell,j}\}_{j=1}^H, \{\mathbf{V}^{\ell,j}\}_{j=1}^H \in \mathbb{R}^{N \times \frac{d}{H}}$, respectively. After splitting \mathbf{W}_O^ℓ into $\{\mathbf{W}_O^{\ell,j}\}_{j=1}^H \in \mathbb{R}^{d \times \frac{d}{H}}$, we follow works

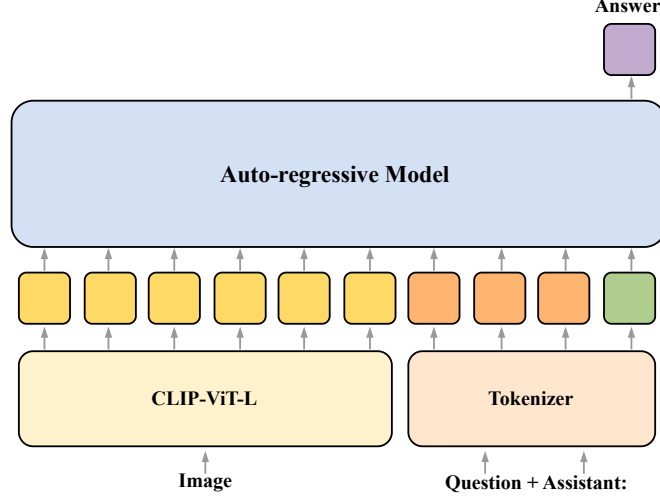


Figure 7.2: The typical architecture of multimodal large language model. It consists of an image encoder and a decoder-only large language model in which the multimodal information is integrated. We omitted the projection matrix for the visual patch feature as it is nonessential for our analysis.

in (Dar et al., 2022; Elhage et al., 2021; Geva et al., 2023) to represent the output of MHAT $\mathbf{A}^\ell = [\mathbf{a}_i^\ell]_{i=1}^N \in \mathbb{R}^{N \times d}$ at layer ℓ as the sum of the output from different heads

$$\mathbf{A}^\ell = \sum_{j=1}^H \mathbf{A}^{\ell,j} \mathbf{V}^{\ell,j} \mathbf{W}_O^{\ell,j} \quad (7.2)$$

$$\mathbf{A}^{\ell,j} = \text{softmax} \left(\frac{\mathbf{Q}^{\ell,j} (\mathbf{K}^{\ell,j})^T}{\sqrt{d/H}} + \mathbf{M}^{\ell,j} \right) \quad (7.3)$$

where $\mathbf{M}^{\ell,j}$ is a strictly upper triangular mask for $\mathbf{A}^{\ell,j}$ for j -th head at layer ℓ . For an auto-regressive transformer model, $\mathbf{M}^{\ell,j}$ is used to guarantee that every position of the input sequence cannot attend to succeeding positions and attends to all preceding positions. Therefore, for the element $M_{s,t}^{\ell,j}$ with the coordinate (s, t) in $\mathbf{M}^{\ell,j}$,

$$M_{s,t}^{\ell,j} = \begin{cases} -\infty & \text{if } t > s, \\ 0 & \text{otherwise.} \end{cases} \quad (7.4)$$

FFN FFN computes the output representation through

$$\mathbf{f}_j^\ell = \mathbf{W}_U^\ell \sigma \left(\mathbf{W}_B^\ell (\mathbf{a}_j^\ell + \mathbf{h}_j^{\ell-1}) \right) \quad (7.5)$$

where $\mathbf{W}_U^\ell \in \mathbb{R}^{d \times d_{\text{ff}}}$ and $\mathbf{W}_B^\ell \in \mathbb{R}^{d_{\text{ff}} \times d}$ are projection matrices with inner-dimensionality d_{ff} , and σ is a nonlinear activation function.

Output The hidden representation \mathbf{h}_N^L corresponding to the last position N of the input sequence at final layer L is projected by an unembedding matrix $\mathbf{E} \in \mathbb{R}^{|\mathcal{V}| \times d}$ and finally the probability distribution over all words in the vocabulary \mathcal{V} is computed by

$$P_N = \text{softmax}(\mathbf{E}\mathbf{h}_N^L), \quad (7.6)$$

where the word with the highest probability in P_N is the final prediction.

7.3.2 Attention knockout

In this paper, we mainly investigate the interaction between different modalities by locating and analyzing the information flow between them. We adopt a reverse-engineering approach to trace the information flow. Specifically, by intentionally blocking specific connections between different components in the computation process, we trace the information flow within them by observing changes in the probability of final prediction.

In MLLMs, the attention module (MHAT) is the only module, which has the function of communication between different types of hidden representation corresponding to different positions in the input sequence. Therefore, we intentionally block the attention edges between hidden representations at different token positions (termed as *attention knockout*) to trace the information flow between them. We take inspiration from the work of (Geva et al., 2023), where the authors use attention knockout to assess how the factual information is extracted from a single-modality LLM by evaluating the contribution of certain words in a sentence to last-position prediction. We extend this method to multimodal research by not only examining the contribution of each modality to the last-position prediction but also the transfer of information between different modalities.

Intuitively, when blocking the attention edge connecting two hidden representations corresponding to different positions of the input sequence leads to a significant deterioration in model performance, it suggests that there exists functionally important information transfer between these two representations. Therefore, we locate the information flow between different hidden representations corresponding to different positions of the input sequence, such as visual inputs, linguistic inputs, and the last position in the input sequence (the position of answer prediction), by blocking the attention edge between them in the MHAT module and observing the resulting decline in performance as compared to the original model with an intact attention pattern.

Formally, in order to prevent information flow from the hidden representations \mathbf{h}_s^ℓ with position s in the source set \mathbb{S} (e.g. all positions of visual tokens in the input sequence) to the hidden representations \mathbf{h}_t^ℓ with position t in the target set \mathbb{T} (e.g. all positions of linguistic tokens in the input sequence) at a specific layer $\ell < L$, we set the corresponding element $M_{s,t}^{\ell,j}$ in $\mathbf{M}^{\ell,j}$ to $-\infty$ and the updated Equation (7.4) is

$$M_{s,t}^{\ell,j} = \begin{cases} -\infty & \text{if } (t > s) \text{ or } (s \text{ in } \mathbb{S} \text{ and } t \text{ in } \mathbb{T}), \\ 0 & \text{otherwise.} \end{cases} \quad (7.7)$$



Name	Structural type	Semantic Type	Open / Binary	Image Example	Question Example	Answer	Num.
ChooseAttr	Choose	Attribute	Open		What was used to make the door, wood or metal?	Wood	1000
ChooseCat	Choose	Category	Open		Which piece of furniture is striated, bed or door?	Bed	1000
ChooseRel	Choose	Relation	Open		Is the door to the right or to the left of the bed?	Right	964
CompareAttr	Compare	Attribute	Open		What is common to the bike and the dog?	Color	570
LogicalObj	Logical	Object	Binary		Are there either women or men that are running?	No	991
QueryAttr	Query	Attribute	Open		In which part of the image is the dog?	Left	1000

Table 7.1: Different types of questions in our VQA dataset. The questions are categorized based on two dimensions: structure and semantics. The structural types define the question format, including: *Choose* for selecting between alternatives, *Compare* for comparisons between objects, *Logical* for logical inference, and *Query* for open-ended questions. The semantic types focus on the subject matter, covering *Object* existence, and *Attribute*, *Category*, *Relation* of objects. Additionally, questions are labeled as *Open* for open-ended queries or *Binary* for *yes/no* answers. The dataset is derived from the GQA dataset (Hudson and Manning, 2019). Due to space limitations, we present two images, noting that 50% of question samples in our dataset have unique images.

This prevents the token position in the target set from attending to that in the source set when MLLM generates the predicted answer.

7.4 Experimental setting

Setup Our paper investigates the inner working mechanism of MLLMs, focusing on visual question answering (VQA). Typically, the VQA setup involves an image and a corresponding question about this image, which the model needs to answer. We first investigate where the information from different modalities (image and textual question) is processed in MLLMs, and then how it is integrated within the model. Finally, we explore how the MLLM makes the final decision using this multimodal information.

Tasks and data We collect our data from the validation set of GQA dataset (Hudson and Manning, 2019). GQA is a dataset designed to support visual reasoning and compositional question-answering, offering the semantic and visual richness of real-world images. It is derived from the Visual Genome dataset, which includes detailed scene graph structures (Krishna et al., 2017). In GQA, the questions are categorized through two dimensions: structure and semantics. The former defines the question format (5 classes) and the latter refers to the semantic information for the main subject of the question (5 classes). The answers to these questions consist of only one word or phrase, which is easy to evaluate. Based on the two dimensions, the questions in GQA are categorized into 15 groups. We exclude most groups that consist of simple binary questions (*yes/no*) and demonstrate poor performance on the model investigated in this paper. Finally, we select 6 out of 15 groups (4 structural and 4 semantic classes) in which their performance is higher than 80% in average performance, as shown

in Table 7.1. The difficulty of our selected groups ranges from simple multimodal perception tasks to more complex multimodal reasoning. For example, *ChooseAttr* and *ChooseCat* ask about basic object attributes and categories for one object in the image, *ChooseRel* and *QueryAttr* involve spatial reasoning, and *CompareAttr* and *LogicalObj* require more challenging comparisons and logical reasoning between two objects in the image. For each selected group, we sample an average of 920 image-question pairs that are correctly predicted by most models used in this paper. For each model, we only use correctly predicted samples for analysis (Each model achieves an accuracy greater than 95% on the dataset we collected). More details about the dataset and the process of collection can be found in Appendix E.1.

Format Formally, given an *image* i and a *question* q (the *question* may contain *answer options* $os = [o_1, o_2]$), the model is expected to generate the *answer* a in the *last position* of the input sequence. In addition, the correct one in the options is referred to as the *true option* (o_t) while the other ones are denoted as the *false option* (o_f). Since the image, question and options might contain multiple input tokens, we use $\mathbb{I}, \mathbb{Q}, \mathbb{O}_t, \mathbb{O}_f$ to represent the set of input positions corresponding to *image*, *question*, *true option* and *false option*, respectively.

Evaluation We quantify the information flow between different input parts by evaluating the relative change in the probability of the answer word which is caused by blocking connections between different input parts (*attention knockout*). Formally, given an *image-question* pair, the MLLM generates the *answer* a with the highest probability p_1 from the output distribution P_N defined in Equation (7.6). After applying *attention knockout* at specific layers, we record the updated probability p_2 for the same answer a as in p_1 . The relative change in probability, $p_c\%$, is calculated as $p_c\% = ((p_2 - p_1) / p_1) \times 100$. In this paper, *attention knockout* is applied to each transformer layer (within a defined window) individually and evaluate their respective p_c values.

Models We investigate the current state-of-the-art and open-source multimodal large language models from the LLaVA series: *LLaVA-1.5-7b*, *LLaVA-1.5-13b* (Liu et al., 2024a), *LLaVA-v1.6-Vicuna-7b* (Liu et al., 2024b) and *Llama3-LLaVA-NEXT-8b* (Imm, 2024), which achieve state-of-the-art performance across a diverse range of 11 tasks including GQA. These models are trained on similar publicly available data but with different architectures and model sizes, which allows us to explore cross-modal interaction and processing over different architectures and minimize interference of unknown factors from training data. All these models have the same image encoder (*CLIP-ViT-L-336px* (Radford et al., 2021b)) but with different LLM: *Vicuna-v1.5-7b* (Zheng et al., 2023) with 32 layers (transformer blocks) in *LLaVA-1.5-7b* and *LLaVA-v1.6-Vicuna-7b*, *Vicuna-v1.5-13b* (Zheng et al., 2023) with 40 layers in *LLaVA-1.5-13b* and *Llama3-8b* (Dubey et al., 2024b) with 32 layers in *Llama3-LLaVA-NEXT-8b*, where *Vicuna-v1.5* is the standard and dense transformer architecture (Vaswani et al., 2017b) and *Llama3* adopts grouped query attention (Ainslie et al., 2023). In terms of image processing, *LLaVA-1.5-7b* and *LLaVA-1.5-13b* directly feed the original fixed-length image patch

features from the image encoder into the LLM as input tokens. In contrast, *LLaVA-v1.6-Vicuna-7b* and *Llama3-LLaVA-NEXT-8b* employ a dynamic high-resolution technique, which dynamically adjusts image resolution, resulting in variable-length image patch features with higher resolution. Due to space limitations, we will primarily present the results for the model *LLaVA-1.5-13b* in the subsequent sections of this paper, while similar findings for other models are presented in Appendix E.5.

7.5 Contribution of different modalities to the final prediction

For a successful answer prediction for the task of *VQA*, the MLLM will process the input image-question pair $[i, q]$ and generate the final answer from the output layer of the model corresponding to the *last position*. We first investigate whether the different modalities directly contribute to the final prediction.

Experiment 1 For each layer ℓ in the MLLM, we block the target set (the *last position*) from attending to each source set (\mathbb{I} or \mathbb{Q}) respectively at the layers within a window of $k = 9$ layers around the ℓ -th layer¹, and measure the change in the probability of the correct answer word. The *last position* means N -th position in the input sequence and it is also the first generated sub-word for the predicted answer. Typically, the answers contain a single word or phrase, which might sometimes be tokenized into several sub-word tokens. Therefore, we also conduct the same experiment and observe the probability change at the final generated sub-word of the predicted answer. Both the first and final generated sub-words yield similar results. Thus, we present all the results of the first generated sub-words in the main body of the paper, with details on the final sub-words provided in Appendix E.3.

Observation 1: the contribution to the prediction at the *last position* is derived from other input components, rather than the input itself at this position. First of all, as an auto-regressive model, it is assumed that the input generated from preceding steps at the final position already encompasses the crucial information required for predicting the correct answer. However, as shown in Figure 7.3, when we block the attention edge from the *last position* to itself ($Last \nrightarrow Last$), there is negligible change observed in the probability of final prediction. This implies that the input at the last position does not encompass crucial information for the final prediction of the model. The prediction is, therefore, mainly influenced by other parts of the input sequence.

Observation 2: The information from the *question* positions plays a direct and predominant role in influencing the final prediction. As shown in Figure 7.3, blocking attention from the *last position* to the hidden representations in \mathbb{Q} ($Question \nrightarrow Last$) results in significant reduction in prediction probabilities across all six tasks.

¹We experimented with different values of k , as described in Appendix E.2, and observed similar trends as in the analysis we present in this section.

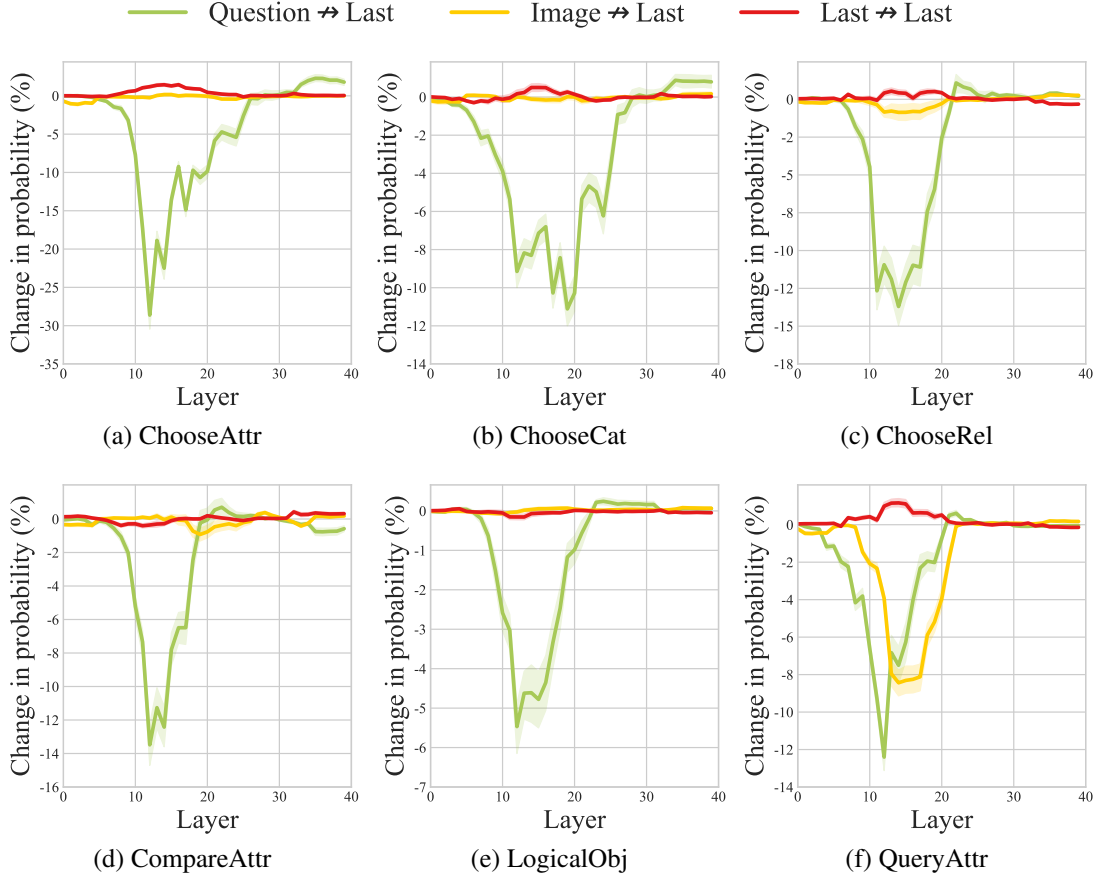


Figure 7.3: The relative changes in prediction probability on *LLaVA-1.5-13b* with six VQA tasks. The *Question* \rightarrow *Last*, *Image* \rightarrow *Last* and *Last* \rightarrow *Last* represent preventing *last position* from attending to *Question*, *Image* and itself respectively.

For example, in the *ChooseAttr* task, this decreases the prediction probability by up to $\sim 30\%$. This highlights the critical flow of information from \mathbb{Q} to the *last position*, directly affecting the final prediction. It is worth noting that this information flow pattern is observed primarily in the middle layers, where performance reductions consistently occur across all six tasks. In contrast, information from the *image* positions (\mathbb{I}) does not directly and significantly impact the final prediction in most tasks, except for *QueryAttr*, where a slight information flow from \mathbb{I} to the *last position* is observed. However, this direct influence is negligible compared to its indirect effects, discussed below. The additional experiment about the information flow between different parts of *question*, such as *options os* and object words, and *last position* can be found in Appendix E.6.

Experiment 2 As the MLLM is auto-regressive and the input format is *image* followed by the *question* in our setting, the information from the *image* (\mathbb{I}) can propagate to the positions of the *question* (\mathbb{Q}), but not the other way around. To establish whether this

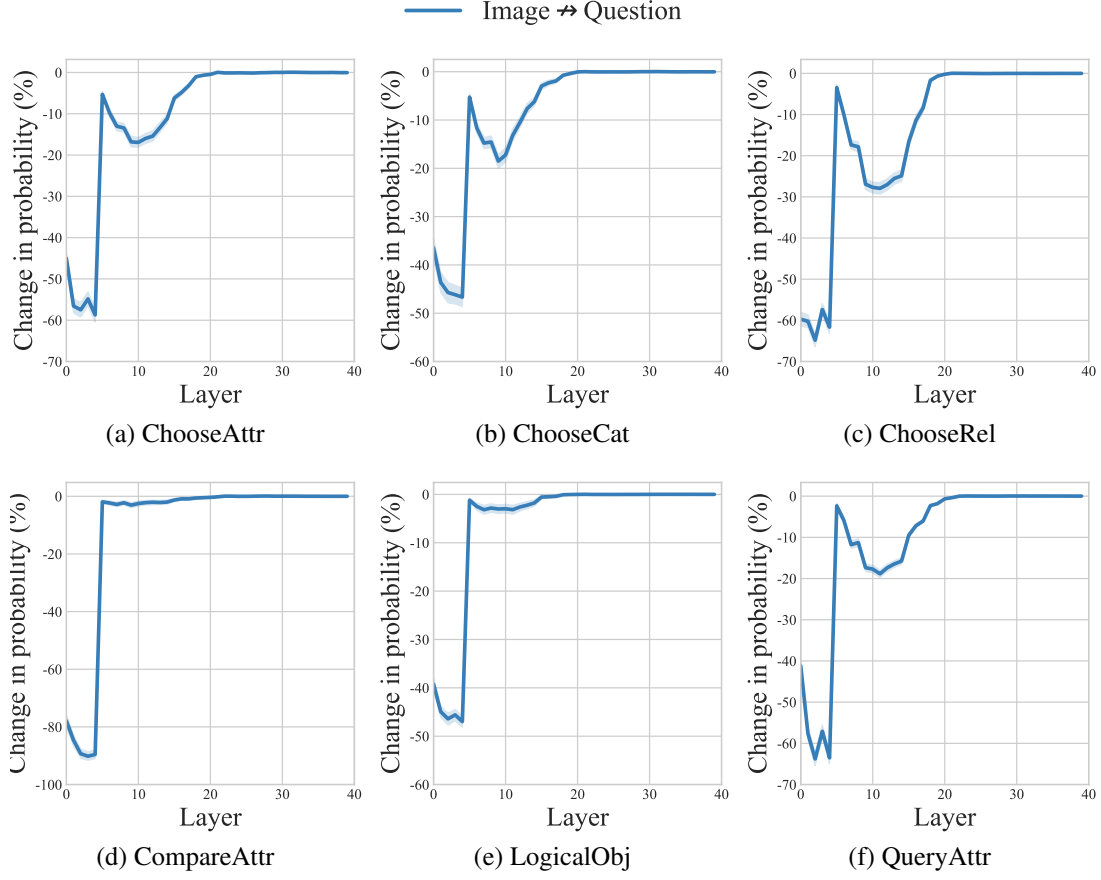


Figure 7.4: The relative changes in prediction probability when blocking attention edges from the *question* positions to the *image* positions on *LLaVA-1.5-13b* with six VQA tasks.

indeed occurs, for each layer ℓ , we block \mathbb{Q} from attending to \mathbb{I} with the same window size ($k = 9$) around the ℓ -th layer and observe the change in the probability of the answer word at the *last position* as above.

Observation: Information flow from the *image* positions to *question* positions occurs twice As shown in Figure 7.4, blocking the *question* positions from attending to the *image* positions leads to a reduction in prediction probability. This is visible in lower layers, in two different parts of the model. We first observe a sharp drop in layers $\sim 0 - 4$ and then a second smaller drop around 10th layer. This indicates a two-stage integration process of visual information into the representations of the *question*. In the first drop, attention knockout reduces the prediction probability by an average of $\sim 60\%$ across all six tasks. In the second drop, tasks such as *ChooseAttr*, *ChooseCat*, *ChooseRel*, and *QueryAttr* show another average reduction of $\sim 21\%$ while *CompareAttr* and *LogicalObj* exhibit smaller decreases. Despite the variability

in the magnitude of the reduction, the layers responsible for information flow remain consistent across all tasks, which is also observed during the first drop. The additional experiment about the information flow between *image* and different parts of *question*, such as *option os* and object words, can be found in Appendix E.6.

Overall information flow Given the input sequence: *image* and *question* with the corresponding sets of positions \mathbb{I} and \mathbb{Q} respectively, the MLLM first propagates information twice from the *image* positions to the *question* positions in the lower-to-middle layers of the MLLM. Subsequently, in the middle layers, the information flows from the *question* positions to the *last position* for the final prediction. Overall, this reveals the existence of distinct and disjoint stages in the computation process of different layers in MLLM, where critical information transfer points from different positions corresponding to different modalities are observed to influence the final predictions of the model. These findings are also observed in the other three MLLMs (Appendix E.5).

7.6 How is the linguistic and visual information integrated?

The results of the above analysis suggest a two-stage integration process of the two modalities within an MLLM. In this section, we further investigate how the information about specific visual and linguistic concepts is integrated across these two stages.

Experiment To investigate how the model uses the image to answer the question, we conducted attention knockout experiments at the level of individual objects and individual words. The dataset used in the paper consists of questions targeting specific objects and each object is annotated with the bounding box for a certain image region. Based on whether an image patch includes the corresponding bounding boxes (objects), we divide the input image patch features V into two groups: V_{obj} corresponding to the patches containing the objects mentioned in the question, and V_{oth} containing the remaining patches. Then, for each layer ℓ , we use the same *attention knockout* method to block the target set \mathbb{Q} from attending each source set, \mathbb{I}_{obj} and \mathbb{I}_{oth} , corresponding to the position of V_{obj} and V_{oth} in the input sequence respectively, at the layers with a window of $k = 9$ layers around the ℓ -th layer, and observe the change in the probability of the correct answer word.

Observation: Shifting focus from comprehensive representation to specific regions of interest As illustrated in Figure 7.5, blocking the attention edges between the position of V_{obj} and the *question* (*related image patches* \rightarrow *question*) and between the position of V_{oth} and the *question* (*other image patches* \rightarrow *question*) appear to account for the two performance drops observed in Figure 7.4, individually. Specifically, *other image patches* \rightarrow *question* clearly results in a significant and predominant reduction in prediction probability during the first stage of cross-modal integration, while *related image patches* \rightarrow *question* plays a dominant role at the second stage. It is noteworthy

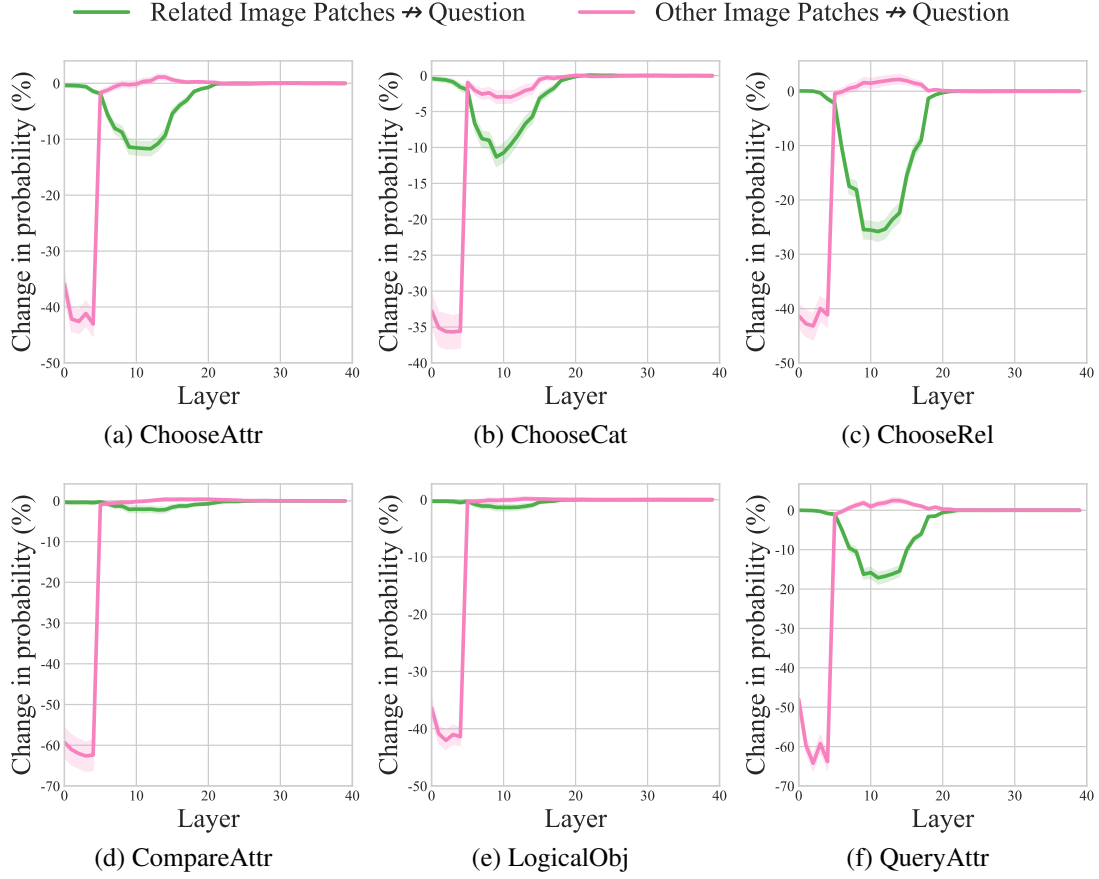


Figure 7.5: The relative changes in prediction probability on *LLaVA-1.5-13b* with six VQA tasks. *Related Image Patches*→*question* and *Other Image Patches*→*question* represent blocking the position of *question* from attending to that of different image patches, region of interest and remainder, respectively.

that both types of cross-modal information transfer occur in similar layers within the MLLM across all six tasks. Even for the *CompareAttr* and *LogicalObj* tasks, although slight changes in probability are observed during the second stage, the layers in which this happens remain consistent with those of the other tasks. This suggests in the lower layers, the model integrates the information from the whole image into the *question* positions building a more generic representation. And it is only in the later layers, that the model starts to pay attention to the specific regions in the image relevant to the *question*, fusing the more fine-grained linguistic and visual representations. The other MLLMs also present similar results as shown in Appendix E.5. The additional more fine-grained analysis on intervention of the attention edge between object words in *question* and image region can be found in Appendix E.6. Moreover, we find compared with *LLaVA-1.5-13b*, the model *LLaVA-1.5-7b* with smaller size has less information flow from the position of V_{oth} to that of *question* in the first stage, as shown in Appendix E.5.

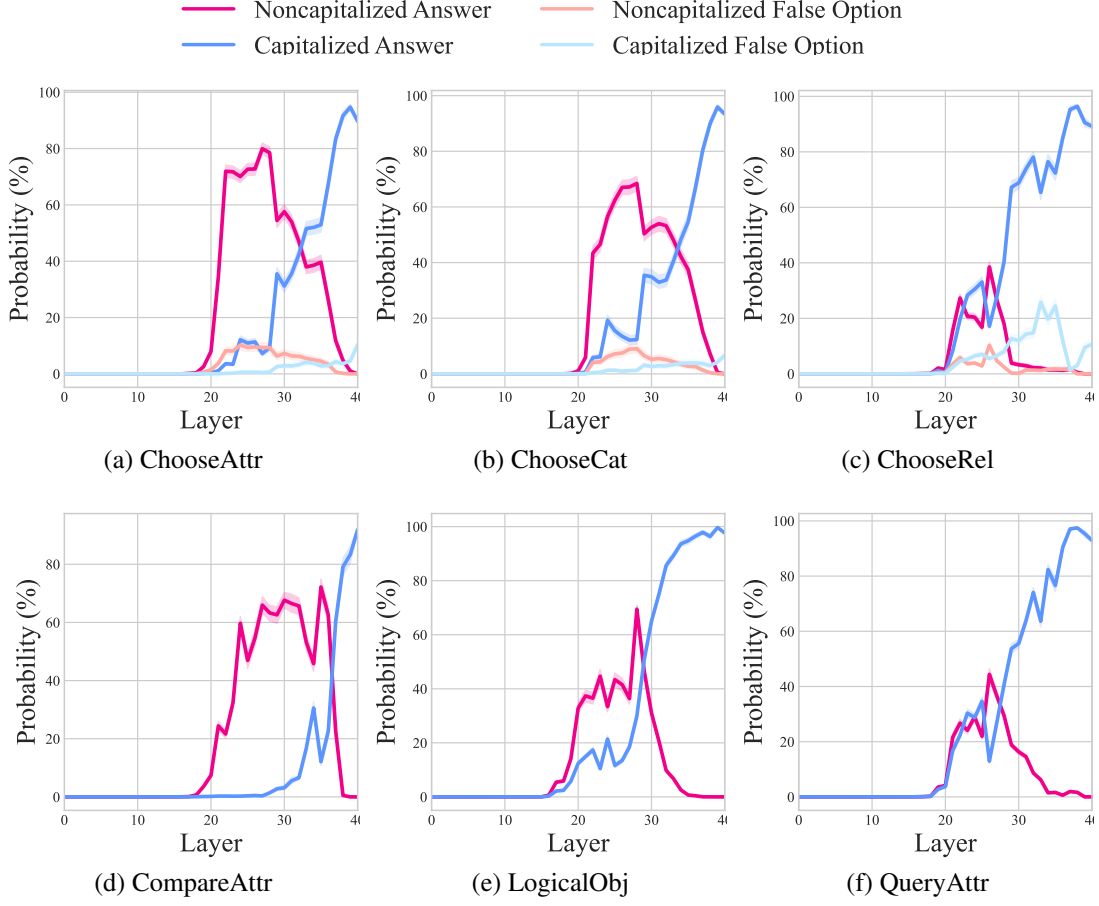


Figure 7.6: The probability of the answer word at the *last position* across all layers in *LLaVA-1.5-13b* with six VQA tasks. *Capitalized Answer* and *Noncapitalized Answer* represent the answer word with or without the uppercase of the initial letter, respectively. As the tasks of *ChooseAttr*, *ChooseCat* and *ChooseRel* contain *false option*, we also provide the probability of it.

7.7 How is the final answer generated?

Experiment To track the process of answer generation in the MLLM, motivated by the approach of logit lens ([Nostalgebraist](#)), we monitor the probability of the correct answer from the hidden representations at the *last position* of the input sequence across all layers. Formally, for each layer ℓ at the last position N , we use the unembedding matrix \mathbf{E} (as defined in Equation (7.6)) to compute the probability distribution over the entire vocabulary \mathcal{V} :

$$P_N^\ell = \text{softmax}(\mathbf{E}\mathbf{h}_N^\ell), \quad (7.8)$$

where the probability of the target answer word w_a is given by the corresponding entry in P_N^ℓ , denoted as $P_N^\ell(w_a)$. As the tokenizer in most MLLMs distinguishes the case

of the word, especially the initial letter of the word, we monitor the probability of the answer word with both those starting with uppercase (*Capitalized Answer*) and lowercase letters (*Noncapitalized Answer*).

Observation 1: The model is able to predict the correct answer starting at the layer immediately following multimodal integration As illustrated in Figure 7.6, the probability of the answer word with a lowercase initial letter (*Noncapitalized Answer*) rises sharply from near 0 to a range of $\sim 20\%$ to $\sim 70\%$ across the six VQA tasks, around the model’s middle layers. This implies that the model rapidly acquires the capability to predict correct answers in these middle layers, where the phase of multimodal information integration has just fully completed (see Figure 7.5) and the multimodal information is still transforming from *question* position to *last position* (see Figure 7.3).

Observation 2: Semantic generation is followed by syntactic refinement As shown in Figure 7.6, across all VQA tasks, the probability of *Noncapitalized Answer* starts to gradually decrease to nearly zero after an increase in middle layers. In contrast, the probability of *Capitalized Answer* remains low in the initial layers following 20th but starts to increase in subsequent layers. This indicates the model has already semantically inferred the answer by about halfway through layers and in the higher layers, the model starts to refine the syntactic correctness of the answer. Similar findings on other models are shown in Appendix E.5.

7.8 Conclusion

In this paper, we unveil the inner working mechanisms of auto-regressive multimodal large language models in handling multimodal tasks. Our experiments reveal that different multimodal tasks exhibit similar processing patterns within the model. Specifically, when provided with an input consisting of an image and a question, within the lower-to-middle layers, the model initially propagates the overall image information into the hidden representations of the *question* in the lower layers and then the model selectively transfers only the question-relevant image information into the hidden representations of the *question*, facilitating multimodal information integration. In the middle layers, this integrated multimodal information is propagated to the hidden representation of the *last position* for the final prediction. In addition, we find that the answers are initially generated in lowercase form in middle layers and then converted to uppercase for the first letter in higher layers. These findings enhance the transparency of such models, offering new research directions for better understanding the interaction of the two modalities in MLLMs and ultimately leading to improved model designs.

This dissertation focused on three challenges from the domains of language and vision, individually and in combination: integrating commonsense knowledge to enhance reasoning ability in multimodal systems, developing efficient adaptation methods for large foundation models (both vision and large language models), and improving the transparency of multimodal large language models (MLLMs).

Part I: Commonsense Knowledge Enhanced Multimodal Reasoning

In Chapter 3, we investigated the integration of commonsense knowledge into multimodal models by proposing the CK-Transformer framework. CK-Transformer effectively incorporates relevant commonsense knowledge into visual object representations, thereby enhancing the reasoning capabilities of multimodal models for the referring expression comprehension (REC) task. Extensive experimental results demonstrated that CK-Transformer significantly improves performance, particularly in scenarios requiring the comprehension of commonsense knowledge, such as social, temporal, and physical commonsense.

Insights and prospects The successful integration of commonsense knowledge via the CK-Transformer underscores the critical role that explicit knowledge plays in closing the reasoning gap of current multimodal models. Key insights indicate that traditional multimodal architectures predominantly rely on superficial pattern recognition, which significantly limits their reasoning capabilities in real-world scenarios that inherently require commonsense understanding. Incorporating structured or non-structured commonsense knowledge directly into visual object representations provides a robust approach to enable deeper, human-like inferential reasoning. Looking forward, the deficiency in commonsense reasoning capabilities primarily originates from existing pre-training methodologies, which predominantly rely on image-caption pairs to learn modality alignment while overlooking the commonsense knowledge essential for complex tasks. Therefore, future research should prioritize developing novel pre-training paradigms that implicitly embed commonsense knowledge within model representations, thereby

enabling models to effectively leverage this knowledge to enhance reasoning capabilities in downstream tasks without requiring explicit knowledge integration during inference. Additionally, extending commonsense-enhanced reasoning to a broader spectrum of multimodal tasks beyond referring expression comprehension, such as video understanding tasks and human-machine collaborative scenarios, holds significant potential for improving the real-world applicability and robustness of multimodal systems.

Part II: Efficient Adaptation

For efficient adaptation of large foundation models, we introduced a new paradigm for parameter-efficient fine-tuning in Chapter 4. Our method, termed Gradient-based Parameter Selection (GPS), differs from existing approaches by selectively fine-tuning only a minimal subset of a pretrained model’s parameters based on gradient significance, without adding any additional parameters. Empirical evaluations showed that GPS achieves comparable or superior performance to full fine-tuning across various vision tasks, including image classification and semantic segmentation.

Building upon this foundation, in Chapter 5, we extended our efficient fine-tuning approach to natural language processing tasks. Considering the dramatic increase in size and computational requirements of large language models compared to vision models, we proposed NeuronAda, a featherlight, scalable, and task-agnostic fine-tuning framework. NeuronAda further reduces training memory and time relative to GPS and eliminates the necessity for preliminary gradient computations. Experimental validations across multiple natural language generation and understanding benchmarks demonstrated that NeuronAda outperforms strong baselines, delivering robust generalization with exceptionally few trainable parameters (less than 0.01%) under stringent memory constraints.

Further exploration of efficient adaptation strategies in multi-task and multilingual scenarios was presented through the sparse training perspective in Chapter 6. Comprehensive experiments revealed that sparse training effectively reduces gradient conflicts between tasks by transforming high-dimensional parameter spaces into lower-dimensional ones, all while preserving the integrity of the original model. Moreover, combining sparse training with other gradient manipulation strategies significantly enhanced performance in multi-task learning scenarios.

Insights and prospects Our investigation into parameter-efficient fine-tuning (PEFT) methods highlights the efficacy of sparse training through selective parameter updates without structural augmentation as a scalable, fine-grained, and memory-efficient PEFT strategy for adapting large-scale models. Both GPS and NeuroAda leverage the inherent representational capacity of pretrained networks by updating only a minimal yet semantically influential subset of the original parameters. Crucially, these methods introduce neuron-wise parameter selection, which activates each neuron’s potential contribution to downstream task adaptation during the fine-tuning process. This approach offers a more fine-grained alternative to conventional layer-level or module-level tuning, achieving

task specialization through neuron-level intervention. These findings further suggest a conceptual shift in the understanding of fine-tuning: rather than expanding a model’s capacity via the addition of new components (e.g. LoRA, Adapter), adaptation can be framed as the reconfiguration of existing representational capacity within the model. From this perspective, fine-tuning becomes a process of selectively reactivating and steering the internal semantics encoded in pretrained neurons, thus achieving efficient adaptation without inflating the model’s parameter footprint.

For future work, the proposed sparse fine-tuning strategy holds strong potential for extension to continual learning and lifelong adaptation scenarios, owing to its minimal parameter update requirements. In such settings, parameter selection strategies could be dynamically updated to reflect evolving task distributions or domain shifts, potentially enabling efficient and stable model adaptation over time without catastrophic forgetting. In addition, while current parameter selection in NeuroAda relies on static heuristics such as weight magnitude or gradient value, learning-to-select approaches where the selection mask is jointly optimized or guided by meta-learning signals may offer further gains in both efficiency and performance. Exploring such methods could enable adaptive selection schemes that better account for task difficulty, data availability, or model capacity.

Part III: Transparency

Finally, in Chapter 7, we investigated transparency and interpretability in multimodal large language models (MLLMs) using attention knockout methods to systematically examine internal mechanisms of popular decoder-only autoregressive multimodal architectures from a perspective of the information flow between different-modality information. Our analysis disclosed a structured and sequential integration of linguistic and visual information: lower-to-middle layers first propagate general image information into the question representations and subsequently selectively transport question-relevant image details. This integrated multimodal information is then transferred to the hidden representation of the final prediction position in the middle layers. Additionally, we observed that answers are initially generated in lowercase at intermediate layers before capitalization occurs at higher layers. These insights substantially advance our understanding of MLLM operations, enhancing model interpretability and offering valuable guidance for developing trustworthy multimodal AI systems.

Insights and prospects This study provided critical insights into their internal operational dynamics, notably the sequential and structured multimodal information integration process. We discovered that monomodal and multimodal representations evolve in the multimodal large language model using a layer-wise manner, transitioning from general to task-specific integration, thereby refining our understanding of model reasoning processes. Observing capitalization and answer generation at distinct layers further highlights nuanced yet systematic behaviors, demonstrating how internal model states progressively refine predictions. Prospectively, deeper interpretability studies are

encouraged, focusing on layer-wise and neuron-level analyses to unravel finer-grained interaction patterns between modalities. Additionally, applying these interpretability insights to refine model architectures or training processes could significantly contribute to the development of more transparent, trustworthy, and accountable multimodal systems suitable for sensitive or safety-critical applications.

A Appendix to Chapter 3

A.1 Referring expression comprehension

Early approaches to REC use joint embedding of image and language by the combination of Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), and predict the target object that has the maximum probability given an input expression and an image (Hu et al., 2016; Mao et al., 2016; Zhang et al., 2018a). In order to model different types of information encoded in input expression (subject appearance, location, and relationship to other objects), subsequent work used modular (attention) networks, to “match” the input to corresponding regions in the image, predicting as the target the region with the highest matched score (Hu et al., 2017; Yu et al., 2018).

A.2 UNITER

UNITER is trained using four pre-training tasks, Masked Language Modeling (MLM), Masked Region Modeling (MRM), Image–Text Matching (ITM), and Word–Region Alignment (WRA), on four large-scale image–text datasets, COCO (Lin et al., 2014), Visual Genome (Krishna et al., 2016), Conceptual Captions (Sharma et al., 2018), and SBU Captions (Ordonez et al., 2011). This enables UNITER to capture fine-grained alignments between images and language. The architecture of UNITER is similar to BERT (Devlin et al., 2018) apart from the input and the output. Specifically, the input consists of an image (a set of visual region candidates), a sentence and [CLS] token, and they respectively lead to different outputs, i.e. vision output, language output and cross-modality output on the top of UNITER.

A.3 Input embedding

Same with UNITER, we extract the input embeddings E_{Inp} consisting of an image and a text embedding corresponding to the object candidate I and text (an expression e or a fact f_i) respectively.

Image embedding The image embedding E_I is computed by summing three types of embeddings: visual feature embedding, visual geometry embedding and modality segment embedding. We first extract the visual features $V = \{v_1, v_2, \dots, v_n\}$ for all candidates using Faster R-CNN (pooled RoI features), and build a geometry feature $G = \{g_1, g_2, \dots, g_n\}$ for all candidates, where g_i is a 7-dimensional vector consisting of the geometry information of the bounding box corresponding to candidate c_i , namely normalized top, left, bottom, right coordinates, width, height, and area, denoted by $g_i = [x1, y1, x2, y2, w, h, w * h]$. Visual feature embeddings and visual geometry embeddings are generated by mapping the visual features and the geometry features into the same vector space through a fully connection layer fc :

$$E_I = LN(fc(V) + fc(G) + M_I) \quad (1)$$

where LN is the layer normalization layer and M_I is the modality segment embedding for the image input (like segment embedding for two sentence in BERT model).

Text embedding Similarly, the text embedding E_T is computed based on three different types of embeddings: token embedding, position embedding and modality embedding (Normally there is a fourth embedding, sentence segment embedding similarly to BERT, but, in our task, both expressions and facts consist of one sentence only and so only the first sentence segment embedding is used). Similar to BERT (Devlin et al., 2018), the text $W = \{w_1, w_2, \dots, w_u\}$ is first tokenized by WordPieces (Wu et al., 2016), which are then built into token embeddings $T = \{t_1, t_2, \dots, t_v\}$ and position embeddings $P = \{p_1, p_2, \dots, p_v\}$ according to their position in the text sequence.

$$E_T = LN(T + P + M_T) \quad (2)$$

where M_T is the modality segment embedding for the text input.

Input embedding The final input embedding E_{Inp} is computed by concatenating image embedding E_I and text embedding E_T :

$$E_{Inp} = [E_I, E_T] \quad (3)$$

A.4 Datasets

We use the KB-Ref dataset (Wang et al., 2020b) aiming at evaluating the task of referring expression comprehension based on commonsense knowledge. KB-Ref consists of 43,284 expressions for 1,805 object categories on 16,917 images, as well as a knowledge base of key-value (category-fact) pairs collected from three common knowledge

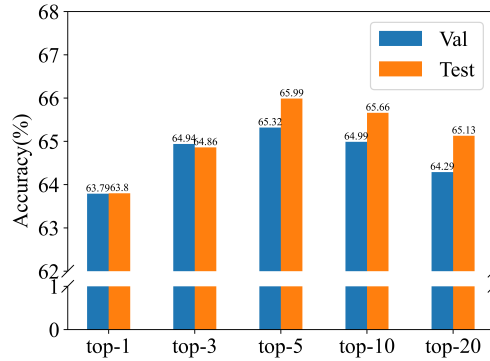


Figure 1: Accuracy across a varying number of facts (top-K).

resources: Wikipedia, ConceptNet (Speer et al., 2017b) and WebChild (Tandon et al., 2017b)). KB-Ref is split into a training set (31,284 expressions with 9,925 images), a validation set (4,000 expressions with 2,290 images) and a test set (8,000 expressions with 4,702 images).

We furthermore introduce commonsense knowledge into traditional tasks/datasets of referring expression comprehension, namely RefCOCO, RefCOCOg and RefCOCO+¹. The datasets are devised from the MSCOCO image dataset (Lin et al., 2014) but differ in the types of expressions and object candidate settings. Specifically, RefCOCO+ does not allow the use of absolute location words in the expressions, and most expressions focus on the appearance of the objects. The expressions in RefCOCOg are longer and contain more descriptive words. RefCOCO and RefCOCO+ contain more objects of the same category within an image.

A.5 Experimental settings

We extract image region features using Faster R-CNN with ResNet-101 (Ren et al., 2015) which was pre-trained on Visual Genome (Krishna et al., 2016) using object and attribute annotations (Anderson et al., 2018). For bounding box detection, we keep the bounding boxes with at least 0.2 confidence score indicating the extent of detection. In the CK-T, the hidden layer dimension is 768 and the number of multi-head attention heads is 12. The models are trained using Adamw (Loshchilov and Hutter, 2017) with a learning rate of $6e^{-5}$ and a batch size of 64 on Titan RTX GPUs. Our CK-Transformer has 120M parameters in total where fact-aware classifier has 34M and bi-modal encoder has 86M. As for UNITER model, we use same setting with *UNITER-base*, except for using Nvidia Apex² for speeding up training. The efficiency of our model is effected by the number of facts. Specifically, we train our CK-Transformer 10000 steps and a batch per step, which takes 2.5, 3, 3.5, 7 days with the number of facts: 3, 5, 10, 20

¹following Apache License 2.0

²<https://github.com/NVIDIA/apex>

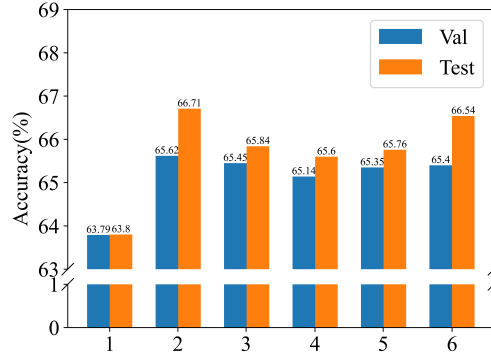


Figure 2: Accuracy across a varying number of fact-aware classifier block (M).

respectively. The CK-Transformer trains 3.8, 2.8, 2.1, 0.7 sample in average per second and tests 8.3, 7.3, 6.6, 1.1 sample per second.

A.6 Impact of CK-T structure

We explore the impact in performance on KB-Ref as we vary the number of top-K facts (K) and fact-aware classifier block (M) on the development set. We first keep the number of the fact-aware classifier block constant and set it to 1 to experiment with different values for K from 1 to 20. As shown in Figure 1, as K increases, performance starts to improves with a peak at K=5 before starting to gradually decrease performance.

In the second experiment, we keep K constant and set it to 3 and explore the effect of varying values for M. We observe that the highest accuracy is achieved with with top-3 facts and 2 integrator layers as shown in Figure 2.

A.7 Introducing facts in traditional REC tasks based on detection

The results of introducing facts in traditional REC tasks based on detected bbxes and categories are shown in Table 1. Compared to result based on ground-truth bbxes and categories (Table 3.3), the improvement on models based on detection is less or even worse than the models without facts.

A.8 McNemar Test

We also report the statistical significance for accuracy (shown in Table 3.3) on the tasks of RefCOCO. Specifically, we conduct the McNemar Test between models before and after introducing facts, on the test set of RefCOCO, RefCOCO+ and RefCOCOg, respectively. As shown in Table 2, as for Test set on RefCOCOg and Test A on RefCOCO $p\text{-value} = 1.19e-08$ and $p\text{-value} = 0.049$ (< 0.05) respectively, which means the proportion of errors is statistically significantly different after introducing facts as

Task		Accuracy (%)	
		U_{REC}	Intro Facts
Ref-COCO	Val ^d	81.15	81.06
	Test A ^d	86.85	86.87
	Test B ^d	74.48	73.97
Ref-COCO+	Val ^d	74.74	74.68
	Test A ^d	81.05	80.70
	Test B ^d	65.88	66.07
Ref-COCog	Val ^d	74.49	74.69
	Test ^d	75.24	74.86

Table 1: Introducing facts into RefCOCO, RefCOCO+ and RefCOCog based on detection (*d*).

Task		McNemar Test
		(<i>p</i> -value)
RefCOCO	Test A	0.049
	Test B	0.905
RefCOCO+	Test A	0.297
	Test B	0.966
RefCOCog	Test	$1.19e-08$

Table 2: The McNemar Test between models before and after introducing facts on the tasks of RefCOCOs.

compared to before. However, the change in the proportion of errors after introducing facts on other tasks (Test B on RefCOCO, Test A and Test B on RefCOCO+) is not statistically significant. This is reasonable, as the error from detection will affect the fact search (we first retrieve facts using the category) and thus more error information is introduced into CK-Transformer, which make the performance worse.

A.9 Example searched fact using different methods

As shown in Figure 3, there are several facts which are selected from three different fact search methods: CK-Transformer, CK-T-Uw/oImage and CK-T-Word2Vec. As we can see in the Table, normally the facts of CK-Transformer model (green) is the best relevant with the referring expression (blue) and the facts in CK-T-Word2Vec model is the worst relevant with the expression.


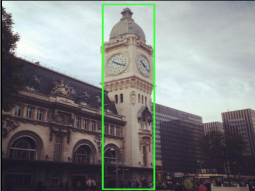
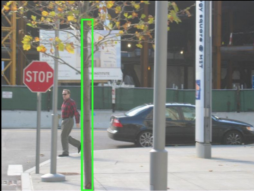

			
Exp: The tool under the mouse can improve the usability of the mouse	Exp: The tall building has instruments on the upper exterior walls that used as a reference to find out the time.	Exp: The most important part of the tree with branches and leaves	Exp: An access point as an underground public utility.
Fact: A mousepad enhances the usability of the mouse compared to using a mouse directly on a table.	Fact: Clock towers are a specific type of building which houses a turret clock and has one or more clock faces on the upper exterior walls.	Fact: The trunk is the most important part of the tree for timber production.	Fact: Manholes are often used as an access point for an underground public utility, allowing inspection, maintenance, and system upgrades.
Fact (Uw/oImage): A mousepad is a surface for placing and moving a computer mouse.	Fact (Uw/oImage): The tower has four clock faces, two of which are in diameter, at about high.	Fact (Uw/oImage): An automobile has a trunk.	Fact (Uw/oImage): A manhole is an opening to a confined space such as a shaft, utility vault, or large vessel.
Fact (Word2Vec): Mousepad on the mouse.	Fact (Word2Vec): Before the middle of the twentieth century, most people did not have watches, and prior to the 18th century even home clocks were rare.	Fact (Word2Vec): A trunk is an example of a box.	Fact (Word2Vec): These covers are traditionally made of metal, but may be constructed from precast concrete, glass reinforced plastic or other composite materials.

Figure 3: Example fact search process (using the top-1 fact) for different search methods: CK-T (green), CK-T-Uw/oImage (orange) and CK-T-Word2Vec (yellow).

B Appendix to Chapter 4

B.1 Details of experiments

Baseline description

Vision Transformer (ViT) As a transformer-based visual model, ViT (Dosovitskiy et al., 2020b) has been widely adopted in various visual tasks. Most of the experiments are conducted on pre-trained ViT architecture in this paper. Given an input image $I \in \mathbb{R}^{H \times W \times 3}$, before feeding the image into the Transformer, the image is partitioned into M patches and appended a [CLS] token for classification purposes, resulting in final input $x \in \mathbb{R}^{(M+1) \times d}$ where d is the dimension of the features. The Transformer typically consists of multiple blocks and each block contains a Multi-head Attention layer (MHA) and two MLP layers (Vaswani et al., 2017a).

Adapter Work in (Houlsby et al., 2019) proposed the Adapter method, which inserts multiple trainable layers (termed as Adapter) into the pre-trained Transformer encoder. Only the Adapter is updated during the fine-tuning stage. These layers can be inserted after either the Multi-head Attention layer or the MLP layer. Adapter comprises two projection matrices, one W^{down} for dimension reduction and the other W^{up} for feature reconstruction to the original dimension. Specifically, given the input $x \in \mathbb{R}^{(M+1) \times d}$, the output of the Adapter is

$$y = [W^{\text{up}} \phi(W^{\text{down}} x^T)]^T \quad (4)$$

where $W^{\text{up}} \in \mathbb{R}^{d' \times d}$, $W^{\text{down}} \in \mathbb{R}^{d \times d'}$ (where $d' \ll d$), and ϕ is a nonlinear activation function.

Prompt Visual prompt tuning (VPT) introduces learnable parameters (*i.e.*, prompts) into the input space (Jia et al., 2022a). When fine-tuning downstream tasks, the backbone is fixed, and just tuning these prompts. Formally, the given input $x \in \mathbb{R}^{(M+1) \times d}$ is concatenated with m introduced prompts $p \in \mathbb{R}^{m \times d}$. The final combined input is

$$x' = [x; p] \quad (5)$$

where $x' \in \mathbb{R}^{(M+1+m) \times d}$ will be feed into the Transformer. There are two versions of VPT, namely VPT-shallow and VPT-deep. The former introduces learnable prompts solely into the input space of the first layer, whereas the latter integrates them into each layer's input space.

Scale and shift feature SSF attempts to scale and shift the features between the layers of the pre-trained model by adding a linear transform layer (Lian et al., 2022). During fine-tuning the downstream tasks, only the linear transform layers are updated while the backbone remains frozen. The transform layer consists of two components, scale factor

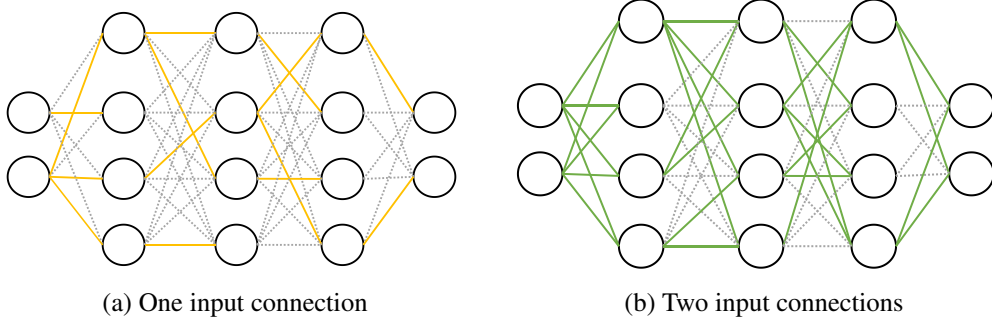


Figure 4: Different number of connections with highest gradient value among all input connections per neuron, such as (a) selecting only one input connection per neuron. (b) two input connections are selected per neuron.

Dataset	Params. (M)	Conne.s	Dataset	Params. (M)	Conne.s	Dataset	Params. (M)	Conne.s
CUB-200-2011	0.47	2	Pets	0.23	1	DMLab	0.20	1
NABirds	1.35	10	SVHN	0.20	1	KITTI/distance	0.20	1
Oxford Flowers	0.29	1	Sun397	0.55	1	dSprites/loc	0.21	1
Stanford Dogs	0.30	1	Patch Camelyon	0.30	2	dSprites/ori	0.21	1
Stanford Cars	1.07	10	EuroSAT	0.20	1	SmallNORB/azi	0.21	1
CIFAR-100*	0.29	1	Resisc45	0.24	1	SmallNORB/ele	0.20	1
Caltech101	0.29	1	Retinopathy	0.20	1	CIFAR-100	0.58	5
DTD	0.24	1	Clevr/count	0.30	1	CIFAR-100 (Swin)	0.82	5
Flowers102	0.29	1	Clevr/distance	0.20	1	CIFAR-100 (ConvNeXt)	0.78	5

Table 3: The number of learnable parameters and connections across all tasks. CIFAR-100* is a subset of CIFAR-100 in VTAB benchmark. In bracket is the model architecture, without bracket represents the one fine-tuned on ViT-B/16. Params. means the learnable parameters and the Conne. represents the number of selected input connections for each neuron in the network.

$\gamma \in \mathbb{R}^d$ and shift factor $\beta \in \mathbb{R}^d$, for feature transformation. To be specific, given the input $x \in \mathbb{R}^{(M+1) \times d}$, the output is calculated by

$$y = \gamma \odot x + \beta \quad (6)$$

where $y \in \mathbb{R}^{(M+1) \times d}$, \odot is the dot product.

The number of parameters on different tasks

For each neuron in the network, our GPS method selected at least one of the connections (weight or parameter) with the highest gradient value, among the input connections of the neuron, as shown in Figure 4a. For downstream tasks that need more learnable parameters to better fit the data, such as those tasks with dissimilar data distributions from the upstream dataset (such as NABirds) or larger amounts of data (such as CIFAR-100), our method can be easily extended by introducing more learnable parameters. Specifically, for each neuron, we can select multiple input connections with the highest

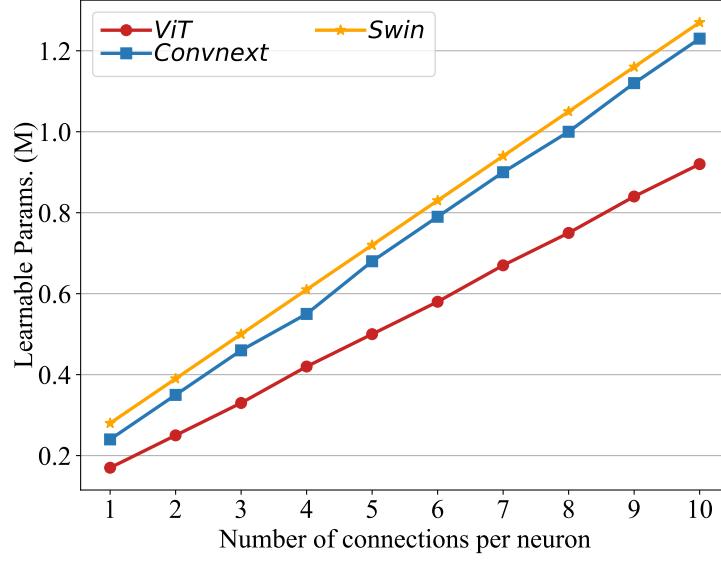


Figure 5: The number of learnable parameters with the different number of connections on ViT-B/16, Swin and Convnext archite. The learnable parameters do not contain the task-specific head.

gradient values instead of limiting them to just one, as shown in Figure 4b. Table 3 show the detailed statics on the number of parameters that are selected in our paper. For most of the tasks in this paper, we just select one of the connections. We also explore the relationship between the number of connections and the number of learnable parameters. As shown in Figure 5, with the increase in the number of selected connections with the highest gradient value among the input connections per neuron, the number of learnable parameters linear ascent.

Parameters distribution of *Net* selection

In contrast to our approach, a simple approach is to select the parameters for a specific task by selecting a certain percentage of parameters with the highest gradient from the entire network (He et al., 2023a). However, as shown in Figures 6a to 6e, most of the selected parameters are located in the upper layers, specifically block 12 and block 11. As a result, the network is primarily focused on fine-tuning abstract features while lacking the ability to fine-tune detailed information from shallower layers. Our approach addresses this challenge by carefully selecting the input connections for each individual neuron, resulting in our selected parameters being evenly distributed on the whole network, as shown in Figure 6f.

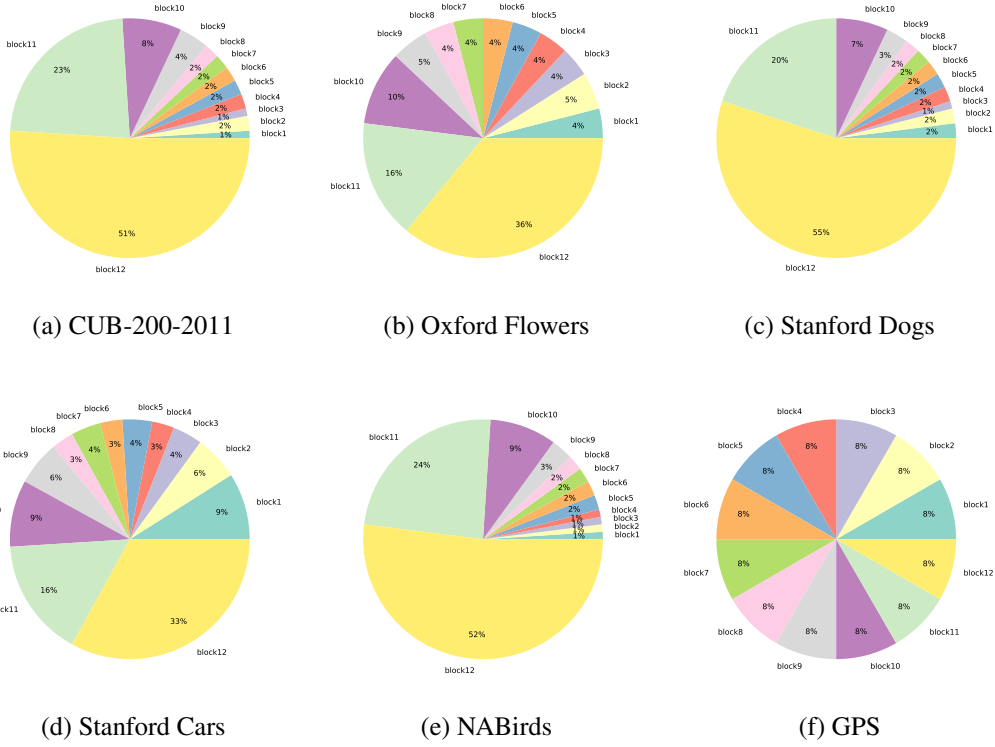


Figure 6: Distribution of the parameter over the whole network ViT-B/16 on 6 FGVC datasets (CUB-200-2011, Oxford Flowers, Stanford Dogs, Stanford Cars and NABirds). Selecting the 1% parameters with the highest gradient value from the whole network, instead of our method selecting at least one of the connections among all input connections per neuron. In contrast to this method, our GPS has the same distribution over different downstream tasks (f).

Method \ Dataset	ImageNet -1K (\uparrow)	ImageNet -A (\uparrow)	ImageNet -R (\uparrow)	ImageNet -C (\downarrow)
Full (Jia et al., 2022a)	83.58	34.49	51.29	46.47
Linear (Jia et al., 2022a)	82.04	33.91	52.87	46.91
Bias (Zaken et al., 2021)	82.74	42.12	55.94	41.90
Adapter (Houlsby et al., 2019)	82.72	42.21	54.13	42.65
VPT-Shallow (Jia et al., 2022a)	82.08	30.93	53.72	46.88
VPT-Deep (Jia et al., 2022a)	82.45	39.10	53.54	43.10
SSF (Lian et al., 2022)	83.10	45.88	56.77	41.47
GPS	83.91	46.11	57.00	42.04

Table 4: Performance comparisons on the ImageNet with different model architectures.

B.2 Additional experiments

Robustness and OOD datasets

In addition to standard classification tasks, we further analyze the robustness and OOD generalization ability of GPS. Based on the Imagenet-A, ImageNet-R, and ImageNet-C datasets, we first fine-tune the model on ImageNet-1K, and then test the fine-tuned model on the three datasets respectively. The results are shown in Table 4. GPS not only achieves the best performance on the standard ImageNet-1K classification task but also achieves good performance in robustness and generalization tests. Among them, GPS achieves the best results on ImageNet-A and ImageNet-R, outperforms the previous optimal SSF by 0.23%, reflecting the strong stability and generalization ability of our method. On ImageNet-C, GPS performs slightly worse, lagging behind SSF, but still higher than addition-based Adapter and VPT. This result indicates that our method can quickly adapt to the data distribution of downstream tasks, but it needs to be improved in anti-interference.

More experiments on different architecture

As mentioned in the main body of our paper, our method is model-agnostic, we further compare GPS with other fine-tuning methods across ViT-B/16, Swin-B, and ConvNeXt-B architectures on the ImageNet-1k (Deng et al., 2009) and CIFAR-100 (Krizhevsky et al., 2009) datasets.

CIFAR-100 As shown in Table 6, unlike FGVC and VTAB, GPS and other efficient tuning methods have difficulty in achieving competitive performance as full tuning on CIFAR-100. This may be due to that CIFAR-100 contains more training data, allowing all parameters of the entire model to be adequately trained, which seriously

Method \ Dataset	CUB-200 -2011	NABrids	Oxford Flowers	Stanford Dogs	Stanford Cars	Mean Acc.	Mean Params. (M)	Mean Params. (%)
ViT-B/16 + Full	87.3	82.7	98.8	89.4	84.5	88.54	85.98	100.00
ViT-B/16 + Linear	85.3	75.9	97.9	86.2	51.3	79.32	0.18	0.21
ViT-B/16 + SSF	89.5	85.7	99.6	89.6	89.2	90.72	0.39	0.45
ViT-B/16 + GPS (Ours)	89.9	86.7	99.7	92.2	90.4	91.78	0.66	0.77
Swin-B + Full	90.7	89.8	99.5	88.9	93.2	92.42	86.98	100.00
Swin-B + Linear	90.6	86.8	99.2	88.3	74.6	87.90	0.24	0.28
Swin-B + SSF	90.5	88.4	99.7	88.7	90.4	91.54	0.49	0.56
Swin-B + GPS (Ours)	90.8	88.9	99.7	92.7	90.7	92.56	0.83	0.95
ConvNeXt-B + Full	91.2	90.4	99.6	89.9	94.1	93.04	87.81	100.00
ConvNeXt-B + Linear	90.6	86.9	99.3	89.7	73.5	88.00	0.24	0.28
ConvNeXt-B + SSF	90.8	89.0	99.7	90.4	92.5	92.48	0.50	0.56
ConvNeXt-B + GPS (Ours)	91.0	89.6	99.7	93.7	92.6	93.32	0.79	0.90

Table 5: Performance comparisons on FGVC benchmark with different model architectures.

Architecture	ViT-B/16		Swin-B		ConvNeXt-B	
	Acc.	Params.(%)	Acc.	Params.(%)	Acc.	Params.(%)
Full (Jia et al., 2022a)	93.82	100.00	93.85	100.00	94.14	100.00
Linear (Jia et al., 2022a)	88.70	0.09	89.27	0.12	89.20	0.12
Bias (Zaken et al., 2021)	93.39	0.21	92.19	0.28	92.80	0.27
Adapter (Houlsby et al., 2019)	93.34	0.36	92.49	0.38	92.86	0.52
VPT-Shallow (Jia et al., 2022a)	90.38	1.07	90.02	0.15	-	-
VPT-Deep (Jia et al., 2022a)	93.17	1.43	92.62	0.81	-	-
SSF (Lian et al., 2022)	<u>93.99</u>	0.33	93.06	0.43	93.45	0.42
GPS (Ours)	94.02	0.68	<u>93.55</u>	0.96	<u>93.58</u>	0.90

Table 6: Performance comparisons on the CIFAR-100 with different model architectures.

reduces the advantages of efficient fine-tuning methods. However, GPS still outperforms all previous parameter-efficient tuning methods (Bias, Adapter, VPT, and SSF) and reduces the gap with full fine-tuning to less than 0.5% on all architectures, which further demonstrates the adaptability of our approach to different models.

ImageNet-1k Similar to the results on CIFAR-100, ImageNet-1K contains more training data, which makes it harder for parameter-efficient fine-tuning algorithms to achieve the same accuracy as full fine-tuning, as shown in Table 7. However, GPS still outperforms full fine-tuning by 0.33% on ViT structure, and outperforms the previous SOTA method SSF on Swin and ConvNeXt structures respectively, which further shows the generalization of GPS to different model structures.

FGVC As mentioned in the main body of our paper, our method achieves the best result on FGVC benchmark. Table 5 shows the full results of Table 4.4 in the main body.

Architecture	ViT-B/16		Swin-B		ConvNeXt-B	
	Acc.	Params.(%)	Acc.	Params.(%)	Acc.	Params.(%)
Full (Jia et al., 2022a)	<u>83.58</u>	100.00	85.20	100.00	85.80	100.00
Linear (Jia et al., 2022a)	82.04	0.89	83.25	1.17	84.05	1.16
Bias (Zaken et al., 2021)	82.74	1.00	83.92	1.32	84.63	1.31
Adapter (Houlsby et al., 2019)	82.72	1.16	83.82	1.43	84.49	1.54
VPT-Shallow (Jia et al., 2022a)	82.08	1.06	83.29	1.19	-	-
VPT-Deep (Jia et al., 2022a)	82.45	1.42	83.44	1.85	-	-
SSF (Lian et al., 2022)	83.10	1.12	84.40	1.47	84.85	1.44
GPS (ours)	83.91	1.37	<u>84.43</u>	1.96	<u>84.87</u>	1.90

Table 7: Performance comparisons on the ImageNet-1k with different model architectures.

Among all three model architectures, GPS consistently outperforms all other baselines, demonstrating its model-agnostic advantage.

Data-efficient tuning

Recent advances in large foundation model fine-tuning have shown considerable promise in reaching state-of-the-art performance on various tasks. However, in order to reach high accuracy, these methods often need significant volumes of training data, which may be time-consuming and costly to obtain. Here we demonstrate that our method is data efficient, that is, with such a few-shot setting, our method requires only a small amount of training data for tuning to achieve outstanding results that other approaches do not. Specifically, we fine-tune the ViT-B/16 by selecting only k samples for each class in the ImageNet dataset to form a few-shot training set. The value of k and the accuracy of predicted results are illustrated in Figure 7, which demonstrates the excellent data efficiency of our method especially in extreme cases like $k=1$.

Random seed for impacts of different selection schemes and ablations

We conduct experiments with three random seeds to investigate the robustness of our method. As shown in Table 8, Our parameter selection method significantly outperforms the other methods with small randomness. Table 8 is a supplementary of Table 4.6 in the main body of our paper.

Relationship between the size of the dataset and tunable parameters.

Our consistent finding is that the accuracy drops as the number of parameters rises for all experiments (different dataset sizes), and most of them achieve optimal results using only 0.37% parameters, as shown in Figure 9.

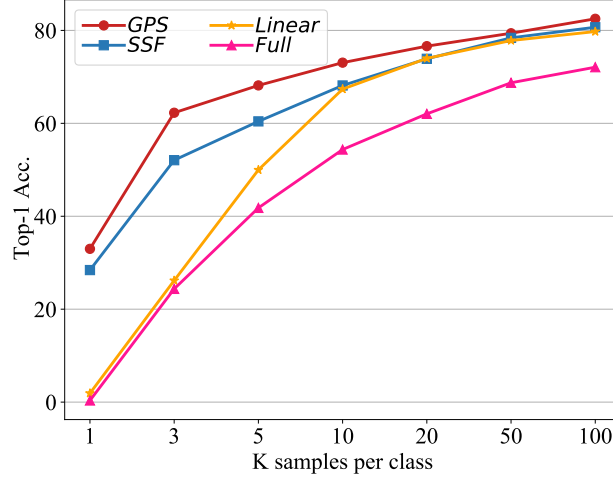


Figure 7: The performance comparison of different fine-tuning methods under k values for each class on ImageNet dataset. Our method is always above the curve of the others. The advantage of our approach is particularly evident in extreme cases where data is extremely scarce (e.g., $k=1$).

		CUB	NABirds	Flowers	Cars	Dogs
(a)	Net Layer	86.86 ± 0.21	86.55 ± 0.03	99.62 ± 0.01	89.65 ± 0.12	91.32 ± 0.07
		87.30 ± 0.13	86.79 ± 0.08	99.64 ± 0.01	90.03 ± 0.13	91.90 ± 0.11
(b)	Net Random	86.60 ± 0.10	85.98 ± 0.07	99.61 ± 0.01	89.10 ± 0.12	91.34 ± 0.12
	Neuron Random	87.17 ± 0.15	86.02 ± 0.10	99.62 ± 0.01	89.52 ± 0.23	91.82 ± 0.23
	Magnitude	87.29 ± 0.12	85.99 ± 0.08	99.62 ± 0.00	89.29 ± 0.02	91.30 ± 0.02
(c)	Head+CE	87.05 ± 0.19	86.20 ± 0.14	99.64 ± 0.01	89.25 ± 0.09	91.29 ± 0.01
	GPS	88.07 ± 0.11	86.64 ± 0.03	99.69 ± 0.01	90.10 ± 0.10	92.30 ± 0.10

Table 8: Impacts of different selection schemes and ablations. (a) Different selection levels. (b) Different selection criteria. (c) Different ways to calculate gradients.

Data scale needed for parameter selection.

Benchmark datasets are relatively small, such as VTAB (19 tasks) with only 1000 samples each, so we used full train set to compute gradients. Additionally, our experiments demonstrate selected parameters from random limited samples significantly overlap with those from the full dataset, and yield similar accuracy: $93.8\% \pm 0.097$, as shown in Figure 8.

The performance with different sizes of pre-trained models.

We investigate the performance of GPS on ViT model with different scales (small, base, large and huge). As shown in Table 9, the performance of our method remains stable

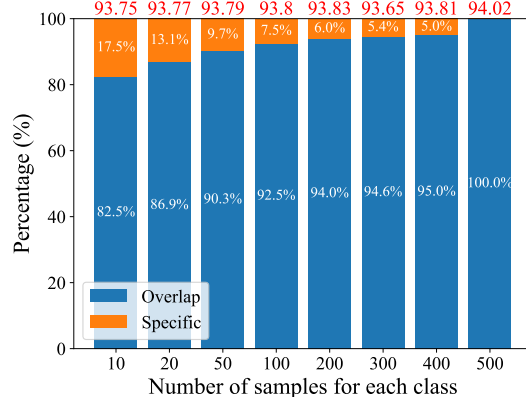


Figure 8: Different number of samples for parameter selection on Cifar-100 that contains 100 classes and 500 images per class. Overlap: the ratio of parameters selected using a limited amount of data to those selected using the entire dataset. The red number at the top: the accuracy (%) trained on the full training dataset and just tuning the corresponding selected parameters.

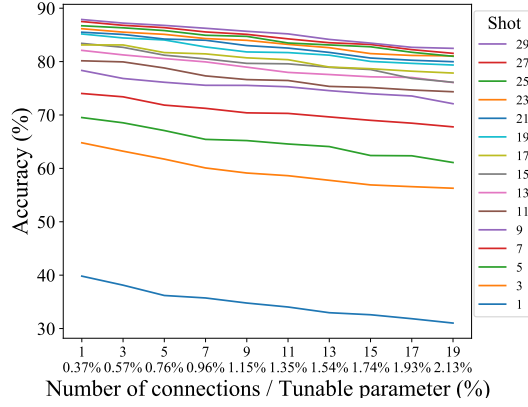


Figure 9: Performance with different dataset sizes (N shot per class) and number of selected parameters on CUB task.

Models	Vit-Small	Vit-Base	Vit-Large	Vit-Huge
SSF	88.47	90.72	91.54	78.75
GPS (Ours)	89.95	91.78	92.21	90.01

Table 9: Performance comparisons on FGVC with different scales of the pre-trained model. The number in the table is the mean accuracy over tasks in FGVC (5 tasks).

when scaling up to larger models, while SSF experiences a significant drop on Vit-H.

Method	Adapter	VPT-Shallow	VPT-Deep	SSF	GPS(ours)
# Params.	$2Ldd'$	nd	nLd	mLd	0
# FLOPs	$2N^2Ldd'$	$2n(2N^2 + n)d$	$2n(2N^2 + n)Ld$	mN^2Ld	0

Table 10: Extra parameters (params.) and FLOPs on ViT over PEFT methods. The dimension and number of input tokens are d and N^2 ($N \times N$ patches). Adapter projects features from d to d' ($2dd'$ extra params.). VPT inserts n prompts (nd extra params.). SSF adds a linear coefficient (md extra params.). L : number of layers. GPS does not introduce any parameters.

Computational cost for parameter selection

The cost is minimal compared to the training process as it only involves computing gradients without any parameter updates. For CUB example, selection only takes 30s, which is negligible compared to the overall training process (3960s). Additionally, our experiments demonstrate that using just a few mini-batches is sufficient to achieve comparable results to employing the full dataset, which could further reduce the computational cost as shown in Figure 8.

FLOPS.

The FLOPS measures computer performance, but you may have meant FLOPs. GPS does not introduce any extra parameters, resulting in no increased computation compared to other PEFT methods as shown in Table 10.

B.3 Visualizations

Semantic segmentation

As mentioned in the main body of our paper, our approach demonstrates highly promising results in the field of semantic segmentation. We apply our method on the pre-trained strong segmentation model (SAM) (Kirillov et al., 2023a) and fine-tune on a medical segmentation task – polyp segmentation (Jha et al., 2020a). Here, we present more case visualizations, which could directly show the effectiveness of our method, as shown in Figure 10.

Distribution of selected parameters across various tasks

As we select different subsets of parameters from the original model for different downstream tasks, a normal question is how different the distribution of the selected parameters is across different tasks. we test on ViT-B/16 with 6 downstream tasks from VTAB (two from Natural, two from Specialized and the other two from Structured). As shown in Figure 11, the chosen parameters exhibit a tendency towards 2/3 shared parameters and 1/3 task-specific parameters, despite the dissimilar data

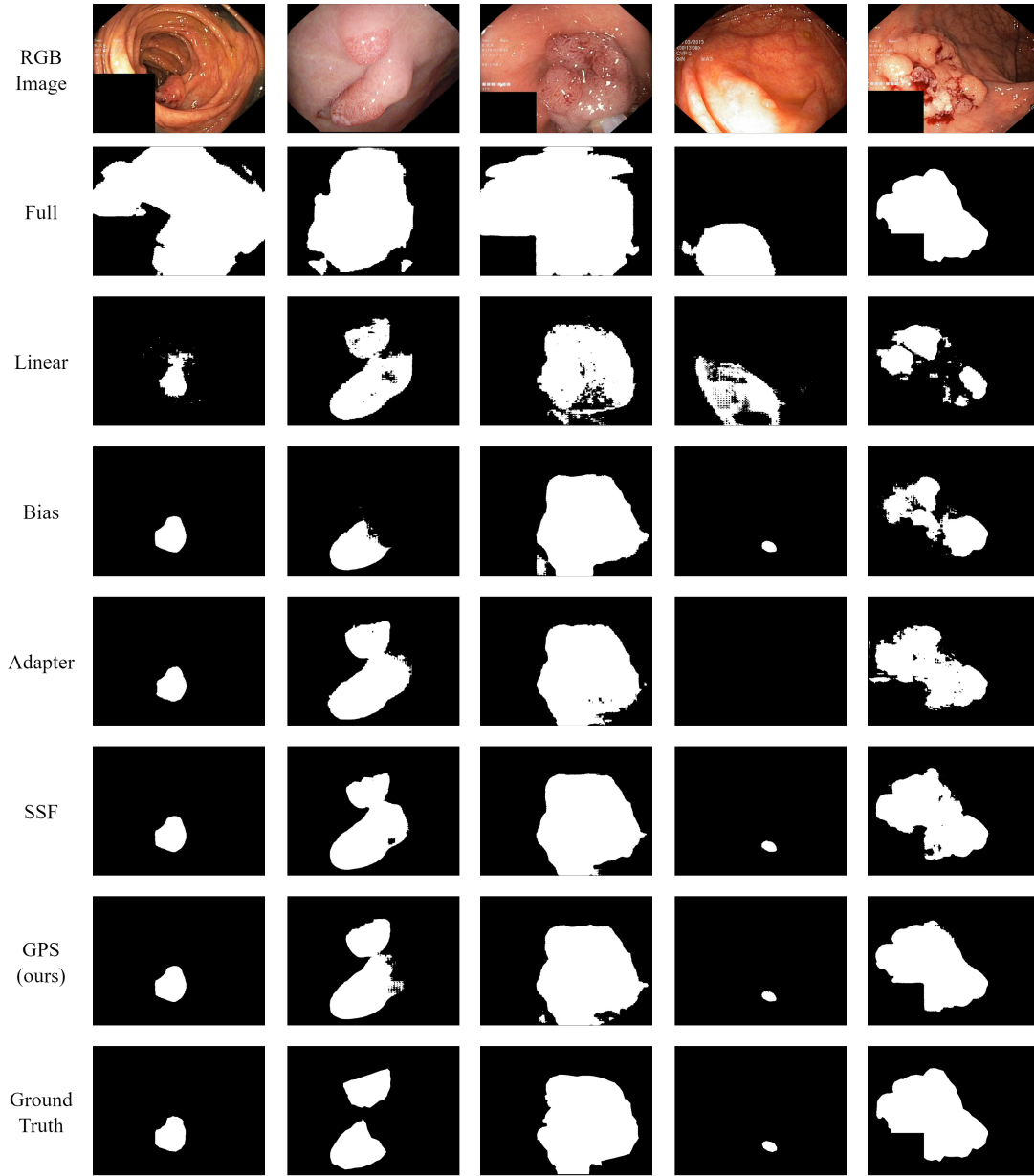


Figure 10: **The Visualization Result of Polyp Segmentation.** Here GT means ground truth. As illustrated in this figure, although SAM and other methods can identify some polyp structures in the image, the result is not accurate. By using GPS, our approach elevates the performance with SAM.

distribution of downstream tasks. This is due to our selection scheme, which makes the parameters evenly distribute on the whole network and thus the parameters from shallow layers tend to share parameters as similar findings from the field of multi-task learning (Karimi Mahabadi et al., 2021; Pfeiffer et al., 2020a; Sung et al., 2022).

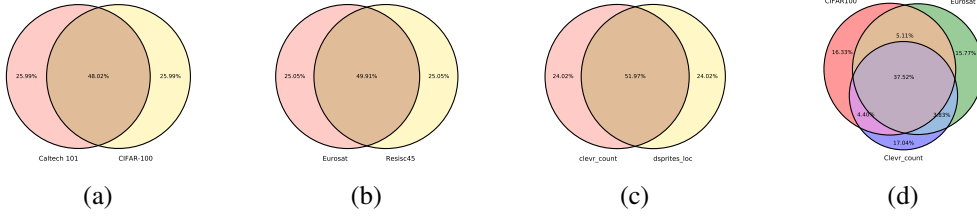


Figure 11: The overlap of the selected parameters across different tasks. Overlap is determined based on parameter position. If selected parameters share the same position in the network, they are considered to have overlap. We test on the ViT-B/16 with following tasks: (a) Cifar100 and Caltech101; (b) Eurosat and Resisc45; (c) Clevr/count and Dsprites/loc; (d) Cifar100, Eurosat and Clevr/count.

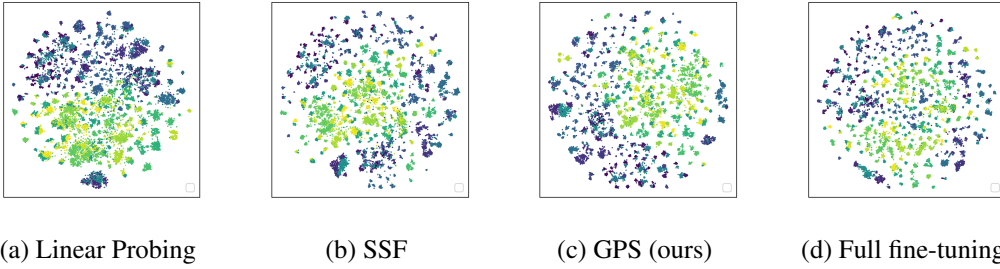


Figure 12: t-SNE visualization of different fine-tuning methods, including linear probing, SSF, GPS, full fine-tuning.

Feature distribution

On the NABirds datasets, we use t-SNE to visualize the feature distribution of different fine-tuning methods. The results of all comparison methods are obtained based on the ViT-B/16 pre-trained on ImageNet-21k. The visualization results are illustrated in Figure 12. Feature clustering results using our GPS are superior to those with linear probing, SSF, and full fine-tuning.

B.4 Details of the evaluation datasets

The statistic of all datasets used in this paper is shown in Table 11.

Image classification

FGVC Fine-Grained Visual Classification (FGVC) benchmark includes 5 downstream tasks, which are CUB-200-2011 (Wah et al.), NABirds (Van Horn et al., 2015), Oxford Flowers (Nilsback and Zisserman, 2008), Stanford Dogs (Khosla et al., 2011) and Stanford Cars (Gebru et al., 2017). Each one contains more than 100 classes and a few

Dataset	Description	#Classes	Train	Val	Test
Fine-Grained Visual Classification (FGVC)					
CUB-200-2011 (Wah et al.)	Bird recognition	200	5,394	600	5,794
NABirds (Van Horn et al., 2015)	Bird recognition	555	21,536	2,393	24,633
Oxford Flowers (Nilsback and Zisserman, 2008)	Flower recognition	102	1,020	1,020	6,149
Stanford Dogs (Khosla et al., 2011)	Dog recognition	120	10,800	1,200	8,580
Stanford Cars (Gebru et al., 2017)	Car classification	196	7,329	815	8,041
Visual Task Adaptation Benchmark (VTAB-1k) (Zhai et al., 2019)					
CIFAR-100 (Krizhevsky et al., 2009)	Natural	100	800/1000	200	10,000
Caltech101 (Fei-Fei et al., 2006)		102			6,084
DTD (Cimpoi et al., 2014)		47			1,880
Flowers102 (Nilsback and Zisserman, 2008)		102			6,149
Pets (Parkhi et al., 2012)		37			3,669
SVHN (Netzer et al., 2011)		10			26,032
Sun397 (Xiao et al., 2010)		397			21,750
Patch Camelyon (Veeling et al., 2018)	Specialized	2	800/1000	200	32,768
EuroSAT (Helber et al., 2019)		10			5,400
Resisc45 (Cheng et al., 2017)		45			6,300
Retinopathy (Graham, 2015)		5			42,670
Clevr/count (Johnson et al., 2017)	Structured	8	800/1000	200	15,000
Clevr/distance (Johnson et al., 2017)		6			15,000
DMLab (Beattie et al., 2016)		6			22,735
KITTI/distance (Geiger et al., 2013)		4			711
dSprites/location (Matthey et al., 2017)		16			73,728
dSprites/orientation (Matthey et al., 2017)		16			73,728
SmallNORB/azimuth (LeCun et al., 2004)		18			12,150
SmallNORB/elevation (LeCun et al., 2004)		9			12,150
General Image Classification Datasets					
CIFAR-100 (Krizhevsky et al., 2009)	General images	100	50,000	-	10,000
ImageNet-1K (Deng et al., 2009)		1,000	1,281,167	50,000	150,000
Robustness and Out-of-Distribution Datasets					
ImageNet-A (Hendrycks et al., 2021b)	Robustness & OOD	200	-	-	7,500
ImageNet-R (Hendrycks et al., 2021a)		200	-	-	30,000
ImageNet-C (Hendrycks and Dietterich, 2019)		1,000	-	-	75*50,000
Cross-domain Semantic Segmentation Dataset					
Kvasir-SEG (Jha et al., 2020a)	Polyp Segmentation	2	880	-	120

Table 11: Detailed statistics of the datasets evaluated on our work. We follow the VPT (Jia et al., 2022a) for train/val split. This table is partially borrowed from VPT (Jia et al., 2022a) and SSF (Lian et al., 2022).

thousand images. We directly use the public splits if one contains, otherwise, we follow the splits in (Jia et al., 2022a).

VTAB-1k Visual Task Adaptation Benchmark (Zhai et al., 2019) contains 19 visual classification tasks which are grouped into 3 sets: (1) Natural – tasks with natural images captured by standard cameras; (2) Specialized – tasks with images captured via specialized equipment, e.g., medical camera or satellite sensor; (3) Structured – tasks with images synthesized from simulated environments, which require geometric

comprehension like object counting and depth estimation. Each one contains only 1000 training examples while a large number of test images (i.e. over 20,000 on average).

CIFAR-100 CIFAR-100 (Krizhevsky et al., 2009) is a widely used general image classification task. It contains 50,000 training and 10,000 test images with 100 categories.

ImageNet-1K ImageNet-1K (Deng et al., 2009) is the most commonly utilized subset of ImageNet for object classification, encompassing 1000 classes and featuring a training set of 1,281,167 images, a validation set of 50,000 images, and a test set of 100,000 images.

Semantic segmentation

Polyp segmentation We select kvasir-SEG (Jha et al., 2020b) for polyp segmentation task. We follow the settings in Medico automatic polyp segmentation‘ task at mediaeval 2020 (Jha et al., 2020a) with a train-valid ratio of 880:120.

Robustness and OOD

ImageNet-A ImageNet-A (Hendrycks et al., 2021b) contains 200 classes, which is selected from ImageNet-1K (1000 classes). All samples are real-world adversarial samples that caused the ResNet model to produce erroneous classifications.

ImageNet-R ImageNet-R (Hendrycks et al., 2021a) contains art, graffiti, sculptures, tattoos, toys, cartoons, paintings, embroidery, deviantart, graphics, patterns, plastic objects, origami, plush objects, sketches, and video game renditions from ImageNet classes.

ImageNet-C ImageNet-C (Hendrycks and Dietterich, 2019) is an open-source collection of algorithmically generated corruptions, such as blur and noise, that have been applied to the ImageNet test set.

B.5 Extended related work

Visual parameter efficient fine-tuning

In the field of computer vision, current work endeavors to pre-train larger models (Devlin et al., 2018; Dosovitskiy et al., 2020b; Liu et al., 2021d; Radford et al., 2021b; Zhou et al., 2021, 2022) on extensive datasets, followed by fine-tuning diverse downstream tasks to achieve superior performance and faster convergence. Conventional arts set all the network parameters learnable and adapt them to the target tasks. However, as foundation models become increasingly large and the number of downstream tasks increases, it becomes impractical due to the significant computational and storage requirements that

it entails. Parameter-efficient fine-tuning (PEFT) methods are proposed to alleviate such a burden, which tunes only a tiny portion of the parameters. The general PEFT can be categorized into addition-based and selection-based methods.

Addition-based methods introduce additional parameters to the pre-trained backbone. Adapter methods keep most of the parameters in the model frozen and update only small-scale injected parameters. Bottleneck-structured adapters (Bapna et al., 2019; Houlsby et al., 2019; Pfeiffer et al., 2020a,b; Rebuffi et al., 2017; Rücklé et al., 2020; Stickland and Murray, 2019; Sung et al., 2022; Wang et al., 2020c; Zhang et al., 2021b) adopt a residual pathway to leverage both original and task-specific knowledge by learning down-projection and up-projection with a nonlinear activation. Others (Mahabadi et al., 2021) propose a hyper-network to generate model weights or decompose the dense weighted matrix into the low-rank matrix to reduce parameters (Karimi Mahabadi et al., 2021). Instead of introducing extra modules, prompt methods (Ding et al., 2021; Gao et al., 2020; Hu et al., 2021b; Ju et al., 2022; Li and Liang, 2021; Liu et al., 2023d, 2022b) wrap the input with context. A representative work VPT (Jia et al., 2022a) prepend learnable prompts to the input tokens before feeding it into each Transformer block. VPT includes two variants VPT-Shallow and VPT-Deep associated with the number of inserted layers. VPT-Shallow simply prepends prompts to the first transformer layer while VPT-Deep prepends prompts to all the layers. However, it’s inflexible when applying the method to new tasks since it relies on hand-crafted prompt length selection. Apart from the adapter and prompt tuning, a recent study SSF (Lian et al., 2022) introduces two learnable vectors to scale and shift the feature map in each transformer operation and achieves promising results. These extra parameters will lead to a substantial increase in computational overhead and hinder the rate of convergence. Our method solves these issues without adding parameters or changing the network topology so it can effectively alleviate such problems.

Selection-based methods (Guo et al., 2020a; Zaken et al., 2021; Zhao et al., 2020) do not introduce any new parameters but directly select part of the parameters to be optimized without modifying the intrinsic architecture of the model. Bitfit (Zaken et al., 2021) only fine-tunes bias vectors in the pre-trained model. Other methods only fine-tune the top-K layers (Houlsby et al., 2019) or the last linear layer (Jia et al., 2022a) with other layers freeze. Despite efficiency, they suffer a significant accuracy drop compared to the full fine-tuning since the manually specified parameters tend to be a non-optimal solution. Our gradient-based parameter selection method falls into this category. Since the gradient can serve as a tool for determining parameter significance, our method is intuitive but surprisingly effective.

Subset network training

Standard pruning technique (Gale et al., 2019; Han et al., 2015a,b; Kruschke and Movellan, 1991; Li et al., 2016a; Wen et al., 2016) naturally uncovers subnetworks whose initializations made them capable of training effectively. The lottery ticket hypothesis (Frankle and Carbin, 2018) articulate that subnetworks can reach test accuracy

comparable to the original network. Drawing from the theory, fine-tuning methods based on subset network are widely studied. SpotTune (Guo et al., 2019) designs a policy network to make routing decisions for subset networks. Child-tuning (Xu et al., 2021) iteratively updates a subset of parameters by masking out the gradients of the non-child network during the backward process. However, the computing overhead led by hyper networks or iterative parameter selection makes none of these methods parameter-efficient. We fix the position of parameters that will be updated by simple gradient weights sorting before training which makes our method parameter efficient.

B.6 Discussion

Why sub-network? There is a lot of research in the field of neural network pruning, where researchers aim to identify the importance of the parameters in a network and eliminate some unnecessary parameters without performance deterioration (about 90% parameters of the model are pruned) (Chen et al., 2020a; Cun et al., 1990; Li et al., 2016b; Prasanna et al., 2020). Motivated by this, we posit the existence of a sub-network containing crucial parameters that can be fine-tuned for optimal performance on downstream tasks.

Magnitude or gradient? In contrast to the approaches of identifying the importance of the parameters in (Chen et al., 2020a; Cun et al., 1990; Li et al., 2016b; Prasanna et al., 2020), which rely on weight magnitude to determine parameter importance, our method identifies parameter importance based on gradient values. An important difference between gradient and magnitude is that the gradient-based method is task-specific, as the gradient is calculated by the backpropagation of the loss for a specific task, while the magnitude-based method use a set of same parameters for all downstream tasks. However, our ablation study in the main body has shown gradient-based method perform better and the Figure 11 also show that each task has its own task-specific parameters.

B.7 Limitations and societal impacts

Limitations Several studies (Karimi Mahabadi et al., 2021; Pfeiffer et al., 2020a; Sung et al., 2022) have demonstrated that certain similar tasks can be optimized together through parameter sharing, resulting in improved performance across all individual tasks. However, our work focuses on selecting distinct parameters for various tasks. Although we already tune affordable parameters, we do not fully exploit the potential of parameter sharing across different tasks. Therefore, we posit that our work can be extended to a multitask setting, where tasks share tuning parameters and thus further reduce the total number of learnable parameters.

Societal impacts Our method can effectively fine-tune pre-trained models for downstream tasks by adjusting less than 1% of the network’s parameters. This is particularly

beneficial when dealing with large pre-trained models and multiple downstream tasks, as it saves computational resources, memory costs, and reduces carbon emissions. Our approach maintains the model’s original structure without introducing any additional parameters during both the training and inference stages, distinguishing it from other methods. However, similar to other fine-tuning approaches, our method relies on a pre-trained model. If this upstream pre-trained model is trained on illicit data, it may also violate the use of fine-tuning methods.

C Appendix to Chapter 5

C.1 Datasets

Our experiments primarily target both natural language understanding (NLU) and natural language generation (NLG) tasks. Specifically, we evaluate commonsense reasoning and arithmetic reasoning for NLG task and GLUE for NLU task.

Commonsense Reasoning

We evaluate the our method on eight widely-used datasets that span various forms of open-ended question answering:

- **BoolQ**(Clark et al., 2019a): a yes/no question-answering dataset composed of naturally occurring questions. Following prior work, we remove the associated passages to focus solely on the question context.
- **PIQA**(Bisk et al., 2020): designed to test physical commonsense, this dataset requires selecting the most plausible action in a given hypothetical situation.
- **SIQA**(Sap et al., 2019): targets social commonsense by asking models to reason about human actions and their social consequences.
- **HellaSwag**(Zellers et al., 2019b): involves selecting the most coherent sentence completion given a narrative context, emphasizing grounded commonsense inference.
- **WinoGrande**(Sakaguchi et al., 2021), inspired by the Winograd Schema Challenge(Levesque et al., 2012), tests pronoun resolution in context, requiring fine-grained commonsense reasoning.
- **ARC-Easy (ARC-e)**(Clark et al., 2018): a benchmark of multiple-choice elementary science questions with relatively straightforward reasoning.
- **ARC-Challenge (ARC-c)**(Clark et al., 2018): a more difficult subset of ARC designed to be resistant to simple co-occurrence-based solutions.
- **OpenBookQA (OBQA)** (Mihaylov et al., 2018): a knowledge-intensive QA dataset requiring multi-hop reasoning across both textual context and external knowledge.

We adopt the same experimental protocol as described in Hu et al. (2023a), aggregating the training sets of the above datasets into a unified corpus referred to as COMMONSENSE170K. Fine-tuning is conducted on this joint dataset, and evaluation is performed on the individual test sets of each benchmark. Detailed dataset statistics and simplified training examples are also available in Hu et al. (2023a).

Arithmetic reasoning

We train and evaluate our method using seven benchmark datasets that span a diverse range of mathematical word problem domains:

- **AddSub**(Hosseini et al., 2014), a dataset composed of elementary arithmetic problems involving addition and subtraction.
- **AQuA**(Ling et al., 2017), which presents algebraic word problems in a multiple-choice format.
- **GSM8K**(Cobbe et al., 2021), consisting of grade-school math problems that require multi-step reasoning.
- **MAWPS**(Koncel-Kedziorski et al., 2016), a collection of math word problems with varied linguistic and arithmetic complexity.
- **MultiArith**(Roy and Roth, 2015), featuring problems that demand reasoning through multiple arithmetic steps.
- **SingleEq**(Koncel-Kedziorski et al., 2015), which includes math problems that can be solved by formulating a single equation of varying complexity.
- **SVAMP** (Patel et al., 2021), designed to test a model’s robustness to structural variations in math problem formulations by rephrasing original problems in a challenging way.
- **MAWPS** (Koncel-Kedziorski et al., 2016) is a collection of math word problems of varying complexity, involving basic arithmetic operations such as addition, subtraction, multiplication, and division. Each instance is annotated with both the natural language problem and its corresponding symbolic equation, facilitating studies in semantic parsing and numerical reasoning.

We follow the experimental design of Hu et al. (2023b), which also provides dataset statistics and representative examples for each benchmark. Our training is conducted on a unified dataset ‘‘MATH10K’’ which comprises training samples from four datasets: GSM8K, MAWPS, MAWPS-single, and AQuA. Following (Wu et al., 2024c), we compare our method with LoReFT and DiReFT on four tasks: GSM8K, MAWPS, SVAMP and AQuA and with other baselines on all above tasks except for MAWPS Hu et al. (2023b).

Natural language understanding

Following the evaluation protocol established by Wu et al. (2024b), we ensure a fair assessment on the GLUE validation set by partitioning it into two subsets. One subset, determined using a fixed random seed, is reserved for in-training evaluation, while the

other is used exclusively for final testing. After each training epoch, we evaluate the model on the in-training subset and select the checkpoint with the best performance across all epochs for testing. For datasets with relatively large validation sets (i.e., QQP, MNLI, and QNLI), we randomly sample 1,000 instances for in-training evaluation. For smaller datasets, we use 50% of the validation data for this purpose. As for the evaluation metrics, we adopt the Matthews correlation coefficient for CoLA, the Pearson correlation coefficient for STS-B, and classification accuracy for the remaining tasks. For MNLI, we report results on the matched subset only.

C.2 Results on natural language understanding tasks

Table 12: Performance comparison with existing PEFT methods on RoBERTa-base for the GLUE benchmark. *Most baseline results are taken from Wu et al. (2024b). The result is presented as the mean with standard deviation (SD) over five runs with different random seeds. †Results are from Wu et al. (2024c) as it shares the same experimental setting with Wu et al. (2024b). For a fair comparison, our NeuroAda also follows the same setting.

Model	PEFT	Params (%)	Accuracy (†) (SD)								
			MNLI	SST-2	MRPC	CoLA	QNLI	QQP	RTE	STS-B	Avg.
RoBERTa	FT	100%	87.3 _(0.34)	94.4 _(0.96)	87.9 _(0.91)	62.4 _(3.29)	92.5 _(0.22)	91.7 _(0.19)	78.3 _(3.20)	90.6 _(0.59)	85.6
	Adapter*	0.318%	87.0 _(0.28)	93.3 _(0.40)	88.4 _(1.54)	60.9 _(3.09)	92.5 _(0.02)	90.5 _(0.08)	76.5 _(2.26)	90.5 _(0.35)	85.0
	LoRA*	0.239%	86.6 _(0.23)	93.9 _(0.49)	88.7 _(0.76)	59.7 _(4.36)	92.6 _(0.10)	90.4 _(0.08)	75.3 _(2.79)	90.3 _(0.54)	84.7
	Adapter ^{FNN*}	0.239%	87.1 _(0.10)	93.0 _(0.05)	88.8 _(1.38)	58.5 _(1.69)	92.0 _(0.28)	90.2 _(0.07)	77.7 _(1.93)	90.4 _(0.31)	84.7
	NeuroAda	0.2674%	87.7 _(0.12)	94.3 _(0.10)	89.7 _(0.13)	56.1 _(0.09)	92.0 _(0.18)	90.2 _(0.23)	82.0 _(0.08)	91.0 _(0.23)	85.4
	Base										
	BitFit*	0.080%	84.7 _(0.08)	94.0 _(0.87)	88.1 _(1.57)	54.0 _(3.07)	91.0 _(0.05)	87.3 _(0.02)	69.8 _(1.51)	89.5 _(0.35)	82.3
	RED*	0.016%	83.9 _(0.14)	93.9 _(0.31)	89.2 _(0.98)	61.0 _(2.96)	90.7 _(0.35)	87.2 _(0.17)	78.0 _(2.06)	90.4 _(0.32)	84.3
	DiReFT [†]	0.015%	82.5 _(0.22)	92.6 _(0.76)	88.3 _(1.23)	58.6 _(1.99)	91.3 _(0.19)	86.4 _(0.27)	76.4 _(1.48)	89.3 _(0.56)	83.2
	LoReFT [†]	0.015%	83.1 _(0.26)	93.4 _(0.64)	89.2 _(2.62)	60.4 _(2.60)	91.2 _(0.25)	87.4 _(0.23)	79.0 _(2.76)	90.0 _(0.29)	84.2
	NeuroAda	0.0297%	85.1 _(0.09)	94.3 _(0.16)	90.7 _(0.07)	59.8 _(0.12)	92.1 _(0.12)	88.1 _(0.24)	79.1 _(0.23)	90.7 _(0.10)	85.0

we evaluate our NeuroAda on the GLUE for natural language understanding tasks. The results are shown in table 12.

C.3 Results of commonsense and math reasoning on LLaMA3-8B

The results of commonsense reasoning and math reasoning are shown in Table 13

C.4 Hyperparameters

Hyperparameter tuning and decoding strategy

Arithmetic reasoning Following Wu et al. (2024c), we adopt their training and validation splits of the GSM8K dataset. Models are trained on the training set, and

Table 13: Performance comparison with existing PEFT methods on eight commonsense reasoning datasets and seven arithmetic reasoning datasets on model LLaMA3-8B. *Most baseline results are taken from Hu et al. (2023a). †Results are from Wu et al. (2024c), ‡ results are taken from He et al. (2024) and *results are from Liu et al. (2024c), as they share the same experimental setting with Hu et al. (2023a). For a fair comparison, our NeuroAda is also trained for 3 epochs to align with these baselines. +When 3-epoch results are not available in the original paper, we re-trained the baselines using the official code and their reported best hyperparameters. All results for our method are averaged over three runs with different random seeds. Our method selects the top-20 and top-1 input connections per neuron for the high-budget and low-budget parameter groups, respectively.

Model	PEFT	Params (%)	Accuracy (↑)								
			Commonsense Reasoning								
			BoolQ	PIQA	SIQA	HellaS.	WinoG.	ARC-e	ARC-c	OBQA	Avg.
Llama3 (8B)	LoRA*	0.700%	70.8	85.2	79.9	91.7	84.3	84.2	71.2	79.0	80.8
	DoRA _{half} *	0.361%	74.5	88.8	80.3	95.5	84.7	90.1	79.1	87.2	85.0
	DoRA*	0.710%	74.6	89.3	79.9	95.5	85.6	90.5	80.4	85.8	85.2
	SMT‡	0.710%	75.7	88.4	81.4	96.2	88.2	92.7	83.2	88.6	86.8
	NeuroAda	0.343%	75.0	89.3	83.0	96.5	89.2	93.0	82.9	89.6	87.3
	DiReFT+	0.026%	73.0	89.8	81.4	96.1	87.8	92.3	79.9	85.4	85.7
	LoReFT+	0.026%	72.9	89.1	81.7	96.1	88.0	92.0	80.1	85.0	85.6
	NeuroAda	0.017%	71.7	84.9	81.4	93.9	85.4	88.8	77.0	83.8	83.4
			Arithmetic Reasoning								
			MulAri	GSM8K	AddSub	AQuA	SinEq	SVAMP	MAWPS	Avg.	
LLaMA3 (8B)	DiReFT+	0.026%	—	57.2	—	30.1	—	68.6	87.8	60.9	
	LoReFT+	0.026%	—	56.9	—	24.8	—	70.9	88.2	60.2	
	NeuroAda	0.017%	—	63.7	—	26.4	—	75.0	88.7	63.5	

hyperparameters are selected based on performance on the validation set. All hyperparameters are tuned using LLaMA-7B, and the resulting configuration is directly applied to LLaMA-13B, LLaMA2-7B, and LLaMA3-8B without additional tuning. We use a maximum sequence length of 512 tokens during training and hyperparameter tuning, and generate up to 32 new tokens during inference. Table 14 summarizes the full hyperparameter search space.

Dataset MATH10K is annotated with language model-generated chain-of-thought reasoning steps. In the tasks, models are required to generate a chain of thought (Wei et al., 2022) before producing the final answer. The included tasks are AddSub (Hosseini et al., 2014), AQuA (Ling et al., 2017), GSM8K (Cobbe et al., 2021), MAWPS (Koncel-Kedziorski et al., 2016), MultiArith (Roy and Roth, 2015), SingleEq (Koncel-Kedziorski et al., 2015), and SVAMP (Patel et al., 2021). See Appendix C.1 for detailed descriptions of each task. For fair comparison, we follow both evaluation settings used in prior work. Specifically, Wu et al. (2024c) report results on GSM8K, AQuA, SVAMP, and MAWPS, while Hu et al. (2023b) evaluate on MultiArith, GSM8K, AddSub, AQuA, SingleEq, and SVAMP. We compare our method with theirs under both settings to

Table 14: Hyperparameter search space for the LLaMA-7B model using our NeuroAda on the validation set of the GSM8K. The best-performing settings are underlined for selecting top-1 and wavy underline for selecting top-20 parameters per neuron in the model. Greedy decoding (without sampling) is used throughout the hyperparameter tuning process.

Hyperparameters (LLaMA-7B on GSM8K)	
Optimizer	AdamW
LR	$\{6 \times 10^{-4}, 9 \times 10^{-4}, 1 \times 10^{-3}, 3 \times 10^{-3}, 6 \times 10^{-3}, \underline{9 \times 10^{-3}}, 1 \times 10^{-2}, 3 \times 10^{-2}\}$
Weight decay	$\{0\}$
LR scheduler	Linear
Batch size	$\{8, \underline{16}, 32\}$
Warmup ratio	$\{0.00, \underline{0.06}, 0.10\}$
Epochs	$\{3\}$
Top- k	$\{\underline{1}, 20\}$

ensure a comprehensive and fair evaluation.

Commonsense reasoning Motivated by Wu et al. (2024c), we perform hyperparameter tuning on COMMONSENSE15K, a subset of the full COMMONSENSE170K benchmark. Details of the search space are provided in Table 15. We divide COMMONSENSE15K into training and validation splits using a ratio 7:3. Hyperparameter tuning is conducted exclusively on LLaMA-7B, and the optimal configuration identified on the validation set is reused for all other models, including LLaMA-7B/13B, LLaMA2-7B, and LLaMA3-8B, during training on the full COMMONSENSE170K dataset.

For the commonsense reasoning benchmark, we adopt greedy decoding without sampling, as the task requires multi-token classification. In contrast, for the arithmetic reasoning benchmark, we follow the decoding setup used by Hu et al. (2023b), employing a higher decoding temperature of 0.3. This change is made to avoid instability issues caused by near-zero token probabilities, which can lead to decoding errors in the HuggingFace implementation when using beam search.

Dataset Our comparative experiments are primarily conducted on LLaMA-7B, using two lightweight reasoning datasets to enable rapid evaluation and comparison: COMMONSENSE15K for commonsense reasoning and GSM8K for arithmetic reasoning. COMMONSENSE15K is a subset of the larger COMMONSENSE170K dataset, originally partitioned by Hu et al. (2023b). (1) COMMONSENSE170K comprises eight commonsense reasoning tasks, including BoolQ (Clark et al., 2019a), PIQA (Bisk et al., 2020), SIQA (Sap et al., 2019), HellaSwag (Zellers et al., 2019b), WinoGrande (Sakaguchi et al., 2021), ARC-e, ARC-c (Clark et al., 2018), and OBQA (Mihaylov et al., 2018), as detailed in Appendix C.1. All examples are presented as multiple-choice questions,

Table 15: Hyperparameter search space for the LLaMA-7B model using our NeuroAda on the validation set of the COMMONSENSE15K. The best-performing settings are underlined for selecting top-1 and wavy underline for selecting top-20 parameters per neuron in the model. Greedy decoding (without sampling) is used throughout the hyperparameter tuning process.

Hyperparameters (LLaMA-7B on COMMONSENSE15K)	
Optimizer	AdamW
LR	$\{7 \times 10^{-4}, 9 \times 10^{-4}, 2 \times 10^{-3}, 4 \times 10^{-3}, 6 \times 10^{-3}, \underline{8 \times 10^{-3}}, 1 \times 10^{-2}, 2 \times 10^{-2}\}$
Weight decay	$\{0\}$
LR scheduler	Linear
Batch size	$\{\underline{8}, 16, 32\}$
Warmup ratio	$\{0.00, 0.06, \underline{0.10}\}$
Epochs	$\{3\}$
Top- k	$\{\underline{1}, 20\}$

where the model is required to directly output the correct option without generating intermediate rationales. We adopt the prompt template from [Hu et al. \(2023b\)](#), with an additional string normalization step (removal of leading and trailing whitespace). We split COMMONSENSE15K into training and validation sets using a 7:3 ratio for our experiments. (2) GSM8K([Cobbe et al., 2021](#)) dataset comprises grade-school-level arithmetic word problems that require multi-step reasoning to arrive at the correct answer. In contrast to COMMONSENSE15K, solving GSM8K typically involves generating a chain-of-thought ([Wei et al., 2022](#)) prior to producing the final answer. We adopt the same prompt template as used in [Hu et al. \(2023b\)](#).

Natural language understanding. Following [Wu et al. \(2024c\)](#), we perform hyperparameter tuning for each GLUE task individually using both RoBERTa-base. Hyperparameters are selected based on performance on a held-out validation set with a fixed random seed of 42. To obtain the final results, we evaluate the models using four additional unseen seeds: 43, 44, 45, 46. We adopt the evaluation protocol of [Wu et al. \(2024b\)](#). For QQP with RoBERTa-large, due to observed stochasticity across repeated runs with the same seed, we report the best result from three trials for each seed. As noted by [Wu et al. \(2024b\)](#), the evaluation results on RTE are unstable due to the dataset’s small size. We follow their approach and adjust the set of random seeds accordingly to ensure fair comparison. Additionally, we replace one or two random seeds for CoLA to improve evaluation stability.

Preliminary Analysis We compare the mask-based method“which applies binary masks to zero out the gradients of unselected (frozen) parameters“with the addition-based method, which introduces new trainable parameters to bypass the selected ones,

Table 16: Hyperparameter search space for the LLaMA-7B model using our NeuroAda on the validation set of the GSM8K and COMMONSENSE15K for different number of trainable parameters. Greedy decoding (without sampling) is used throughout the hyperparameter tuning process.

Hyperparameters on LLaMA-7B for different number of trainable parameters	
Optimizer	AdamW
LR	$\{6 \times 10^{-5}, 9 \times 10^{-5}, 5 \times 10^{-4}, 7 \times 10^{-4}, 9 \times 10^{-4}, 3 \times 10^{-3}, 6 \times 10^{-3}, 9 \times 10^{-3}, 1 \times 10^{-2}\}$
Weight decay	$\{0\}$
LR scheduler	Linear
Batch size	$\{16\}$
Warmup ratio	$\{0.06\}$
Epochs	$\{3\}$
Top- k	$\{1\ 5\ 10\ 20\ 50\ 100\ 300\ 500\}$

rather than tuning them directly, for the purpose of sparse fine-tuning. This comparison is conducted across three dimensions: effectiveness, GPU memory usage, and training efficiency. We further investigate the effectiveness of the proposed addition-based method, NeuroAda, which is designed to ensure that all neurons in the network retain the potential to update their activation states during fine-tuning. To this end, we analyze the proportion of neurons actively involved in the tuning process and evaluate various parameter selection strategies for determining which neurons are activated.

To ensure a fair comparison between the addition-based and mask-based approaches, as well as across different parameter selection strategies, we conduct a hyperparameter search over a range of learning rates for each experiment using the training set. The optimal configuration is selected based on performance on the validation set. This step is essential, as PEFT methods are generally sensitive to the choice of learning rate (Wu et al., 2024c). The complete hyperparameter search space is provided in Table 16.

C.5 Advantages of NeuroAda

NeuroAda significantly reduces the backward computation costs with four core advantages. (1) It achieves **mask-free sparsity** by statically selecting top- k weights per neuron and storing only a compact set of BF16 deltas and integer indices. This avoids the memory and compute overhead associated with dynamic binary masks commonly used in gradient-based sparse methods, as quantified in Table 5.1. (2) It requires **no warm-up or task-specific signal**: the magnitude-based selection operates entirely offline and consistently across tasks, eliminating the need for gradient accumulation or adaptive heuristics. (3) It ensures **neuron-level coverage** by allocating trainable updates to every row of the weight matrix. This guarantees that all neurons participate in learning and avoids the dead filter problem often observed in block- or layer-wise pruning. (4) It introduces **no inference-time overhead**: the sparse update tensor Δ is

merged into the original weights post-training, preserving the model’s structure, runtime efficiency, and compatibility with standard inference stacks.

C.6 Algorithm

NeuroAda follows a three-phase procedure designed for efficiency, simplicity, and compatibility with standard inference infrastructure. As shown in Algorithm 1, the process begins with an offline selection phase, where the top- k highest-magnitude weights are identified per neuron based on the pretrained weight matrix. This static selection removes the need for gradient-based importance scoring or dynamic masking during training.

During training, only the selected top- k coordinates per neuron are updated, with all other parameters kept frozen. This enables mask-free, neuron-wise sparse adaptation that significantly reduces gradient computation and optimizer state memory. Crucially, NeuroAda performs updates directly over a small number of stored deltas, requiring no structural changes or auxiliary layers.

After training, the sparse deltas are merged into the frozen weights via an in-place update, resulting in a model that retains its original structure and supports efficient, standard inference. This design ensures minimal runtime overhead while offering strong adaptation capabilities through fine-grained parameter selection.

Algorithm 1: NeuroAda: Sparse top- k neuron-wise adaptation with merge-compatible updates.

Input: pretrained weight matrix $\Phi \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$, top- k budget $k \ll d_{\text{in}}$, training mini-batches $\{(\mathbf{x}, \mathbf{y})\}$, learning-rate η

Output: adapted model with merged weights $\Phi + \Delta$

Phase 1: Offline Top- k Selection;

```

foreach  $i = 1, \dots, d_{\text{out}}$  do                                     // row  $\equiv$  neuron
| // store  $k$  index positions  $\mathcal{I}_i \leftarrow \text{TopK}(|\Phi_{i,:}|, k)$ 
end

```

Phase 2: Sparse Training (mask-free);

For each neuron i , initialize $\Delta_{i, \mathcal{I}_i} \leftarrow 0$;

```

foreach mini-batch  $(\mathbf{x}, \mathbf{y})$  do
| // forward pass;
|  $\mathbf{h} \leftarrow (\Phi + \Delta) \mathbf{x}$ ;
| compute loss  $\mathcal{L}(\mathbf{h}, \mathbf{y})$ ;
| // backward: update only selected entries;
| foreach  $i = 1, \dots, d_{\text{out}}$  do
| |  $g_i \leftarrow \nabla_{\Delta_{i, \mathcal{I}_i}} \mathcal{L}$ ;
| |  $\Delta_{i, \mathcal{I}_i} -= \eta g_i$ 
| end
end

```

Phase 3: One-shot Merge and Inference;

```

foreach  $i = 1, \dots, d_{\text{out}}$  do
|  $\Phi_{i, \mathcal{I}_i} \leftarrow \Phi_{i, \mathcal{I}_i} + \Delta_{i, \mathcal{I}_i}$ ;
end
Delete  $\Delta$  from memory.;
# PyTorch (illustrative)
y = F.linear(x, merged.weight, bias)

```

D Appendix to Chapter 6

In this supplemental material, we provide extra details about the content in the main body of the paper. First, we provide detailed proof for Equation (6.12) in Appendix D.1. Then, we discuss the limitations of our work in Appendix D.2. Moreover, the broader impacts of our research are discussed in Appendix D.3. In addition, we present all hyperparameters and experiment settings in Appendix D.6 for a better understanding of the experiments and reproduction to the readers. We also provide the extended related works in Appendix D.5. Finally, the additional experiment results are demonstrated in Appendix D.6, which further indicate the effectiveness of our proposed method and the consistency with the claim in the main body of the paper.

D.1 Proof for Equation

We provide the proof for Equation (6.12). According to Lagrangian duality, Equation (6.10) can be reformulated as:

$$\begin{aligned} L &= \min_{\Theta} \max_{\lambda} \mathcal{L}(\Theta) + \lambda \|(I - M)(\theta_{\text{sha}} - \theta_{\text{sha}}^{\text{in}})\|^2 \\ &\geq \max_{\lambda} \min_{\Theta} \mathcal{L}(\Theta) + \lambda \|(I - M)(\theta_{\text{sha}} - \theta_{\text{sha}}^{\text{in}})\|^2 \\ &\geq \min_{\Theta} \mathcal{L}(\Theta) + \|(I - M)(\theta_{\text{sha}} - \theta_{\text{sha}}^{\text{in}})\|^2 \end{aligned}$$

where λ is the Lagrangian multiplier.

D.2 Limitations

Due to the limited computational resources, we employ grid searches in the *Joint train* method to determine the optimal hyperparameter for the number of trainable parameters, which is then utilized across all gradient manipulation methods. However, it is possible that these methods may benefit from a more optimized hyperparameter selection for the number of trainable parameters. Furthermore, sparse training can effectively mitigate gradient conflicts between tasks in MTL by reducing the dimensionality of parameter space and limiting their impact on updates between tasks. The regularization constitutes one of the theory’s reasons. Nevertheless, we anticipate that our future research will contribute to a deeper comprehension of multi-task learning and subsequently enhance the performance of MTL.

D.3 Broader Impacts

The nature of our research does not directly contribute to societal impact; however, like any machine learning paper, it has the potential to adversely affect society through automation and job displacement. While it is challenging to predict specific risks, similar to any technology, inadequate regulation may lead to an exacerbation of social

Method	Pre-trained model				Random initialized model	
	SAM	Swin	ViT		MTAN	
	NYU-v2	CelebA	Clevr	SmallNORB	NYU-v2	CityScapes
Joint Train w/ ST	30.97	37.60	29.38	29.38	62.19	76.02
PCGrad w/ ST	30.97	37.60	29.38	19.63	62.19	76.02
CAGrad w/ ST	30.97	72.85	29.38	29.38	62.19	76.02
GradDrop w/ ST	30.97	49.58	29.38	29.38	62.19	76.02
MGDA w/ ST	30.97	37.60	29.38	29.38	62.19	62.19
IMTL-G w/ ST	30.97	37.60	29.38	29.38	62.19	62.19
NashMTL w/ ST	30.97	37.60	29.38	29.38	62.19	83.48
FAMO w/ ST	30.97	37.60	29.38	29.38	62.19	62.19

Table 17: Number of trainable parameters. The values in the table are expressed as percentages (%). As we select Top-K input parameters among all input connections for each neuron, therefore the same K might lead to different percentages of trainable parameters for different models. For example, K=300 results in 30.97% in SAM, 37.60% in Swin, and 29.38% in ViT for the pre-trained model.

and economic inequality. The positive aspect lies in the potential environmental impact of our work, as multi-task learning enables information sharing among tasks, thereby reducing data requirements and further minimizing energy consumption during training.

D.4 Detailed experiment setting

Number of trainable parameters

We provide the number of trainable parameters for all experiments conducted in our paper. As shown in Table 17, most of them have the same percentage of trainable parameters within a model across different methods. In addition, in general, we can observe that sparse training for the pre-trained model needs $\sim 30\%$ while that for random initialized model needs $\sim 60\%$.

Implementation details

Following the work of Nash (Navon et al., 2022), we apply all gradient manipulation techniques to the gradients of the shared weights. We set the hyperparameter c of CAGrad to 0.4, as it has been reported to yield optimal performance for NYUv2 and Cityscapes datasets (Liu et al., 2021a). The experiments were conducted on the A100 80G GPU. Typically, training with SAM using NYU-v2 and Swin with CelebA requires approximately 1 day for a gradient manipulation method. Training ViT with SmallNORB takes around 18 minutes for a gradient manipulation method, while training ViT with Clevr takes about 30 minutes. On the other hand, training MTAN with NYU-

v2 demands roughly 18 hours for a gradient manipulation method, whereas training MTAN with CityScapes necessitates approximately 12 hours.

SAM, ViT, Swin For all methods, including single-task learning, the gradient manipulation method, and our sparse training, we employed a batch size of 3 and searched for the optimal learning rate from the set $\{2e-4, 5e-5\}$, and then the best results are reported. The reason is that we find the optimal learning rate for sparse training is bigger than that for full parameters training. Therefore, for most methods, the optimal learning rate for sparse training is $2e-4$ and that for the full parameters training is $5e-5$. we also use data augmentations for all methods, following (Liu et al., 2021a). The batch size used is set to be 3 for NYUv2 dataset, and 256 for CelebA, and 128 for SmallNORB and Clevr.

MTAN Following the works in (Liu et al., 2021a; Navon et al., 2022), we incorporate data augmentations during training for both *Joint Train* method and all gradient manipulation methods. Each method is trained for 200 epochs with an initial learning rate of 0.0001, which is then reduced to 0.00005 after 100 epochs. For Multi-Task Learning (MTL) methods, we utilize a Multi-Task Attention Network (MTAN) (Liu et al., 2019b) based on SegNet architecture proposed by (Badrinarayanan et al., 2017). Similar to previous studies (Liu et al., 2021a), the STL baseline refers to training task-specific SegNet models. The batch size used is set to be 2 for NYUv2 dataset and 8 for CityScapes dataset respectively. To align with prior research on MTL including (Badrinarayanan et al., 2017; Liu et al., 2021a; Yu et al., 2020), we report the test performance averaged over the last 10 epochs.

D.5 Extended related work

Multi-task learning Multi-task learning (Zhang and Yang, 2021) aims to improve the overall performance of all tasks. In this work, we focus on a conventional setup of multi-task learning (Vandenhende et al., 2021): given a single input, multi-task models perform different and related predictions, such as segmentation, depth and surface normal. In other words, the input is shared by different tasks. In this paper, we roughly divide existing MTL into two categories:

i) *Multi-task optimization*. Recent works (Chen et al., 2020e; Liu et al., 2023a, 2021a,c; Navon et al., 2022; Sener and Koltun, 2018; Yu et al., 2020) provide impressive results in solving the task imbalance during optimization. The rationale behind these works is that re-weighting all task gradients or losses helps multi-task models reduce conflicting gradients among tasks (Liu et al., 2021a; Sener and Koltun, 2018). Specifically, some works (Chen et al., 2020e; Liu et al., 2021c; Sener and Koltun, 2018) propose to form a new update gradient at each optimization by linearly combining task gradients. Other works (Kendall et al., 2018; Liu et al., 2023a) learn dynamic loss scale to balance different tasks during training. However, it is challenging to scale up most existing optimization works to giant foundation models due to non-trivial

Methods	Segmentation		Depth		Surface Normal					$\Delta m\% \downarrow$
	mIoU \uparrow	Pix Acc \uparrow	Abs Err \downarrow	Rel Err \downarrow	Angle Distance \downarrow		Within $t^\circ \uparrow$			
					Mean	Median	11.25	22.5	30	
Random	59.85	80.09	0.3357	0.1359	22.17	16.12	36.08	64.50	75.56	6.014
Global	59.53	79.38	0.3380	0.1373	22.32	16.32	35.62	63.93	75.16	6.855
Reverse	59.35	79.57	0.3417	0.1396	22.31	16.25	35.98	64.07	75.16	6.960
Ours	60.03	79.96	0.3320	0.1353	21.98	15.92	36.69	64.92	75.82	5.314

Table 18: Different sparse training methods on SAM model with NYU-v2 datasets.

computational and memory costs. In this paper, we propose a neuron-based parameter selection to sparsely fine-tune the pre-trained model, which boosts the performance of most optimization methods.

ii) *Multi-task architecture* In this branch, multi-task methods design different architectures to improve the exchanging or sharing of information among tasks (Vandenhende et al., 2021). Regarding where tasks interact, multi-task architectures are separated into *encoder-focused* and *decoder-focused*. The former shares the information in the encoder by the transformation of activations among tasks (Misra et al., 2016), learnable task-specific attention modules (Liu et al., 2019b), branching networks for similar tasks (Guo et al., 2020b) and so on. The latter recursively uses task predictions to improve overall performance (Xu et al., 2018; Zhang et al., 2018b, 2019). However, these architectures still suffer from the task imbalance issue during multi-task optimization. In this paper, our work focuses on boosting multi-task optimization. As one of the multi-task optimization methods, our method can seamlessly generalize to different backbone models.

Training with subset of parameters several methods already proposed in single-task learning. several methods select a subset of parameters based on a certain pre-defined rule, such as gradient (Fu et al., 2023; Zhang et al., 2023d) and magnitude of parameters (Lagunas et al., 2021). In addition to selecting parameters by hand design, (Mostafa and Wang, 2019; Sanh et al., 2020; Xu et al., 2021) automatically select the subset of parameters through optimization. Although sparse training has been extensively investigated in single-task learning, its application in multi-task learning remains relatively unexplored. (Calandriello et al., 2014; Sun et al., 2020) learning to share information between tasks using a sparse model. Differing from them, in this paper, we systematically research the gradient conflict via the sparse training perspective.

D.6 Detailed experiment results

In this section, we provide the detailed experiment results conducted in the main body of our paper, including the average incident of gradient conflict, the incident of gradient conflict for all epochs, and visualization of the gradient conflict for *Joint Train* and all

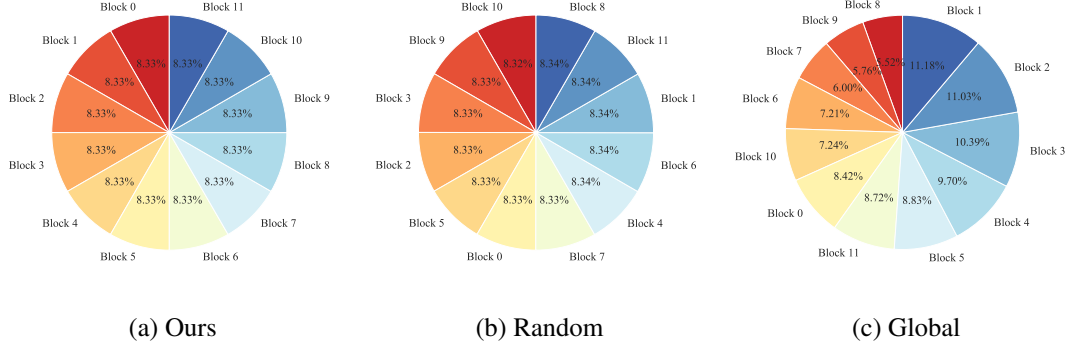


Figure 13: The distribution of selected trainable parameters for different sparse training methods over different blocks. The experiments are conducted on SAM model with NYU-v2 dataset.

gradient manipulation methods.

Ablation study

The detailed results for various sparse methods are provided in Table 18, which is the full version of Figure 6.5c. It can be observed that, with the exception of *Pix Acc* in segmentation, our sparse method outperforms other methods. In addition, we provide the distribution of the selected parameters using different sparse training over different blocks of the model. As shown in Figure 13, the parameters selected by our sparse training method and *Random* are evenly distributed over the whole network. As for *Global* selecting the parameters with the highest magnitude, the distribution of selected parameters is largely different over different blocks

NYU-v2 on SAM

The incidence of gradient conflict for *Joint Train* and gradient manipulation method over all epochs are shown in Figure 14, which is the full version of Figure 6.4 in the main body of the paper.

NYU-v2 on MTAN

We also conduct experiments on MTAN with NYU-v2 dataset. MTAN is a random initialized model. As we can see in Table 19, even for the random initialized model, sparse training can also reduce the incidence of gradient conflict. The visualization of the occurrence of gradient conflict for each epoch is shown in Figure 16 and the average incidence of gradient conflict across all epochs for different methods is shown in Figure 15. As for the performance of the overall tasks on NYU-v2, the sparse training improves not only the overall performance ($\Delta m\%$) but also the performance of each task

Methods	Average incidence of GC (%)	
	All epochs	Last 50% epochs
Joint Train	36.01	39.87
w/ ST	33.86 (2.15)	36.45 (3.42)
PCGrad	35.71	39.51
w/ ST	34.05 (1.66)	37.25 (2.26)
CAGrad	37.21	40.93
w/ ST	34.14 (3.07)	37.04 (3.89)
GradDrop	36.37	39.71
w/ ST	34.42 (1.95)	37.10(2.61)
MGDA	37.76	42.1
w/ ST	37.15 (0.61)	41.25 (0.85)
IMTL-G	37.14	41.22
w/ ST	35.81 (1.33)	39.17 (2.05)
NashMTL	37.19	40.79
w/ ST	35.83 (1.36)	39.0 (1.79)

Table 19: Average incidence of gradient conflict between tasks over epochs for different methods. The improvement by sparse training is provided in (•). We calculate the average incidence of gradient conflict over all epochs and the last 50% epochs during training MTAN on NYUv2.

for all methods including *Joint Train* and all gradient manipulation methods, as shown in Table 20. In addition, following (Navon et al., 2022), we conduct the experiments three times with three different seeds. The *mean* ‘ *std* is presented in Table 20, we can observe that the sparse training is robust to the random seed.

NYU-v2 on Swin

In order to investigate how the incidence of gradient conflict changes with varying model sizes, we conduct experiments on Swin/Tiny, Swin/Base and Swin/Large through the *Joint Train*. As depicted in Table 21, there is an observed increase in the incidence of gradient conflict as the model size increases. Additionally, the performance of tasks improves as the model size increases Table 22.

CelebA on Swin

Following (Navon et al., 2022), we train CelebA on Swin for only 30 epochs, because there are many more tasks in this dataset compared with other datasets, which leads to a

Methods	Segmentation		Depth		Surface Normal					$\Delta m\%$ ↓
	mIoU ↑	Pix Acc ↑	Abs Err ↓	Rel Err ↓	Angle Distance ↓		Within t° ↑			
					Mean	Median	11.25	22.5	30	
STL	38.30	63.76	0.6754	0.2780	25.01	19.21	30.14	57.20	69.15	—
Joint Train	39.29	65.33	0.5493	0.2263	28.15	23.96	22.09	47.50	61.08	5.59
w/ ST	41.04 (*0.28)	66.05 (*0.12)	0.5417 (*0.0008)	0.2232 (*0.0011)	27.40 (*0.05)	22.90 (*0.12)	23.58 (*0.13)	49.59 (*0.14)	63.01 (*0.09)	2.49 (*0.11)
PCGrad	38.06	64.64	0.5550	0.2325	27.41	22.80	23.86	49.83	63.14	3.97
w/ ST	40.49 (*0.32)	66.17 (*0.23)	0.5441 (*0.0023)	0.2264 (*0.0030)	27.09 (*0.08)	22.55 (*0.03)	24.22 (*0.12)	50.34 (*0.17)	63.63 (*0.12)	1.98 (*0.12)
CAGrad	39.79	65.49	0.5486	0.2250	26.31	21.58	25.61	52.36	65.58	0.20
w/ ST	39.93 (*0.33)	66.19 (*0.16)	0.5299 (*0.0025)	0.2097 (*0.0038)	25.71 (*0.02)	20.70 (*0.03)	26.86 (*0.13)	54.22 (*0.15)	67.30 (*0.13)	-2.76 (*0.10)
GradDrop	39.39	65.12	0.5455	0.2279	27.48	22.96	23.38	49.44	62.87	3.58
w/ ST	40.84 (*0.35)	66.84 (*0.24)	0.5288 (*0.0021)	0.2209 (*0.0021)	27.18 (*0.03)	22.56 (*0.07)	24.10 (*0.11)	50.33 (*0.14)	63.67 (*0.13)	1.38 (*0.12)
MGDA	30.47	59.90	0.6070	0.2555	24.88	19.45	29.18	56.88	69.36	1.38
w/ ST	32.42 (*0.41)	61.61 (*0.21)	0.5851 (*0.0015)	0.2239 (*0.0032)	24.35 (*0.02)	18.61 (*0.03)	31.14 (*0.12)	58.63 (*0.15)	70.62 (*0.13)	-3.09 (*0.14)
IMTL-G	39.35	65.60	0.5426	0.2256	26.02	21.19	26.20	53.13	66.24	-0.76
w/ ST	40.73 (*0.33)	66.00 (*0.17)	0.5219 (*0.0015)	0.2100 (*0.0021)	25.6 (*0.05)	20.64 (*0.04)	26.81 (*0.16)	54.38 (*0.15)	67.49 (*0.12)	-3.18 (*0.11)
NashMTL	40.13	65.93	0.5261	0.2171	25.26	20.08	28.40	55.47	68.15	-4.04
w/ ST	39.75 (*0.21)	66.45 (*0.05)	0.5156 (*0.0006)	0.2121 (*0.0009)	24.96 (*0.01)	19.80 (*0.05)	28.80 (*0.11)	56.20 (*0.10)	68.93 (*0.09)	-5.11 (*0.07)

Table 20: The test performance on NYU-v2 dataset training on MTAN model, involving three tasks: semantic segmentation, depth estimation and surface normal. The result is the mean over three random seeds (*std* is presented in ‘•’). The green cell color indicates that sparse training improves the performance of joint training or gradient manipulation methods. The best result is highlighted in bold.

Model / Size	Average incidence of GC (%)
Swin / Tiny	37.42
Swin / Base	40.34
Swin / Large	41.84

Table 21: The average incidence of gradient conflict across all epochs during joint training with NYU-v2 on different sizes of Swin transformer.

significant increase in computation. As we can observe in Table 23, most of the methods including *Joint Train* and gradient manipulation methods can be improved by sparse training in terms of average incidence of gradient conflict between tasks over epochs. It is noted that the improvement by sparse training here is not significant, which is because of the limited training epoch. Specifically, as shown in Table 6.1 and Table 19, our sparse training improves more for later epochs. As for the performance of CelebA on Swin, please refer to Table 6.3. The visualization for the occurrence of gradient conflict for each epoch and average incidence of gradient conflict over all epochs for different methods, including *Joint Train* and all gradient manipulation methods, are shown in Figure 17 and Figure 18

SmallNORB on ViT

SmallNORB is a much more difficult benchmark compared to other benchmarks in this paper. It comprises artificial objects observed under varying conditions and includes two tasks: object azimuth and camera-elevation prediction. As shown in Table 24, even

Model	Segmentation		Depth		Surface Normal				
	mIoU \uparrow	Pix Acc \uparrow	Abs Err \downarrow	Rel Err \downarrow	Angle Distance \downarrow		Within t° \uparrow		
					Mean	Median	11.25	22.5	30
Swin/Tiny	55.22	76.54	0.3746	0.1542	27.47	21.70	27.81	52.40	64.05
Swin/Base	59.60	79.16	0.3419	0.1388	25.88	19.74	31.23	56.24	67.32
Swin/Large	61.34	80.28	0.3321	0.1345	25.09	18.73	33.05	58.12	68.86

Table 22: The test performance on NYU-v2 dataset jointly training on Swin models.

for the *STL*, the Top 1 accuracy only achieves $\sim 30\%$, therefore, we use Top 5 as an extra metric here. We observed that even for this difficult task, sparse training can still achieve better performance compared with *Joint Train* and all gradient manipulation methods.

CityScapes on MTAN

We also conduct experiments on MTAN with CityScapes dataset. MTAN is a random initialized model. As we can see in Table 26, even for the random initialized model, sparse training can also reduce the incidence of gradient conflict. The reduction in the incidence of gradient conflict for CityScapes is observed to be comparatively smaller than that for NYU-v2. This discrepancy can be attributed to the fact that CityScapes, which involves only two tasks, has a lower likelihood of encountering gradient conflicts between tasks compared to NYU-v2, which encompasses three tasks. The visualization of the occurrence of gradient conflict for each epoch is shown in Figure 19 and the average incidence of gradient conflict across all epochs for different methods is shown in Figure 20. As for the performance of the overall tasks on CityScapes, the sparse training improves all methods including *Joint Train* and all gradient manipulation methods, as shown in Table 25.

FAMO

FAMO (Liu et al., 2023a) is an approximation method for gradient manipulation by using the history of loss to compute the current task weight. We also try our sparse training with FAMO on NYU-v2, CelebA, Clevr, SmallORB datasets with ViT, SAM, MTAN and Swin models. As shown in Table 27, Table 29, ?? and Table 28, even for the approximation method, sparse training method achieves the best results and further show the effectiveness of our sparse training methods.

Methods	Average incidence of GC (%)	
	All epochs	Last 50% epochs
Joint Train	47.61	48.78
w/ ST	46.96 (0.65)	48.48 (0.30)
PCGrad	48.48	50.83
w/ ST	47.24 (1.24)	48.88 (1.95)
CAGrad	48.21	50.23
w/ ST	48.33 (-0.12)	50.40(-0.17)
GradDrop	47.36	48.72
w/ ST	47.13 (0.23)	48.57 (0.15)
MGDA	44.56	45.65
w/ ST	44.30 (0.26)	44.26(1.39)
IMTL-G	46.89	47.77
w/ ST	45.03 (1.86)	46.32(1.45)
NashMTL	46.83	47.67
w/ ST	46.78(0.05)	47.34(0.33)

Table 23: Average incidence of gradient conflict between tasks over epochs for different methods. The improvement by sparse training is provided in (●). We calculate the average incidence of gradient conflict over all epochs and the last 50% epochs during training Swin on CelebA.

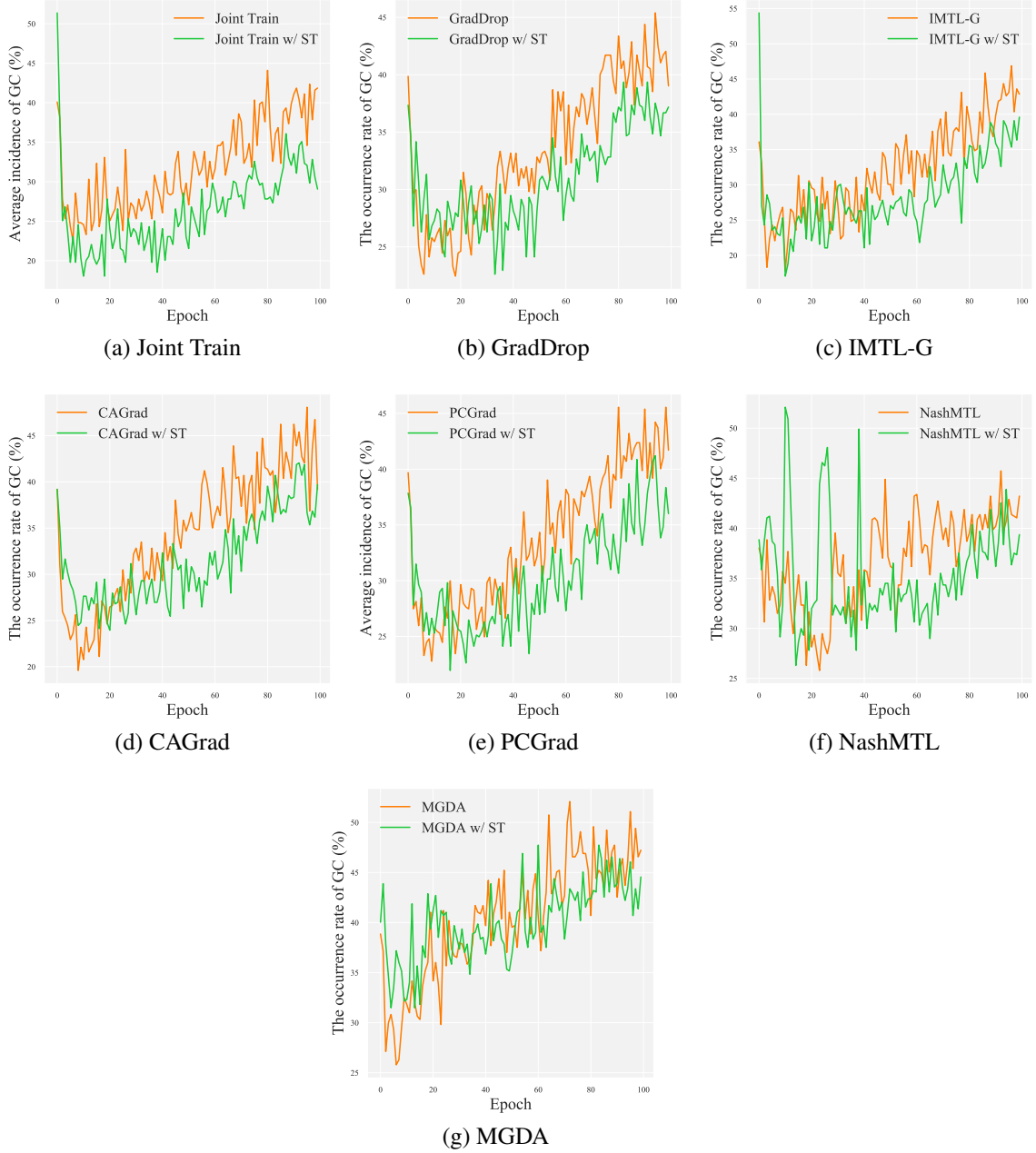


Figure 14: The number of occurrence gradient conflictions between tasks during training SAM on NYUv2 dataset.

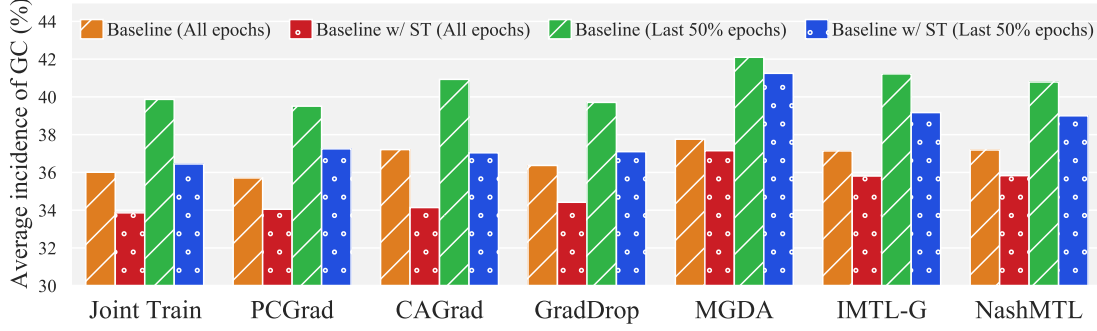


Figure 15: The average occurrence percentage of gradient conflict over epochs (all epochs/last 50% epochs) during training on MTAN model with NYU-v2 datasets was evaluated using various methods, including joint training and gradient manipulation techniques.

Methods	Object Azimuth		Camera Elevation		$\Delta m\%$ ↓
	Top 1 ↑	Top 5 ↑	Top 1 ↑	Top 5 ↑	
STL	32.92	70.06	36.56	94.67	—
Joint Train	28.01	67.05	29.84	89.75	10.70
w/ ST	27.33	68.35	30.73	89.87	10.11
PCGrad	28.79	67.85	30.10	88.44	9.99
w/ ST	27.539	67.92	31.18	90.18	9.71
CAGrad	28.72	68.42	29.33	87.93	10.50
w/ ST	28.59	68.21	29.82	88.37	10.22
GradDrop	27.50	66.13	29.86	88.50	11.73
w/ ST	28.34	67.79	29.52	88.38	10.76
MGDA	30.82	70.13	27.29	86.16	10.15
w/ ST	28.28	68.88	30.01	89.47	9.79
IMTL-G	29.57	69.92	28.51	86.74	10.19
w/ ST	27.65	69.09	30.01	89.66	10.15
NashMTL	27.02	66.88	30.83	89.74	10.84
w/ ST	28.17	67.93	31.01	89.35	9.57

Table 24: The test performance on SmallNORB dataset trained on ViT. The green cell color indicates that sparse training improves the performance of joint training or gradient manipulation methods. The best result is highlighted in bold.

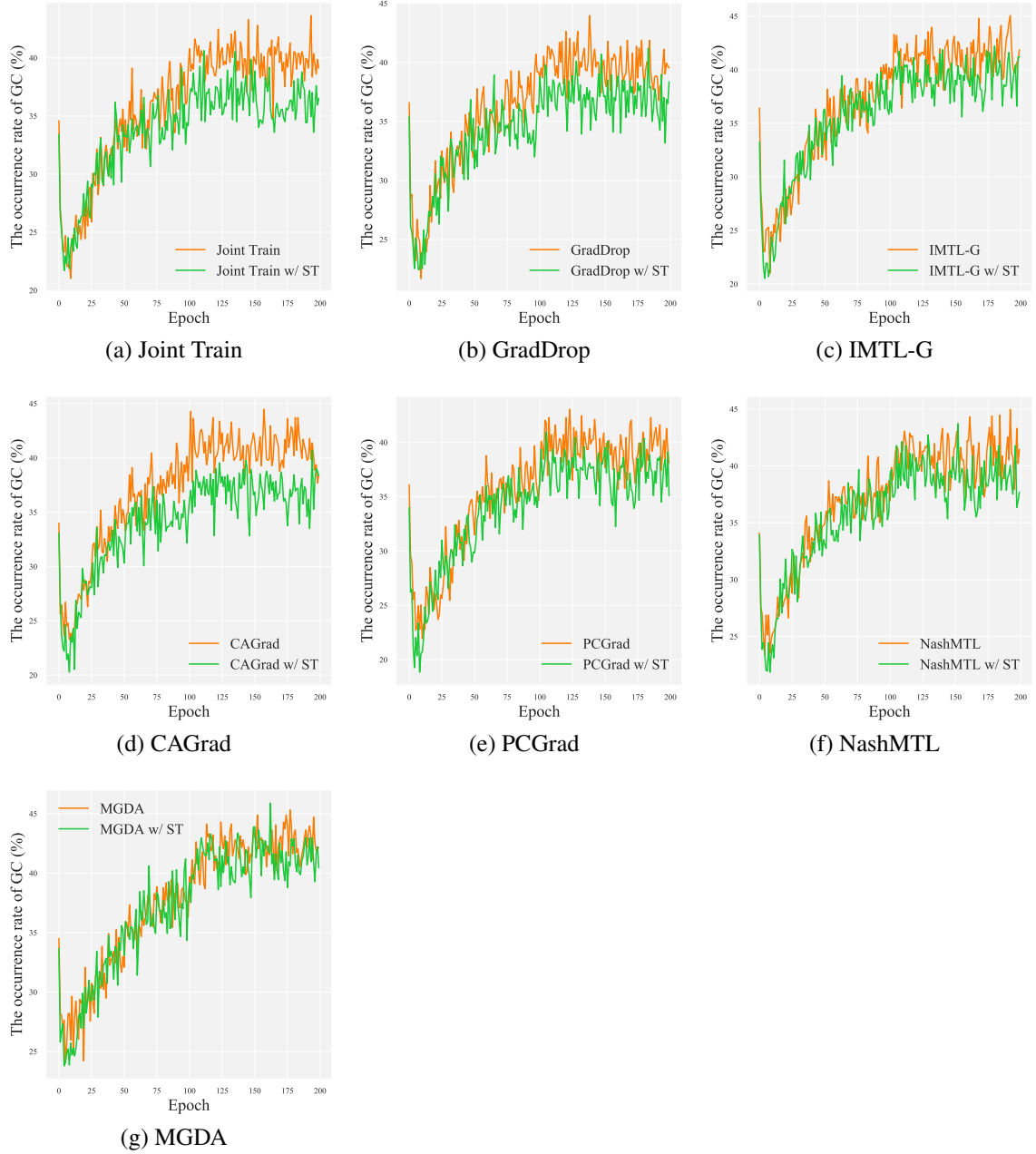


Figure 16: The number of occurrence gradient conflictions between tasks during tuning MTAN on NYUv2 dataset.



Figure 17: The number of occurrence gradient conflictions between tasks during tuning Swin on CelebA dataset.

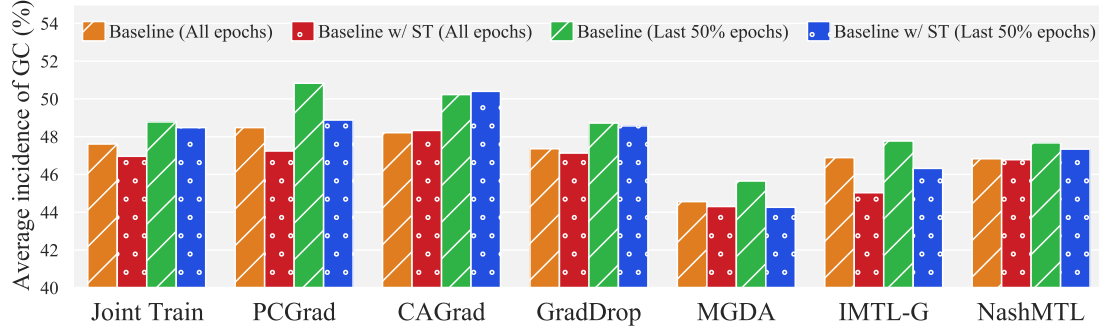


Figure 18: The average occurrence percentage of gradient conflict over epochs (all epochs/last 50% epochs) during training on Swin model with CelebA datasets was evaluated using various methods, including joint training and gradient manipulation techniques.

Methods	Segmentation		Depth		$\Delta m\% \downarrow$
	mIoU \uparrow	Pix Acc \uparrow	Abs Err \downarrow	Rel Err \downarrow	
STL	77.61	94.15	0.0122	35.68	—
Joint Train	78.14	94.29	0.0174	59.21	26.87
w/ ST	78.34	94.34	0.0143	55.00	17.48
PCGrad	77.79	94.21	0.0155	51.99	19.96
w/ ST	77.79	94.26	0.0160	51.99	19.22
CAGrad	76.82	93.70	0.0138	53.74	16.26
w/ ST	77.20	94.01	0.0150	39.85	8.88
GradDrop	77.91	94.28	0.0154	55.58	20.34
w/ ST	78.34	94.38	0.0163	48.95	17.45
MGDA	69.91	92.17	0.0124	40.68	6.91
w/ ST	68.38	91.91	0.0128	33.19	3.17
IMTL-G	77.55	94.10	0.0135	47.17	10.65
w/ ST	75.75	93.98	0.0138	40.16	7.10
NashMTL	77.51	94.22	0.0152	36.36	6.68
w/ ST	76.87	94.09	0.0148	33.30	3.99

Table 25: The test performance on CityScapes dataset training on MTAN model. The green cell color indicates that sparse training improves the performance of joint training or gradient manipulation methods. The best result is highlighted in bold.

Methods	Average incidence of GC (%)	
	All epochs	Last 50% epochs
Joint Train	39.72	40.99
w/ ST	38.79 (0.93)	40.02 (0.97)
PCGrad	39.98	41.06
w/ ST	38.66(1.32)	39.97(1.09)
CAGrad	39.39	40.94
w/ ST	37.77(1.62)	39.42(1.52)
GradDrop	39.32	40.72
w/ ST	39.03(0.29)	40.12(0.60)
MGDA	36.37	39.69
w/ ST	36.14(0.23)	39.38(0.31)
IMTL-G	37.72	39.51
w/ ST	36.83(0.89)	38.72(0.79)
NashMTL	38.40	40.69
w/ ST	38.04(0.36)	40.26(0.43)

Table 26: Average incidence of gradient conflict between tasks over epochs for different methods. The improvement by sparse training is provided in (●). We calculate the average incidence of gradient conflict over all epochs and the last 50% epochs during training MTAN on CityScapes.

Methods	Segmentation		Depth		Surface Normal					$\Delta m\%$ ↓
	mIoU ↑	Pix Acc ↑	Abs Err ↓	Rel Err ↓	Angle Distance ↓		Within t° ↑			
					Mean	Median	11.25	22.5	30	
FAMO	57.64	78.59	0.3574	0.1463	19.396	12.846	45.61	71.87	80.59	−0.5669
w/ ST	57.68	78.79	0.3520	0.1430	19.279	12.711	46.12	72.06	80.72	−1.353

Table 27: The test performance on NYU-v2 dataset training on SAM model. The green cell color indicates that sparse training improves the performance of joint training or gradient manipulation methods. The best result is highlighted in bold.

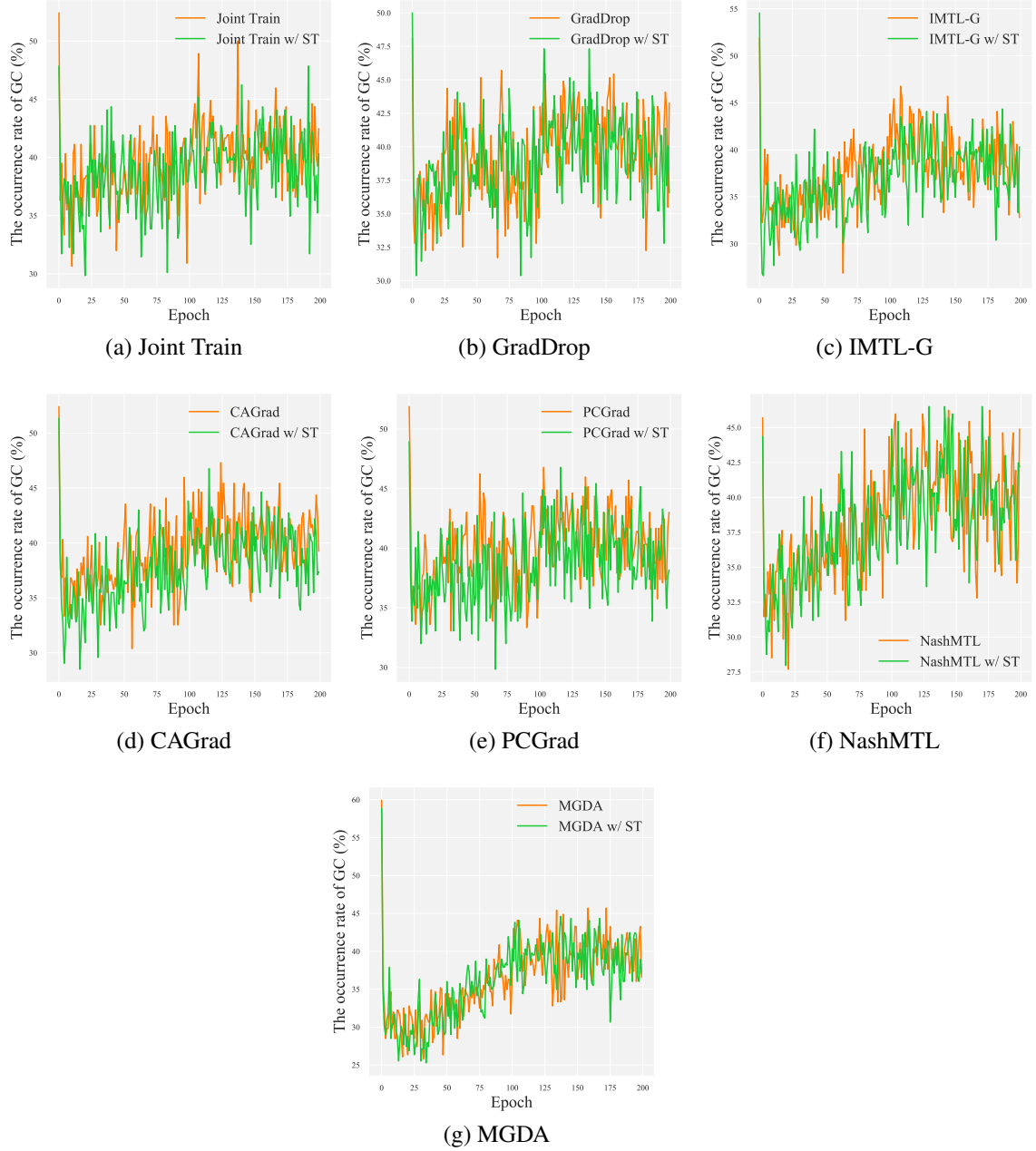


Figure 19: The number of occurrence gradient conflictions between tasks during tuning MTAN on CityScapes dataset.

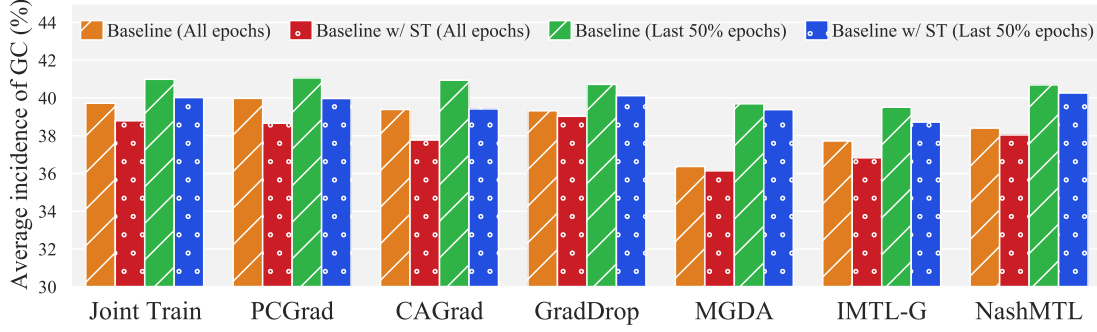


Figure 20: The average occurrence percentage of gradient conflict over epochs (all epochs/last 50% epochs) during training on MTAN model with CityScapes datasets was evaluated using various methods, including joint training and gradient manipulation techniques.

Methods	CelebA	Clevr		NYU-v2	
	$\Delta m\% \downarrow$	Counting	Depth	$\Delta m\% \downarrow$	$\Delta m\% \downarrow$
	(F1)	(Top 1 \uparrow)	(Top 1 \uparrow)		
FAMO	2.35	55.83	56.80	3.16	-4.10
w/ ST	2.32	62.57	56.04	-1.93	-4.46

Table 28: The test performance on CelebA, Clevr and NYU-v2 dataset. CelebA is trained on Swin Transformer and Clevr is trained on ViT. NYU-v2 is trained on MTAN. The green cell color indicates that sparse training improves the performance of joint training or gradient manipulation methods. The best result is highlighted in bold.

Methods	Segmentation		Depth		Surface Normal					$\Delta m\%$ \downarrow
	mIoU \uparrow	Pix Acc \uparrow	Abs Err \downarrow	Rel Err \downarrow	Angle Distance \downarrow		Within t° \uparrow			
					Mean	Median	11.25	22.5	30	
FAMO	38.88	64.90	0.5474	0.2194	25.06	19.57	29.21	56.61	68.98	-4.10
w/ ST	37.85	65.27	0.5543	0.2215	25.09	19.15	30.03	57.49	69.52	-4.46

Table 29: The test performance on NYU-v2 dataset training on MTAN model. The green cell color indicates that sparse training improves the performance of joint training or gradient manipulation methods. The best result is highlighted in bold.

	Object	Attribute	Category	Relation	Global
Verify	86.21%	83.00%	—	87.82%	95.56%
Query	—	71.20%	62.88%	52.84%	55.74%
Choose	—	90.17%	92.03%	87.19%	96.76%
Logical	88.92%	76.17%	—	—	—
Compare	—	71.23%	—	—	—

Table 31: The accuracy of the validation set of GQA dataset (Hudson and Manning, 2019) on *LLaVA-1.5-13b* (Liu et al., 2024a). and represent binary (yes/no) and open question respectively. represents that this category contains both binary and open questions.

E Appendix to Chapter 7

E.1 Dataset collection

We collect our data from the validation set of the *GQA* dataset (Hudson and Manning, 2019), which is designed for visual reasoning and compositional question-answering. Derived from the Visual Genome dataset (Krishna et al., 2017), *GQA* provides real-world images enriched with detailed scene graphs. Questions in *GQA* dataset are categorized along two dimensions: structure (5 classes, defining question formats) and semantics (5 classes, specifying the main subject’s semantic focus). Structural classes include: (1) *verify* (yes/no questions), (2) *query* (open questions), (3) *choose* (questions with two alternatives), (4) *logical* (logical inference), and (5) *compare* (object comparisons). Semantic classes are: (1) *object* (existence questions), (2) *attribute* (object properties or positions), (3) *category* (object identification within a class), (4) *relation* (questions about relational subjects/objects), and (5) *global* (overall scene properties like weather or location). Based on the combination of these two dimensions, the questions in *GQA* are categorized into 15 groups, as shown in Table 31.

We select 6 out of 15 groups according to the following steps. First, we exclude the *verify* type, as it is quite simple and involves only straightforward binary questions (e.g., “Is the apple red?”). Then we focus on types with an average accuracy above 80% on *LLaVA-1.5-13b* model (Liu et al., 2024a), retaining *ChooseAttr*, *ChooseCat*, *ChooseRel*, and *LogicalObj*. *ChooseGlo* is excluded due to its limited sample size in the validation set of *GQA* (only 556 instances). After that, to enhance question-type diversity, we select high-performing subtypes (accuracy > 80%) in *CompareAttr* and *QueryAttr* from the *GQA* dataset. Specifically, we use the *positionQuery* subtype for spatial-relation questions in *QueryAttr* and the *twoCommon* subtype for comparing common attributes between two objects in *CompareAttr*. Finally, for each type of the six selected types, we sample at most 1000 data that are predicted correctly on model *LLaVA-1.5-13b* from the validation set of *GQA* resulting in our final data in this paper, as shown in Table 7.1.

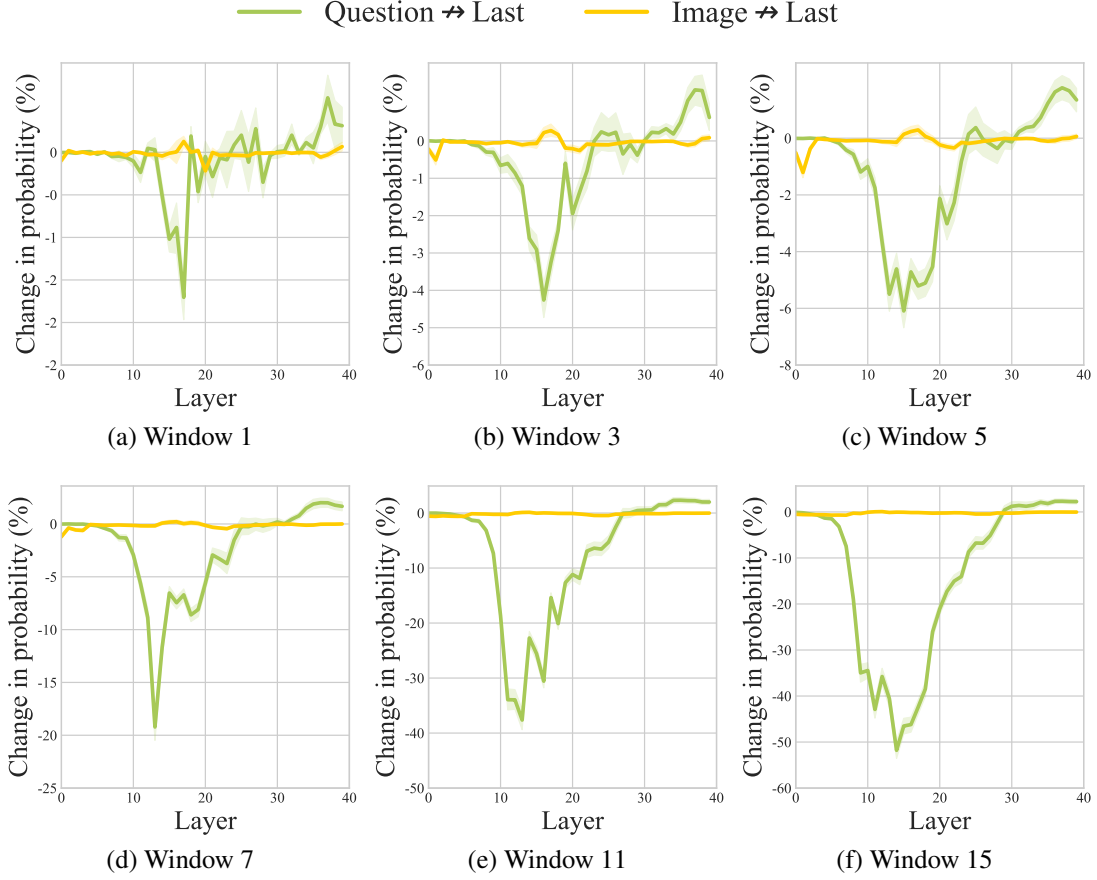


Figure 21: The relative changes in prediction probability on *LLaVA-1.5-13b* with the tasks of *ChooseAttr* for different window size. The *Question → Last* and *Image → Last* represent preventing *last position* from attending to *Question* and *Image* respectively.

E.2 Informaion flow for different window size k

In the main body of the paper, we use a window size $k = 9$ for an easier analysis of the internal working mechanism of the multimodal large language models when performing multimodal tasks. We present the relative change in probability on *LLaVA-1.5-13b* and the task of *ChooseAttr* with different window sizes of $k = 1, 3, 5, 7, 9, 11, 15$. The resulting information flow between different parts of the input sequence (*image* and *question*) and *last position*, and between *image* and *question* are shown in Figure 21 and Figure 22, respectively. Overall, the observations on the information flow are consistent across different window sizes k . Specifically, the critical information flow from *question* to *last position* occurs in the middle layers while the critical information flow from *image* to *last position* is not observed across different window sizes, as shown in Figure 21. For critical information from *image* to *question*, the two different critical information flows are observed across different window sizes where both occur in

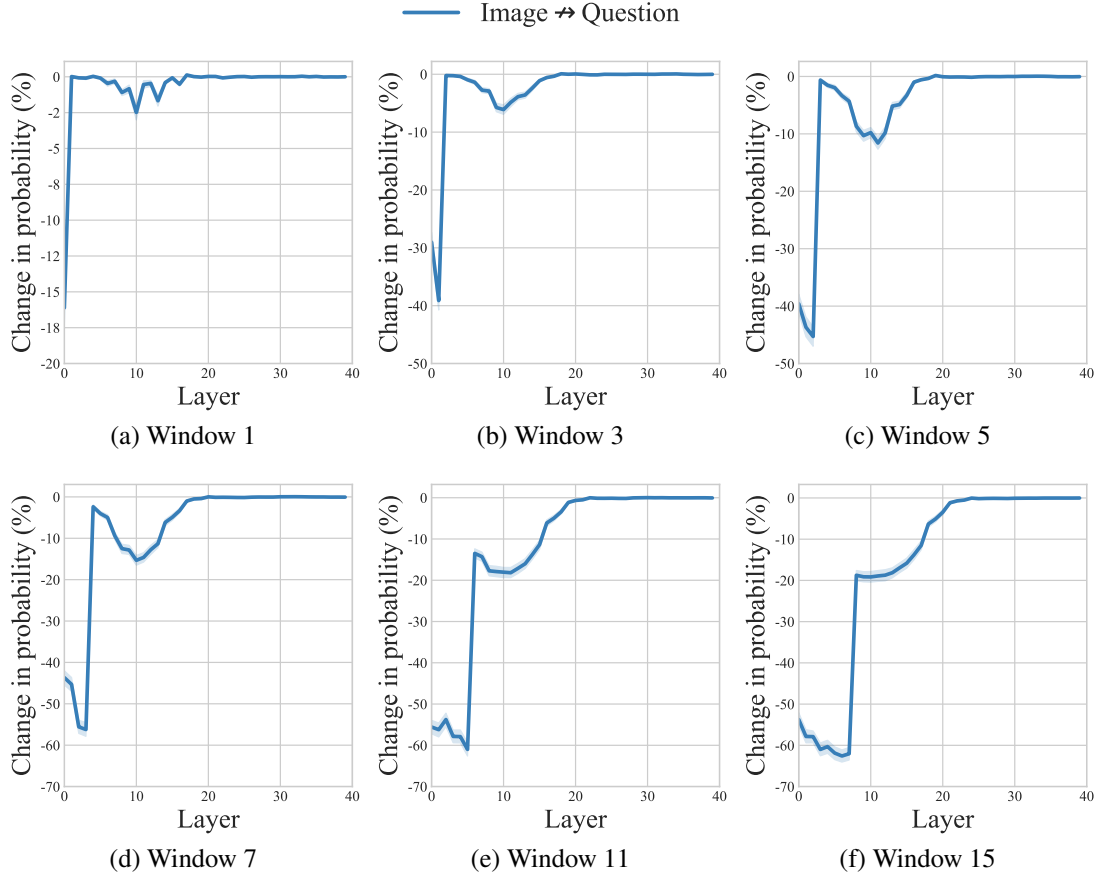


Figure 22: The relative changes in prediction probability when blocking attention edges from the *question* positions to the *image* positions on *LLaVA-1.5-13b* with the tasks of *ChooseAttr* for different window sizes.

lower layers and sequentially follow each other, as illustrated in Figure 22. In addition, we observe that as the window size increases, the two information flows gradually merge into one, which is because the larger window encompasses layers involved in both information flows. Moreover, the decrease in the prediction probability becomes more pronounced with the increase of the value k . This is expected, as blocking more attention edges in the computation hinders the model’s ability to properly contextualize the input.

E.3 Changes in probability of the last sub-word generation

In this paper, the *answer* in our used dataset normally contains one word or one phrase, which might result in several sub-word tokens. In the main body of the paper, we present the relative change in probability of the first generated sub-word while the final generated sub-words also yield similar results. Specifically, we conduct the

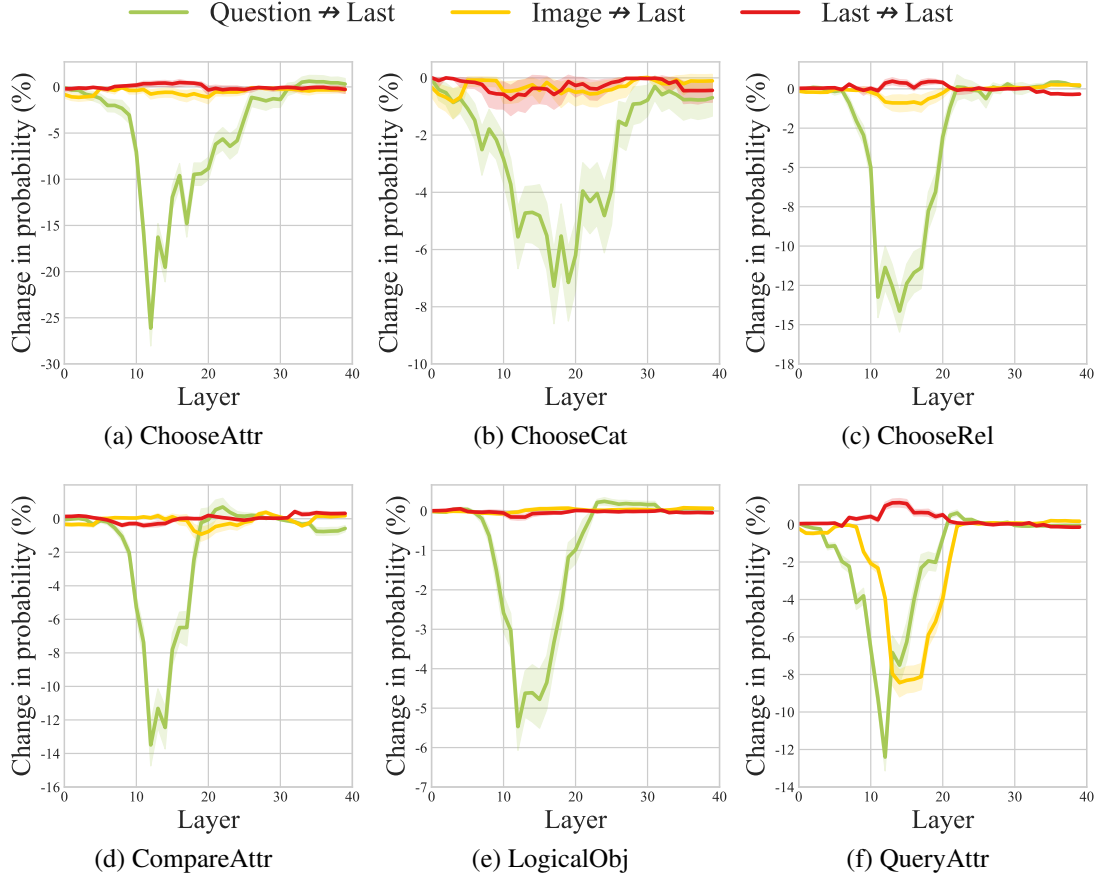


Figure 23: The relative changes in prediction probability for the final generated sub-word of the *answer* on *LLaVA-1.5-13b* with six VQA tasks. The *Question*→*Last*, *Image*→*Last* and *Last*→*Last* represent preventing *last position* from attending to *Question*, *Image* and itself respectively.

same experiments as in the main body of the paper: six tasks (*ChooseAttr*, *ChooseCat*, *ChooseRel*, *LogicalObj*, *QueryAttr* and *CompareAttr*) on *LLaVA-1.5-13b* model. Instead of calculating the relative change in probability for the first generated sub-word token, we calculate that for the final generated sub-word token of the correct answer word. As shown in Figure 23, Figure 24 and Figure 25, the information flow from different parts of the input sequence (*image* and *question*) to *last position*, from *image* to *question* and from different image patches (*related image patches* and *other image patches*) to *question* are consistent with the observations in Figure 7.3, Figure 7.4 and Figure 7.5 in the main body of the paper.

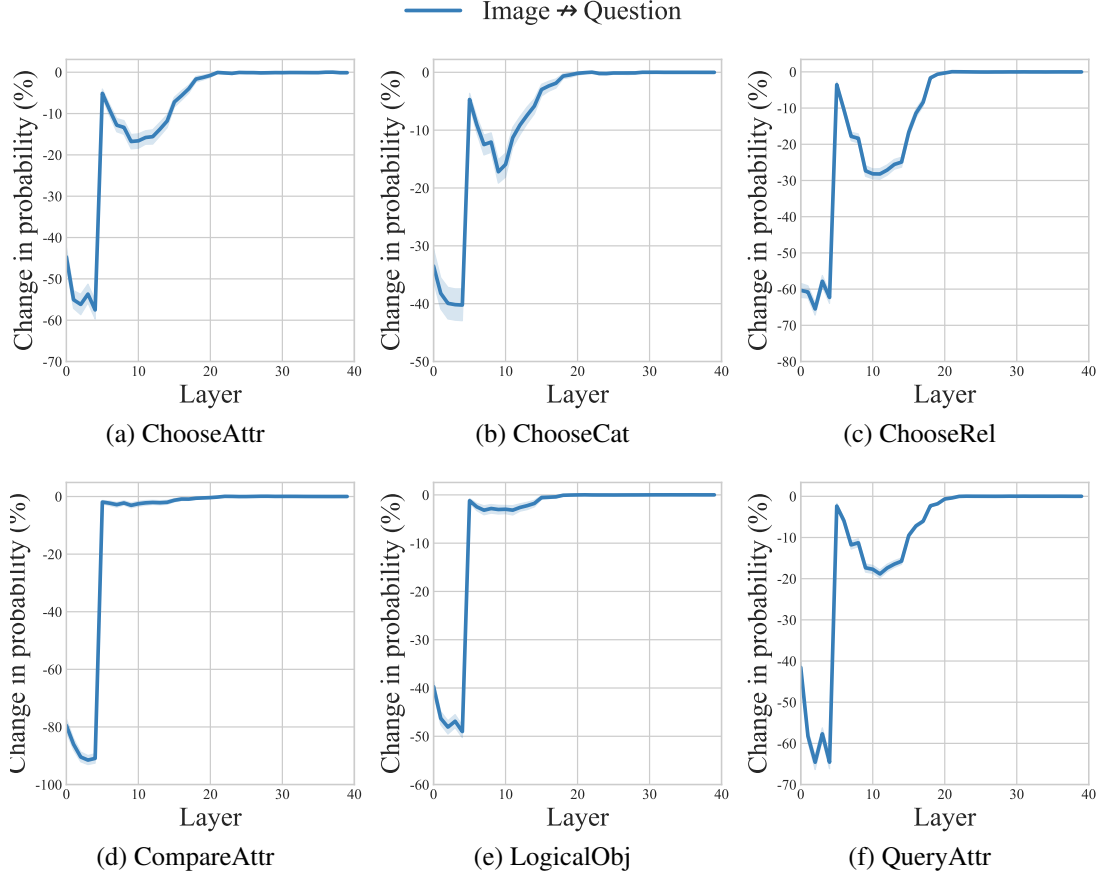


Figure 24: The relative changes in prediction probability for the final generated subword of the *answer* when blocking attention edges from the *question* positions to the *image* positions on *LLaVA-1.5-13b* with six *VQA* tasks.

E.4 Constructing multimodal semantic representations

We have investigated how multimodal information is integrated through the MHAT module in Section 7.6. We now take a closer look at how the multimodal semantic representation is constructed.

Experiment To identify which module in the transformer contributes to the formulation of multimodal semantic information within hidden representations, we employ a *module knockout* approach to evaluate the significance of individual transformer modules. As shown in Equation (7.1), the hidden representation at layer ℓ is computed by adding \mathbf{a}_i^ℓ and \mathbf{f}_i^ℓ to $\mathbf{h}_i^{\ell-1}$, where \mathbf{a}_i^ℓ and \mathbf{f}_i^ℓ are derived from the MHAT (Equation (7.2)) and MLP (Equation (7.5)) modules, respectively. This allows us to determine which module contributes to constructing semantic information by selectively zeroing out the outputs of MHAT or MLP—two additive modules in the transformer layer. Specifically,

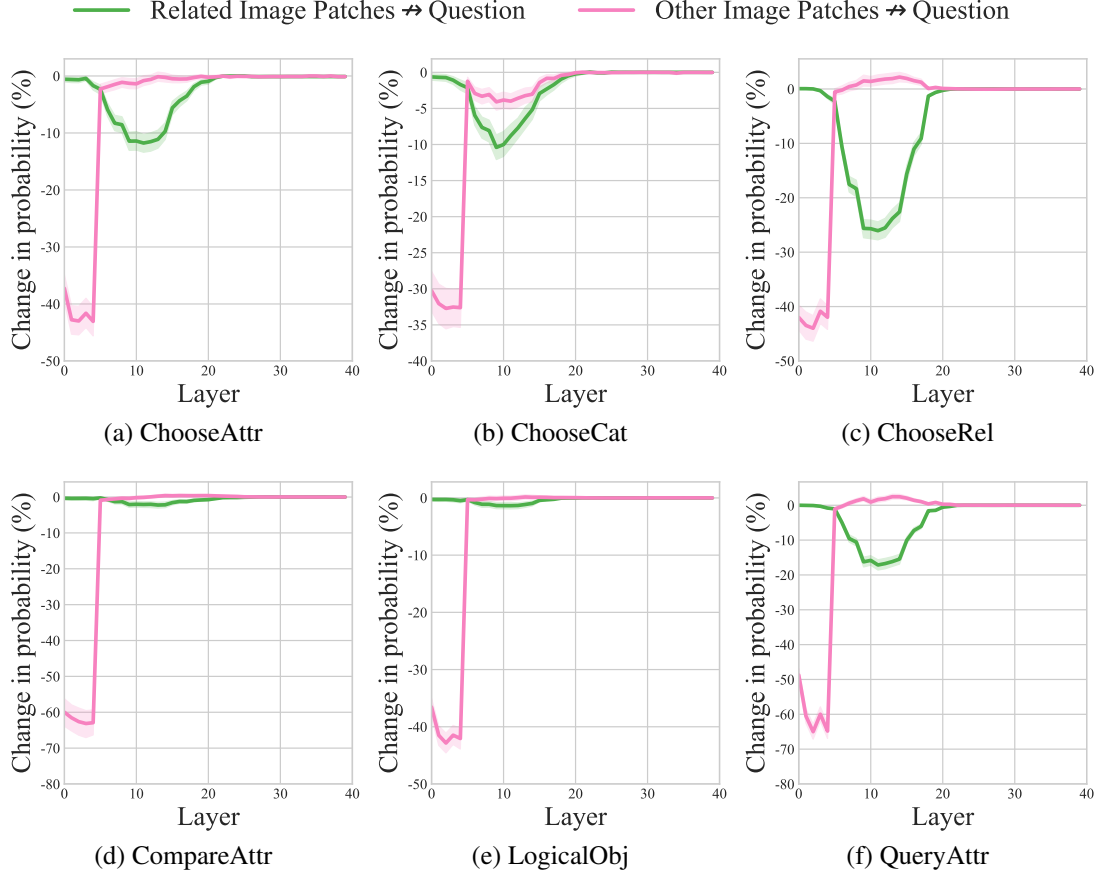


Figure 25: The relative changes in prediction probability for the final generated sub-word of the answer on *LLaVA-1.5-13b* with six VQA tasks. *Related Image Patches*→*question* and *Other Image Patches*→*question* represent blocking the position of *question* from attending to that of different image patches, region of interest and remainder, respectively.

for each layer ℓ , we intervene by setting $\mathbf{a}_i^{\ell'}$ or $\mathbf{f}_i^{\ell'}$ ($i \in \mathbb{Q}$) to zero across 9 consecutive layers $\{\ell'\}_{\ell'=\ell}^{\min\{\ell+8, L\}}$. We then measure the importance of constructing multimodal semantic information by observing the semantic change of the hidden representation corresponding to *question* position \mathbb{Q} at the final layer L . Our focus on layer L is inspired by Geva et al. (2023), who highlight that semantic information peaks in the final layer. We follow Wang et al. (2023), who evaluate the semantic content of a hidden representation using top-k words from this representation. We estimate semantic content using the top-10 words predicted from each hidden representation in \mathbb{Q} , derived from Equation (7.6), where \mathbf{h}_N^L is replaced by \mathbf{h}_i^L ($i \in \mathbb{Q}$). We then quantify the change in semantic content of hidden representation resulting from our interventions using Jaccard Similarity:

$$J(\mathbb{W}_o, \mathbb{W}_i) = \frac{|\mathbb{W}_o \cap \mathbb{W}_i|}{|\mathbb{W}_o \cup \mathbb{W}_i|} \quad (7)$$

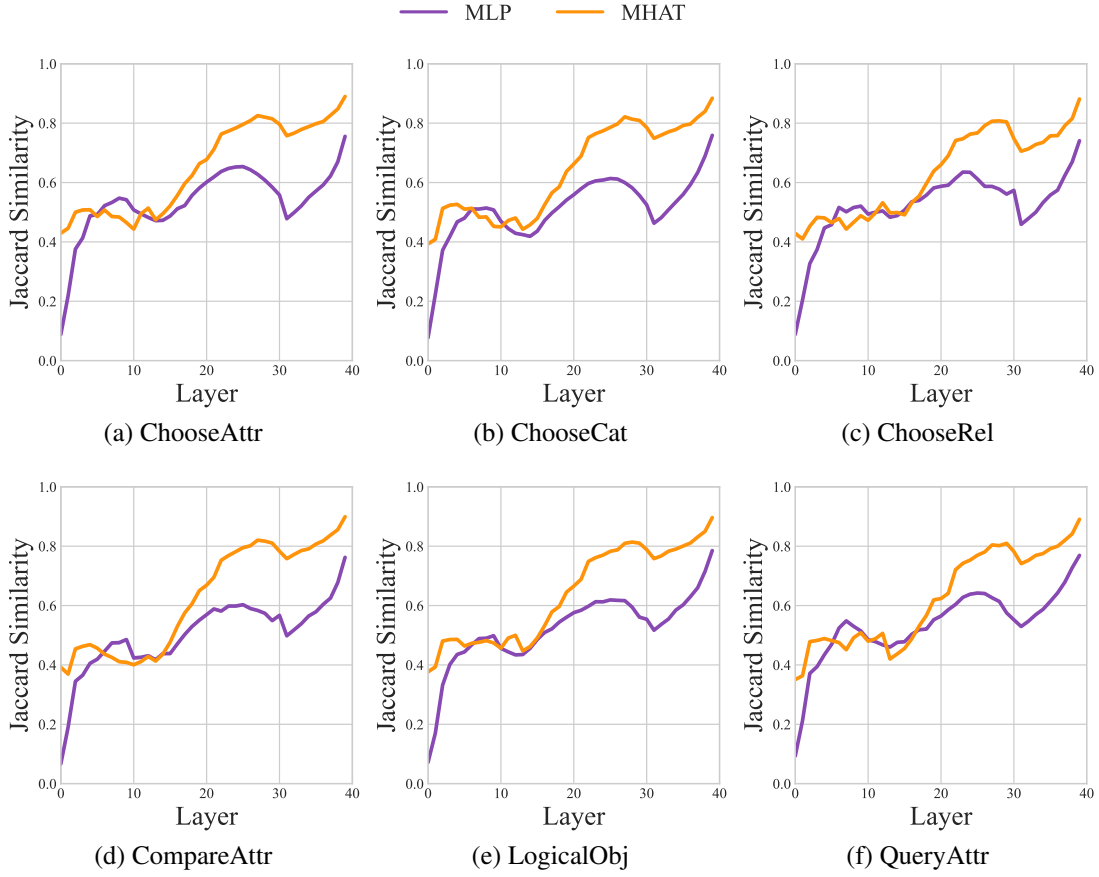


Figure 26: The Jaccard similarity between the predicted words of the original model and those of the intervened model, with the MLP and MHAT modules removed individually (*LLaVA-1.5-13b*).

where \mathbb{W}_o and \mathbb{W}_i denote the sets of $10 \cdot |\mathbb{Q}|$ predicted words from the original and intervened models, respectively.

Observation: The MLP module plays a greater role in constructing semantic representations compared to the MHAT Module As shown in Figure 26 for model *LLaVA-1.5-13b*, removing the MLP module severely impacts semantic representation, reducing average Jaccard Similarity across six tasks by $\sim 90\%$ when MLP is removed in the first layer and $\sim 25\%$ in the last layer. In contrast, removing the MHAT module has a smaller effect, with reductions of $\sim 60\%$ and $\sim 10\%$ at the first and last layers, respectively. This highlights the MLP module’s important role in generating multi-modal semantic representations. These results align with the findings from (Dai et al., 2022; Geva et al., 2021; Meng et al., 2022), who demonstrate that factual information is primarily stored in the MLP module, emphasizing its contribution to enriching semantic information. This is also observed in the model *LLaVA-1.5-7b*, as shown in Figure 27.

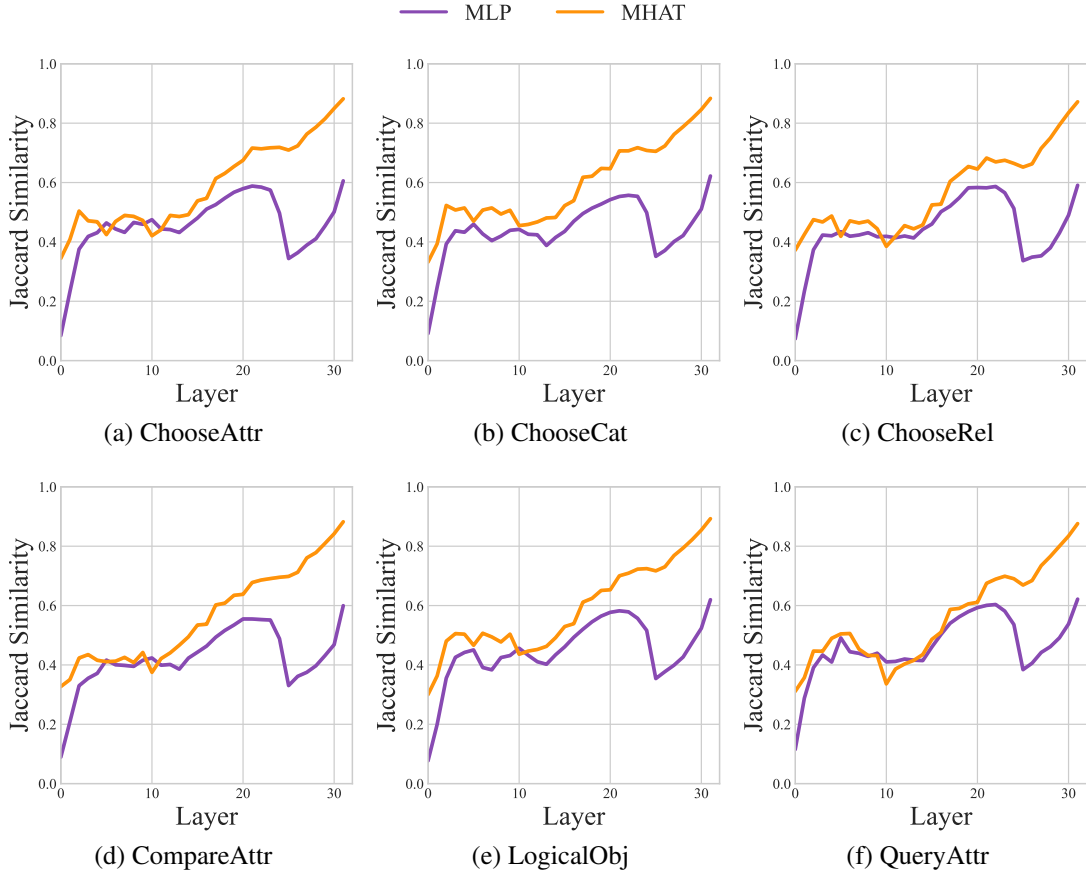


Figure 27: The Jaccard similarity between the predicted words of the original model and those of the intervened model, with the MLP and MHAT modules removed individually (*LLaVA-1.5-7b*).

E.5 Experiments on other models

We conduct the same experiments (six *VQA* task types) as in the main body of the paper with the other three models. Six *VQA* task types include (*ChooseAttr*, *ChooseCat*, *ChooseRel*, *LogicalObj*, *QueryAttr* and *CompareAttr*). The other three models include *LLaVA-1.5-7b*, *LLaVA-v1.6-Vicuna-7b* and *Llama3-LLaVA-NEXT-8b*.

LLaVA-1.5-7b

LLaVA-1.5-7b is a small version of *LLaVA-1.5-13b* presented in the main body of the paper. It contains 32 transformer blocks (layers) instead of 40 layers in *LLaVA-1.5-13b*. The information flow from different parts of the input sequence (*image* and *question*) to *last position*, from *image* to *question* and from different image patches (*related image patches* and *other image patches*) to *question*, as shown in Figure 28, Figure 29 and Figure 30 respectively, are consistent with the observations for the *LLaVA-1.5-13b*

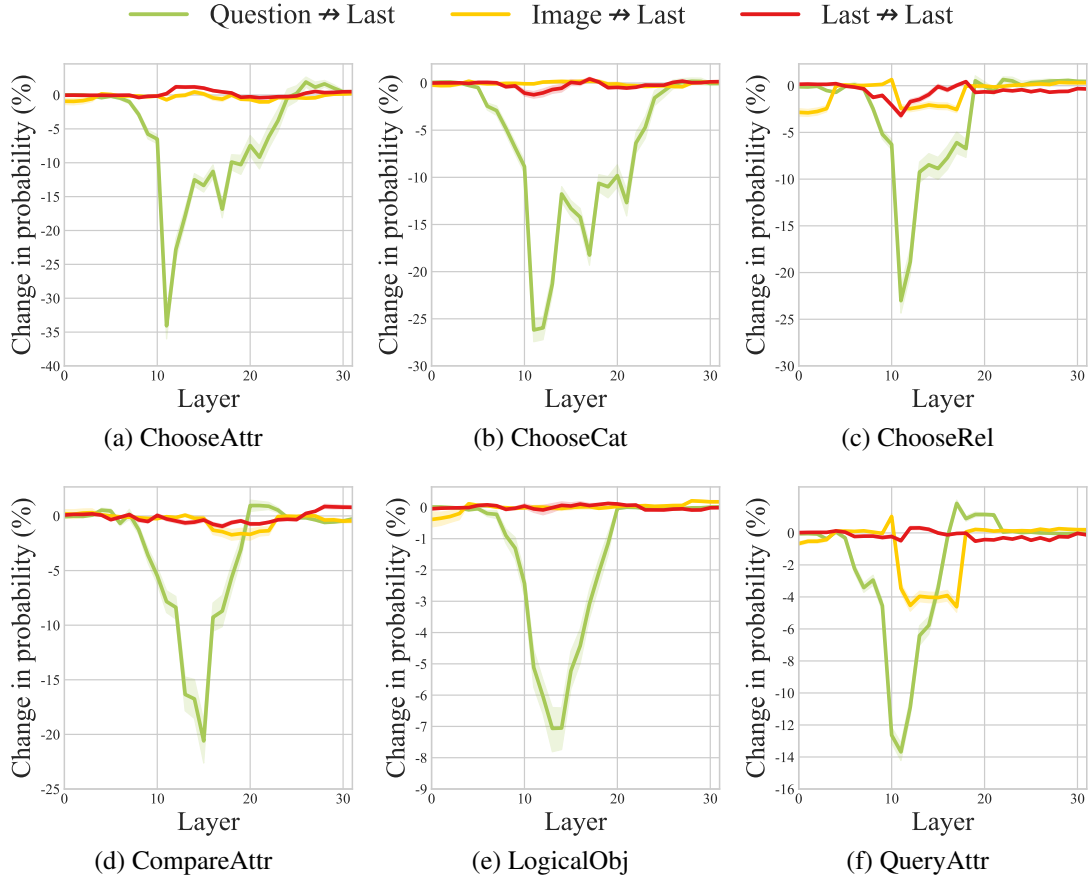


Figure 28: The relative changes in prediction probability on *LLaVA-1.5-7b* with six VQA tasks. The *Question* → *Last*, *Image* → *Last* and *Last* → *Last* represent preventing *last* position from attending to *Question*, *Image* and itself respectively.

model, as shown in Figure 7.3, Figure 7.4 and Figure 7.5 respectively, in the main body of the paper. Specifically, the model first propagates critical information twice from the *image* positions to the *question* positions in the lower-to-middle layers of the MLLM. For the twice multimodal information integration, the first one focuses on producing the generative representations over the whole image while the second one tends to construct question-related representation. Subsequently, in the middle layers, the critical multimodal information flows from the *question* positions to the *last position* for the final prediction. The difference between the two models is the magnitude of reduction in the probability when blocking the attention edge between *image* and *question*. In model *LLaVA-1.5-7b*, the first drop is rather smaller than that in model *LLaVA-1.5-13b*. However, this does not conflict with our conclusion that the information flows from *image* to *question* twice and one after the other in the main body of the paper. Moreover, the probability change of the answer word across all layers as shown in Figure 31 is also consistent with the result in Figure 7.6 in the main body of

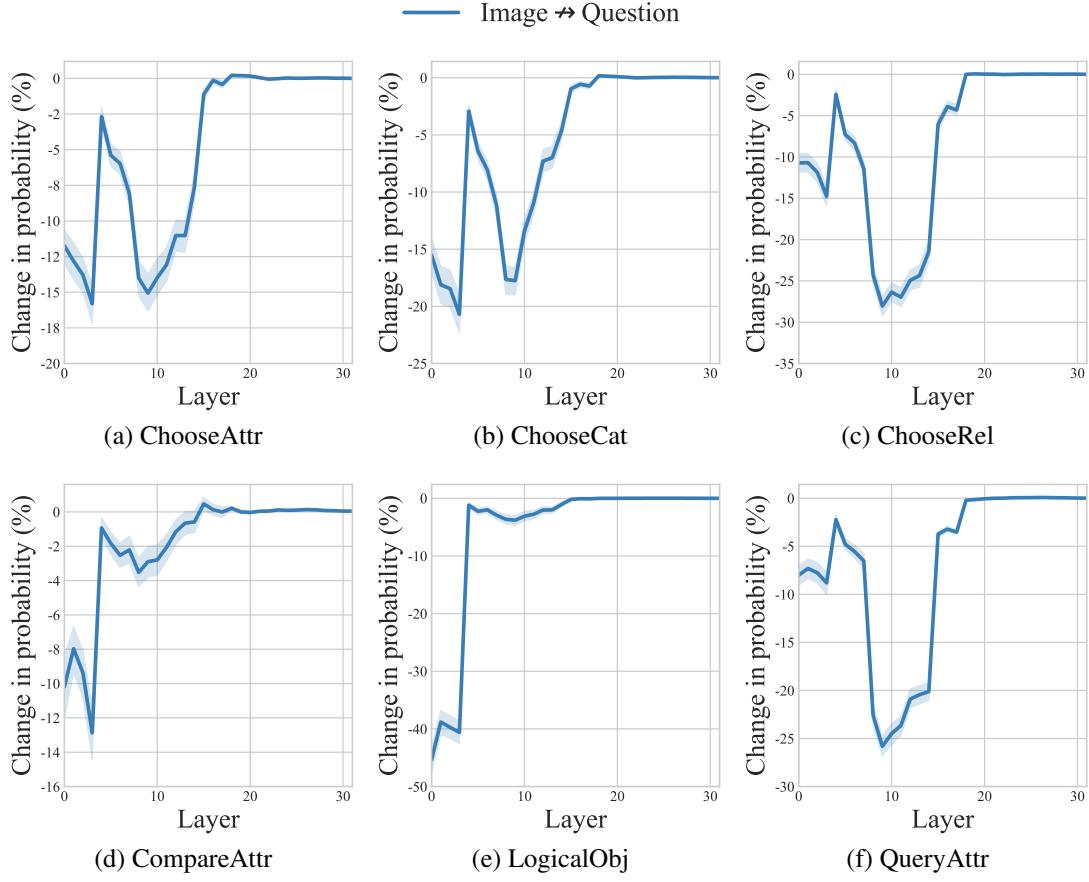


Figure 29: The relative changes in prediction probability when blocking attention edges from the *question* positions to the *image* positions on *LLaVA-1.5-7b* with six *VQA* tasks.

the paper. Specifically, the model first generates the answer semantically in the middle layers and then refines the syntactic correctness of the answer in the higher layers.

LLaVA-v1.6-Vicuna-7b

LLaVA-v1.6-Vicuna-7b has the similar architecture with *LLaVA-1.5-13b* in the main body of the paper. The difference between them includes the layer number and the way processing image patch features. The *LLaVA-v1.6-Vicuna-7b* has 32 layers versus 40 layers in *LLaVA-1.5-13b*. *LLaVA-1.5-13b* directly feeds the original fixed-length image patch features from the image encoder into the LLM as input tokens. In contrast, *LLaVA-v1.6-Vicuna-7b* employs a dynamic high-resolution technique, which dynamically adjusts image resolution, resulting in variable-length image patch features with higher resolution. Specifically, the higher resolution is implemented by splitting the image into grids and encoding them independently.

The information flow from different parts of the input sequence (*image* and *question*)

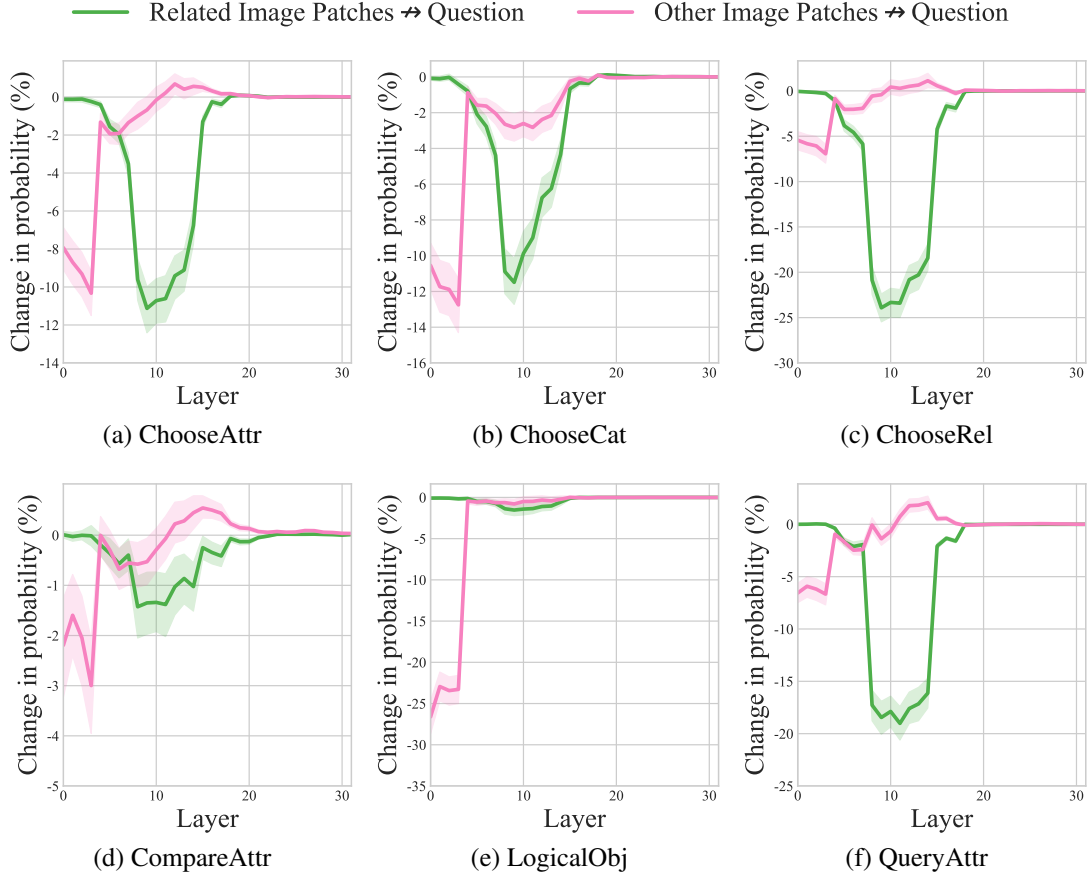


Figure 30: The relative changes in prediction probability on *LLaVA-1.5-7b* with six VQA tasks. *Related Image Patches → question* and *Other Image Patches → question* represent blocking the position of *question* from attending to that of different image patches, region of interest and remainder, respectively.

to *last position*, from *image* to *question* and from different image patches (*related image patches* and *other image patches*) to *question*, as shown in Figure 32, Figure 33 and Figure 34 respectively, are consistent with the observations for the *LLaVA-1.5-13b* model, as shown in Figure 7.3, Figure 7.4 and Figure 7.5 respectively, in the main body of the paper. Specifically, the model first propagates critical information twice from the *image* positions to the *question* positions in the lower-to-middle layers of the MLLM. For the dual-stage multimodal information integration, the first stage emphasizes generating holistic representations of the entire image, while the second stage focuses on constructing representations that are specifically aligned with the given question. Subsequently, in the middle layers, the critical multimodal information flows from the *question* positions to the *last position* for the final prediction. Moreover, the probability change of the answer word across all layers as shown in Figure 35 is also consistent with the result in Figure 7.6 in the main body of the paper. Specifically, the

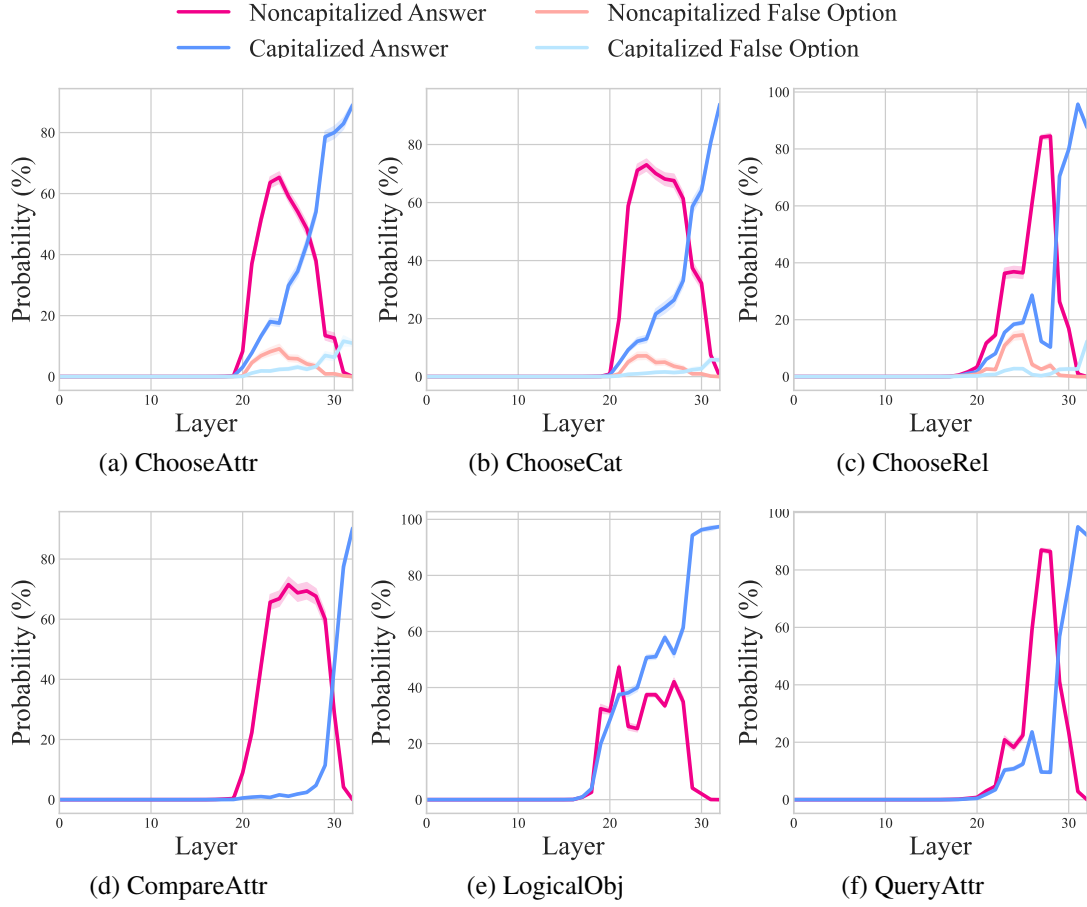


Figure 31: The probability of the answer word at the *last position* across all layers in LLaVA-1.5-7b with six VQA tasks. *Capitalized Answer* and *Noncapitalized Answer* represent the answer word with or without the uppercase of the initial letter, respectively. As the tasks of *ChooseAttr*, *ChooseCat* and *ChooseRel* contain *false option*, we also provide the probability of it.

model first generates the answer semantically in the middle layers and then refines the syntactic correctness of the answer in the higher layers.

Llama3-LLaVA-NEXT-8b

Llama3-LLaVA-NEXT-8b has quite different architecture with *LLaVA-1.5-13b* in the main body of the paper. The difference between them includes the layer number, the way of processing image patch features, and the attention mechanism. The *Llama3-LLaVA-NEXT-8b* has 32 layers verse 40 layers in *LLaVA-1.5-13b*. *LLaVA-1.5-13b* directly feeds the original fixed-length image patch features from the image encoder into the LLM as input tokens. In contrast, *Llama3-LLaVA-NEXT-8b* employs a dynamic high-resolution technique, which dynamically adjusts image resolution, resulting in variable-length

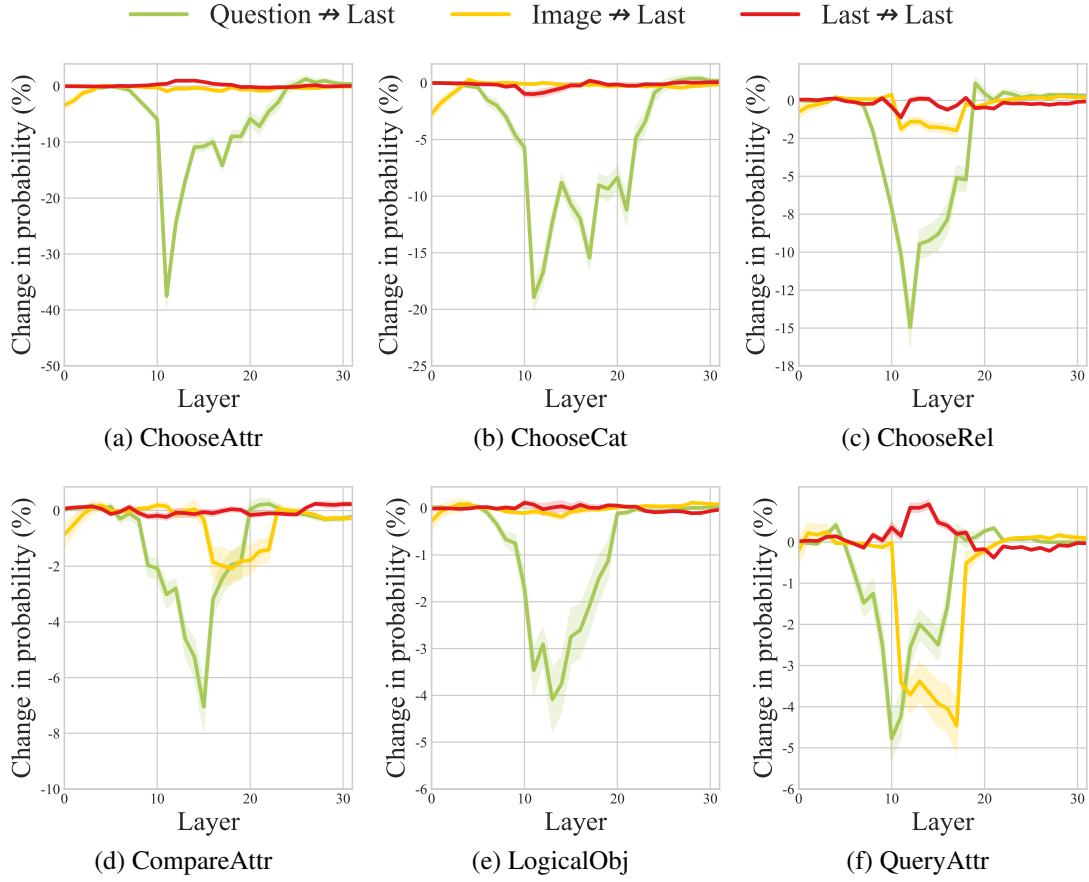


Figure 32: The relative changes in prediction probability on *LLaVA-v1.6-Vicuna-7b* with six VQA tasks. The *Question \rightarrow Last*, *Image \rightarrow Last* and *Last \rightarrow Last* represent preventing *last position* from attending to *Question*, *Image* and itself respectively.

image patch features with higher resolution. Specifically, the higher resolution is implemented by splitting the image into grids and encoding them independently. As for the attention mechanism, *LLaVA-1.5-13b* use a standard and dense transformer architecture (Vaswani et al., 2017b) while *Llama3-LLaVA-NEXT-8b* adopts grouped query attention (Ainslie et al., 2023) where the queries are grouped and the queries in the same group has shared key and value.

The information flow from different parts of the input sequence (*image* and *question*) to *last position*, from *image* to *question* and from different image patches (*related image patches* and *other image patches*) to *question*, as shown in Figure 36, Figure 37 and Figure 38 respectively, are consistent with the observations for the *LLaVA-1.5-13b* model, as shown in Figure 7.3, Figure 7.4 and Figure 7.5 respectively, in the main body of the paper. Although the information flow from *image* to *question* in Figure 37 appears to exhibit only a single drop, the Figure 38 reveals that, in lower layers, the information flow from *Other Image Patches* to the *question* play a dominant role compared to that

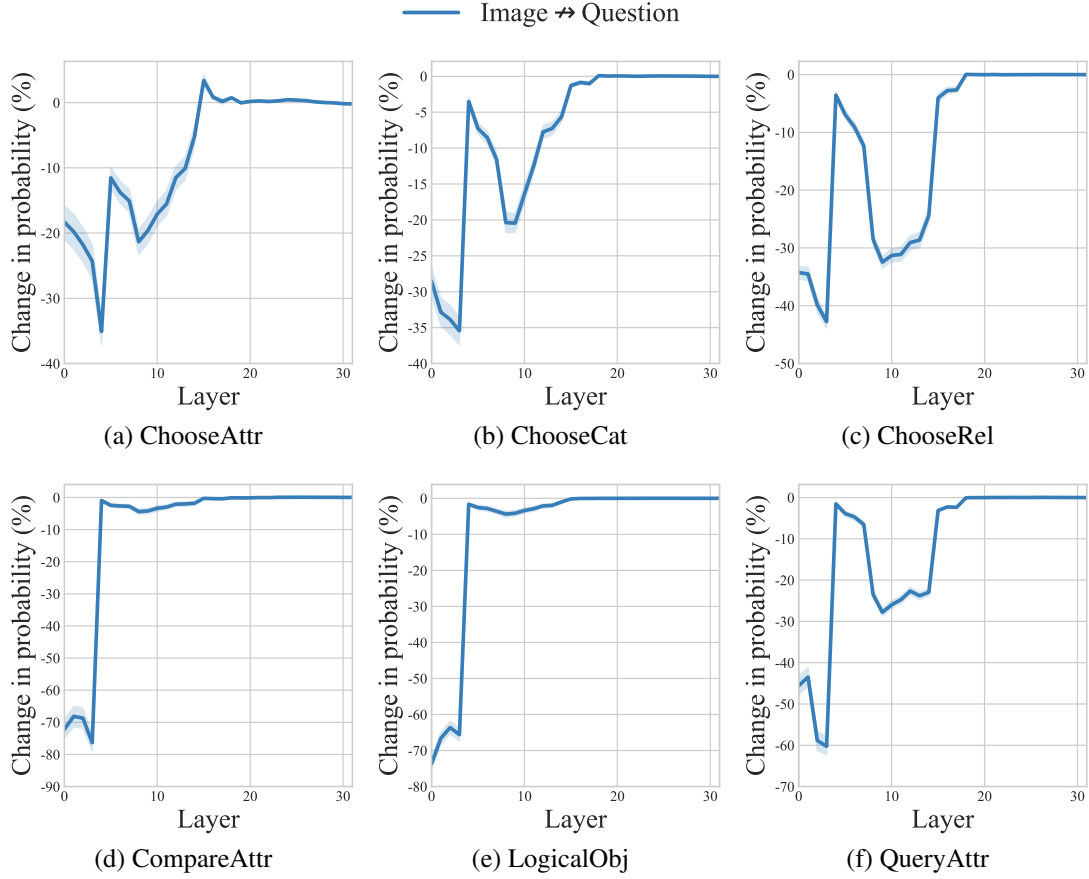


Figure 33: The relative changes in prediction probability when blocking attention edges from the *question* positions to the *image* positions on *LLaVA-v1.6-Vicuna-7b* with six VQA tasks.

from *Related Image Patches* to *question* and in following layers, information flow from *Related Image Patches* to *question* are more notable than that from *Other Image Patches* to *question*. This observation indicates that the model still has a two-stage multimodal information integration process. Specifically, in the first stage, the model focuses on generating holistic representations of the entire image. In the second stage, it refines these representations to align them more closely with the specific given question. Subsequently, in the middle layers, the critical multimodal information flows from the *question* positions to the *last position* for the final prediction. Moreover, the probability changes for the *Capitalized Answer* across all layers, as illustrated in Figure 39, align closely with the results in the main body of the paper while no such pattern is observed for the *Noncapitalized Answer*. This suggests that the model generates the syntactically correct answer directly, without a distinct intermediate step of semantic generation followed by syntactic correction. A potential explanation for this behavior is that when *Llama3* generates an answer to a given question, it first outputs a “\n” token, which

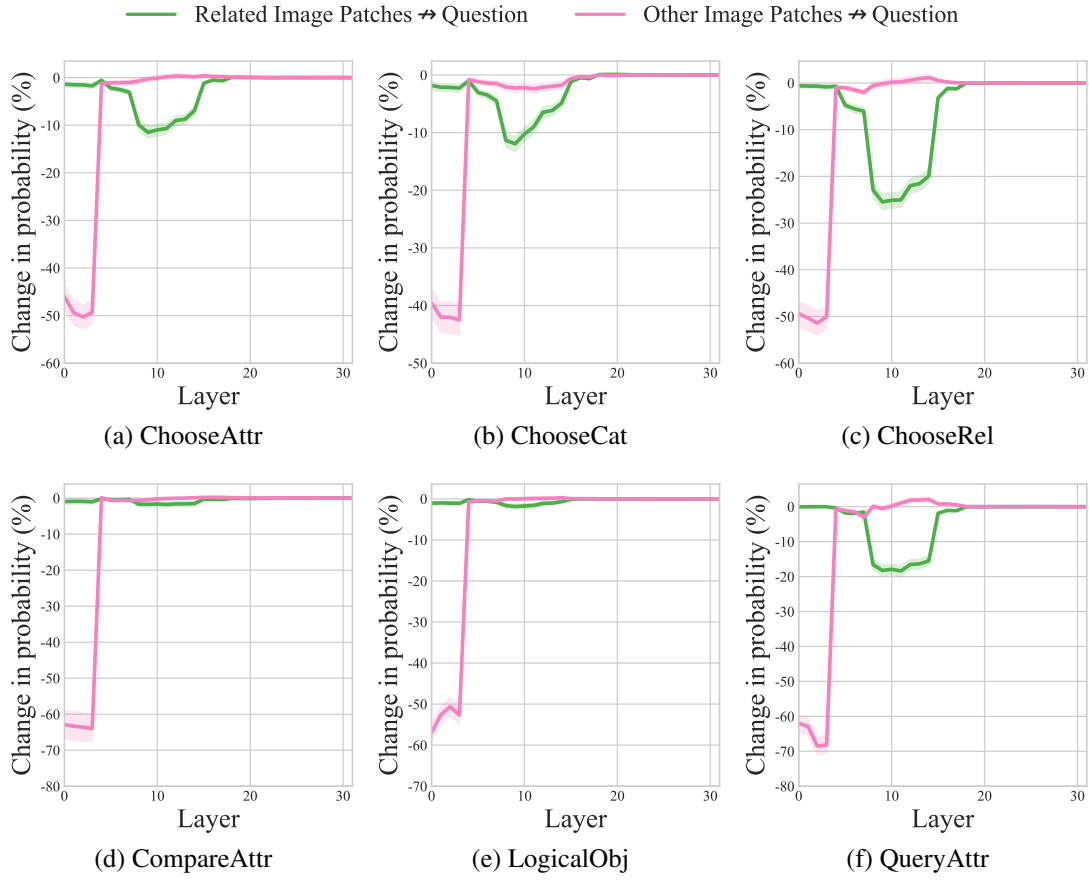


Figure 34: The relative changes in prediction probability on *LLaVA-v1.6-Vicuna-7b* with six VQA tasks. *Related Image Patches → question* and *Other Image Patches → question* represent blocking the position of *question* from attending to that of different image patches, region of interest and remainder, respectively.

may act as a cue to produce an answer word starting with an uppercase letter.

E.6 The fine-grain analysis for information flow

In the main body of the paper, we primarily focus on analyzing the information flow between one specific combination of (*image*, *question*, and *last position*) for analyzing the multimodal information integration. In this section, we will further investigate the information flow between fine-grain parts of the input sequence, including the *question without options*, *true option*, *false option*, *objects in the question*, *question without objects*, *related image patches* and *other image patches*. We also use the same *attention knockout* method to block the attention edge between them to investigate the information flow between them.

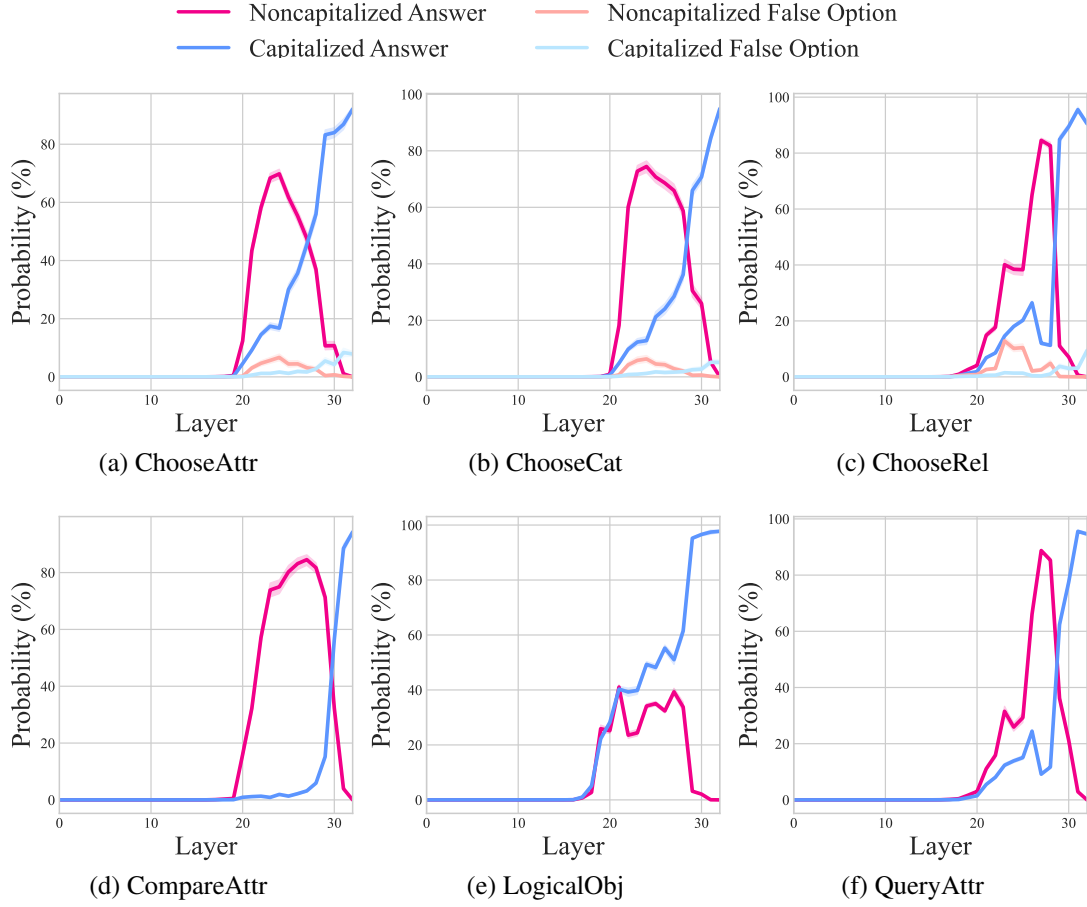


Figure 35: The probability of the answer word at the *last position* across all layers in LLaVA-v1.6-Vicuna-7b with six VQA tasks. *Capitalized Answer* and *Noncapitalized Answer* represent the answer word with or without the uppercase of the initial letter, respectively. As the tasks of *ChooseAttr*, *ChooseCat* and *ChooseRel* contain *false option*, we also provide the probability of it.

Different parts of the question to the last position

In the tasks of *ChooseAttr*, *ChooseCat* and *ChooseRel*, for each layer ℓ , we block *last position* from attending to different parts of *question*, including *question without options*, *true option*, *false option*, with the same window size ($k = 9$) around the ℓ -th layer and observe the change in the probability of the answer word at the *last position*. In the tasks of *CompareAttr*, *LogicalObj* and *QueryAttr*, we conduct the same operations with the above tasks except for blocking *last position* from attending to *objects* or *question without objects* as these tasks do not contain *options* in the question.

As shown in Figure 40 (a), (b) and (c), for the tasks of *ChooseAttr*, *ChooseCat* and *ChooseRel*, the *true option* and *false option* flowing the information to the *last position* occur in similar layers (higher layers) in the model. When blocking *last position* from

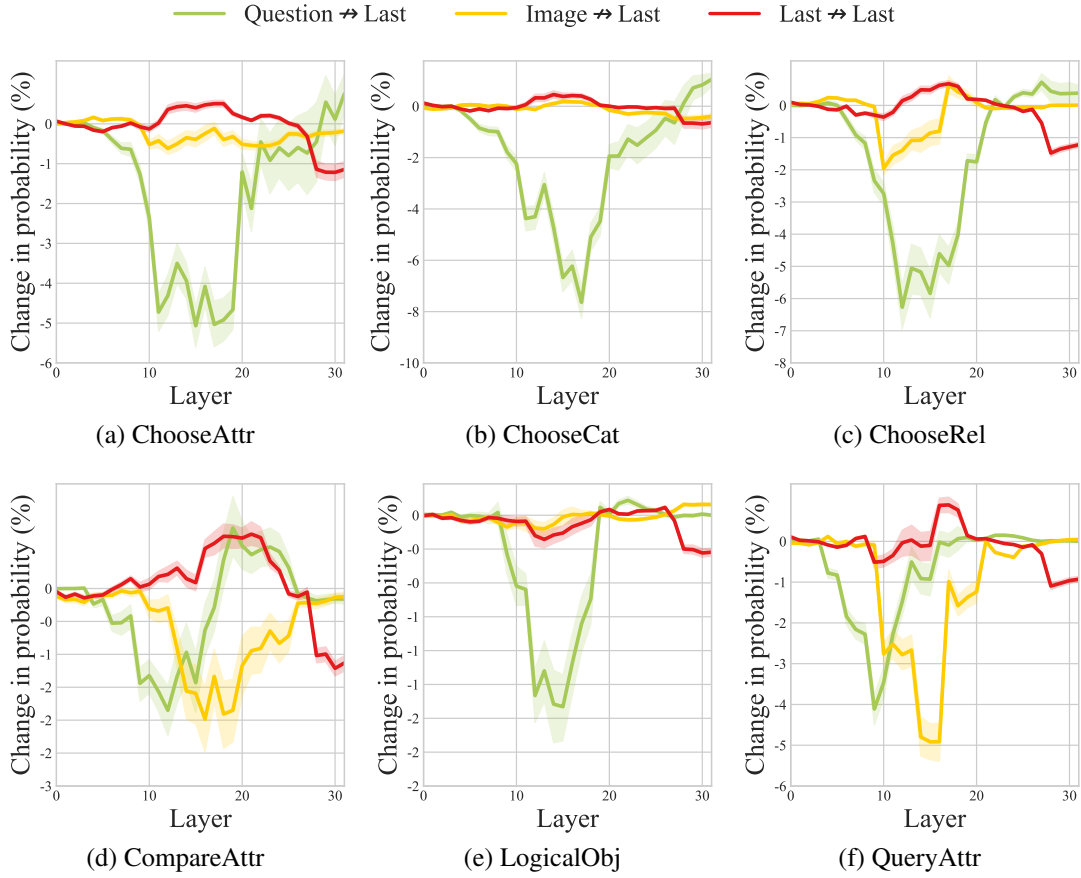


Figure 36: The relative changes in prediction probability on *llama3-llava-next-8b* with six VQA tasks. The *Question \rightarrow Last*, *Image \rightarrow Last* and *Last \rightarrow Last* represent preventing *last position* from attending to *Question*, *Image* and itself respectively.

attending *true option*, the probability obtain a reduction, while blocking *last position* from attending *false option* increases the probability of the correct answer word. The increase is reasonable because the question without the false option becomes easy for the modal. For the tasks of *ChooseAttr* and *ChooseCat*, in the information flowing to *last position*, the *options* play a dominant role while *question without options* only results in a small reduction for the probability fo the correct answer word. In contrast, for the *ChooseRel* task, the *true option* does not significantly reduce the probability of the correct answer word. This may stem from the format of the *ChooseRel* questions, where the options are positioned in the middle of the question, rather than at the end as in the *ChooseAttr* and *ChooseCat* tasks. As a result, the options in *ChooseRel* are less effective at aggregating the complete contextual information of the question within an auto-regressive transformer decoder. Consequently, the flow of information from the *option* to the *final position* becomes less critical in determining the correct answer.

As the questions in our dataset target one or more specific objects in the image,

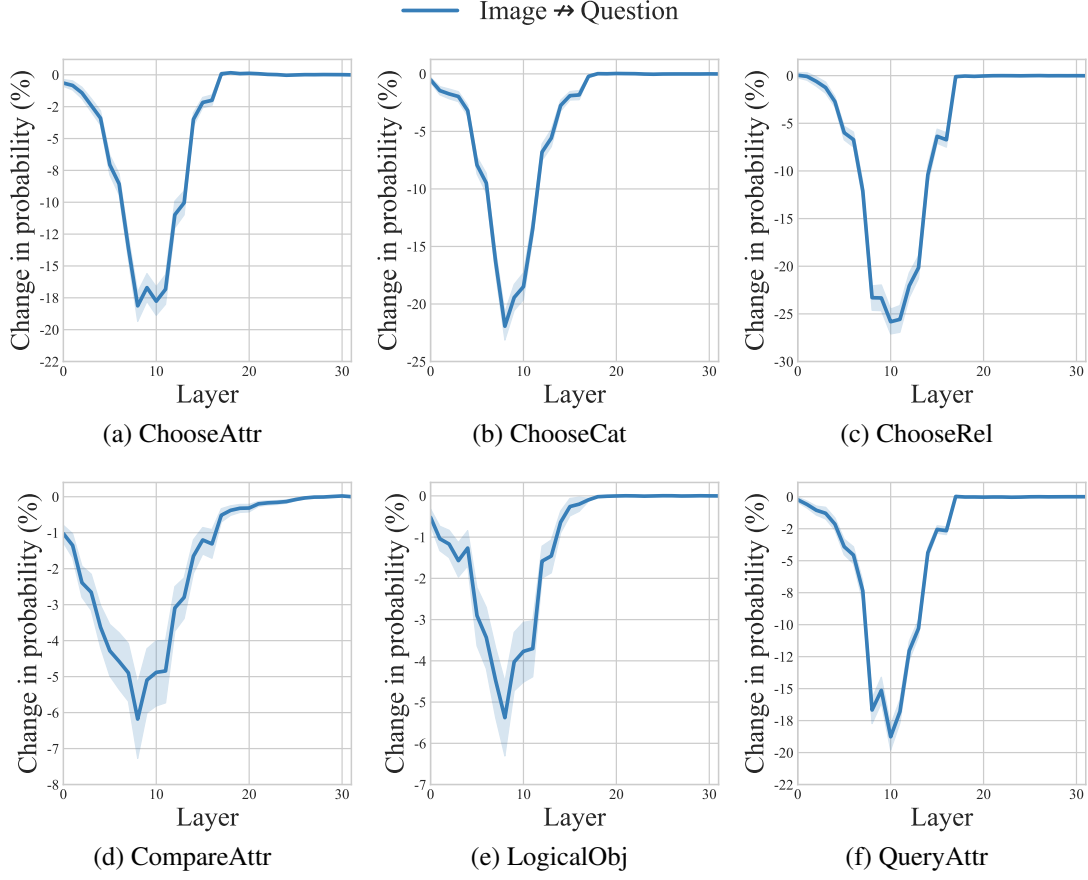


Figure 37: The relative changes in prediction probability when blocking attention edges from the *question* positions to the *image* positions on *llama3-llava-next-8b* with six VQA tasks.

we also conduct experiments on blocking *last position* from attending to *objects* or *question without objects*. As shown in Figure 40 (d), (e) and (f), the critical information from the *objects* does not directly transfer into the *last position* compared to that from *question without objects* to *last position*. This implies that the *objects* might affect the final prediction in an indirect way.

Different parts of the *question* to different parts of the *question*

In the tasks of *ChooseAttr*, *ChooseCat* and *ChooseRel*, for each layer ℓ , we block *options* from attending to *question without options* with the same window size ($k = 9$) around the ℓ -th layer and observe the change in the probability of the answer word. In the tasks of *CompareAttr* and *LogicalObj*, we conduct the same operations with the above tasks except for blocking *objects* from attending to *question without objects*.

As shown in Figure 41 (a), (b) and (c), for the tasks of *ChooseAttr*, *ChooseCat*

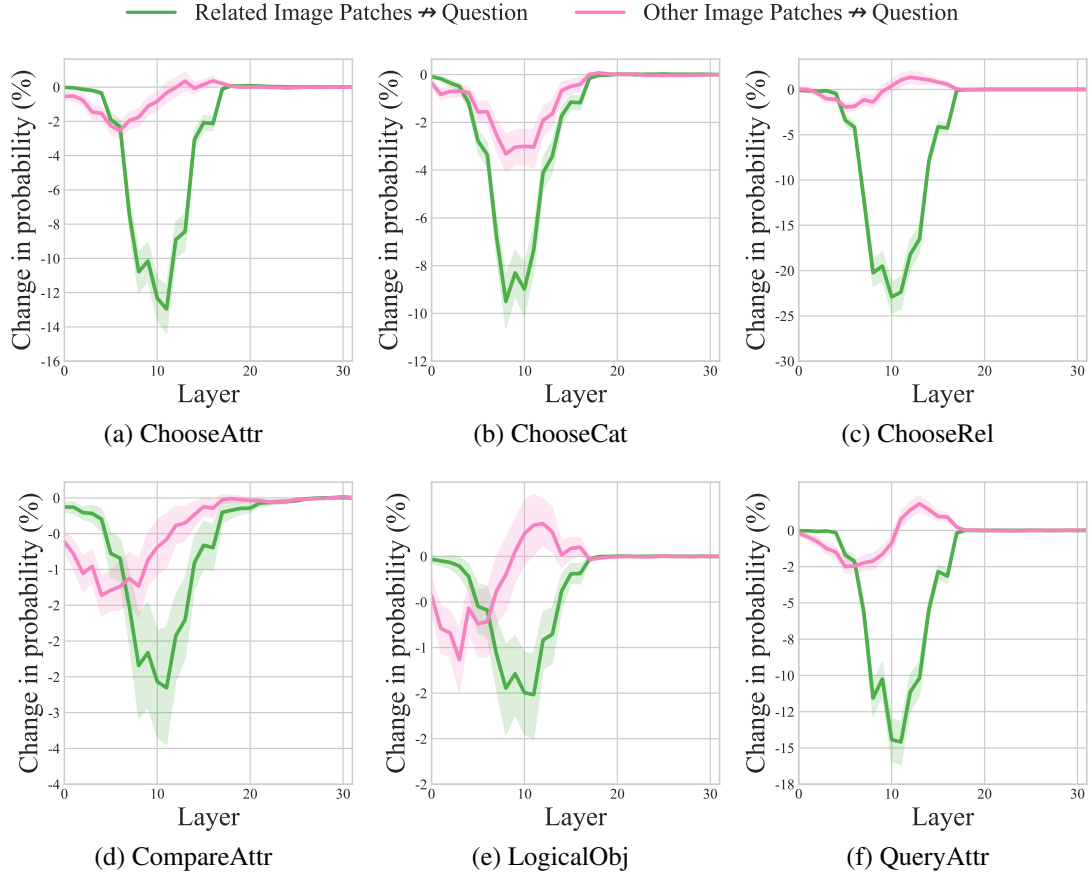


Figure 38: The relative changes in prediction probability on *llama3-llava-next-8b* with six VQA tasks. *Related Image Patches → question* and *Other Image Patches → question* represent blocking the position of *question* from attending to that of different image patches, region of interest and remainder, respectively.

and *ChooseRel*, the information flow from *question without options* to *true option* occurs in similar transformer layers with that from *question without options* to *false option*. We also observe that these indirect information flows from *question without options* to *false option* occur before the information flow from *options* to *last position* as shown in Figure 40. This indicates that the information of the question is aggregated into the *options* in lower layers and then the information in *options* is transferred to the *last position* for the prediction of the final answer in higher layers. For the tasks of *CompareAttr* and *LogicalObj*, we observe that the information flow from *question without objects* to *objects* occurs in lower layers.

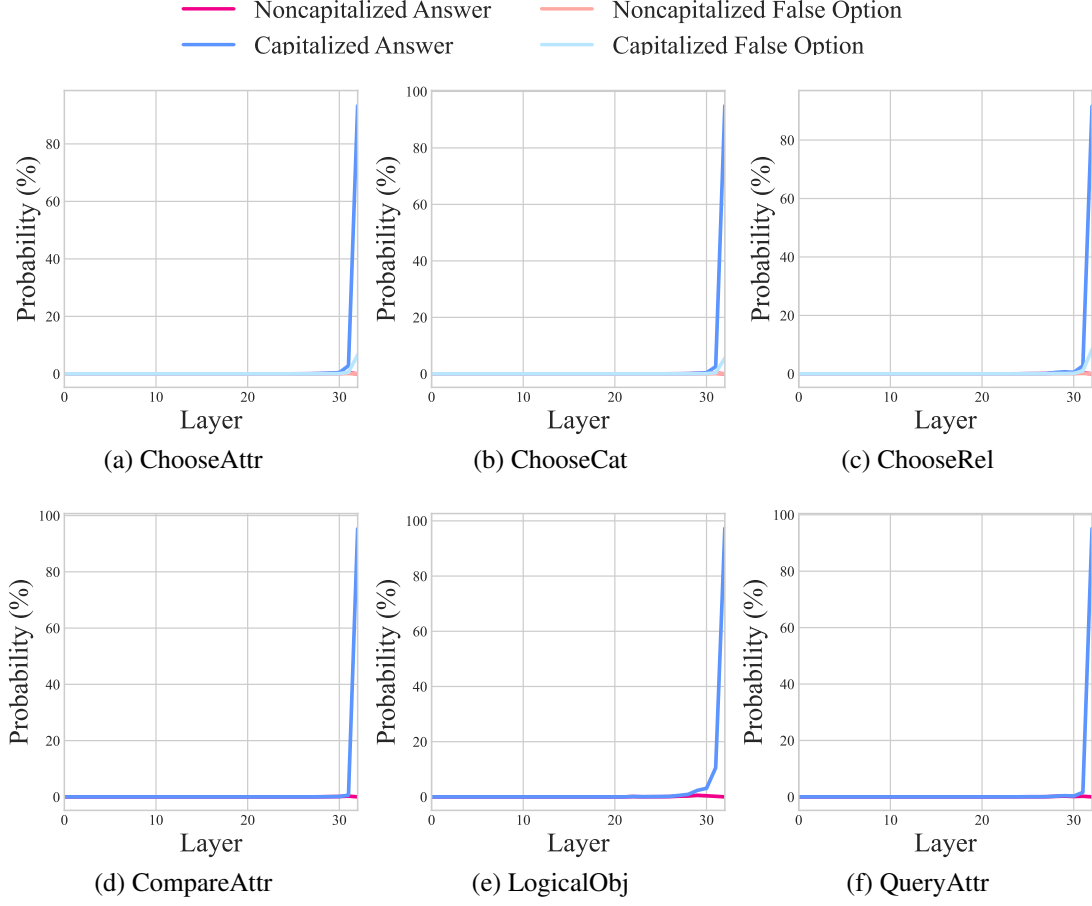


Figure 39: The probability of the answer word at the *last position* across all layers in llama3-llava-next-8b with six VQA tasks. *Capitalized Answer* and *Noncapitalized Answer* represent the answer word with or without the uppercase of the initial letter, respectively. As the tasks of *ChooseAttr*, *ChooseCat* and *ChooseRel* contain *false option*, we also provide the probability of it.

Image to different parts of question

In the tasks of *ChooseAttr*, *ChooseCat* and *ChooseRel*, for each layer ℓ , we block attention edge between *image* and different parts of *question*, including *question without options*, *true option* and *false option*, with the same window size ($k = 9$) around the ℓ -th layer and observe the change in the probability of the answer word. In the tasks of *CompareAttr*, *LogicalObj* and *QueryAttr*, we carry out the same operations as in the above tasks except for blocking the edge of the attention between *image* and *question without objects* or *objects* respectively.

As illustrated in Figure 42, the overall information flow from *image* to different parts of the *question* aligns consistently with the information flow from the *image* to the entire *question*, as depicted in Figure 7.4 in the main body of the paper. Specifically, two

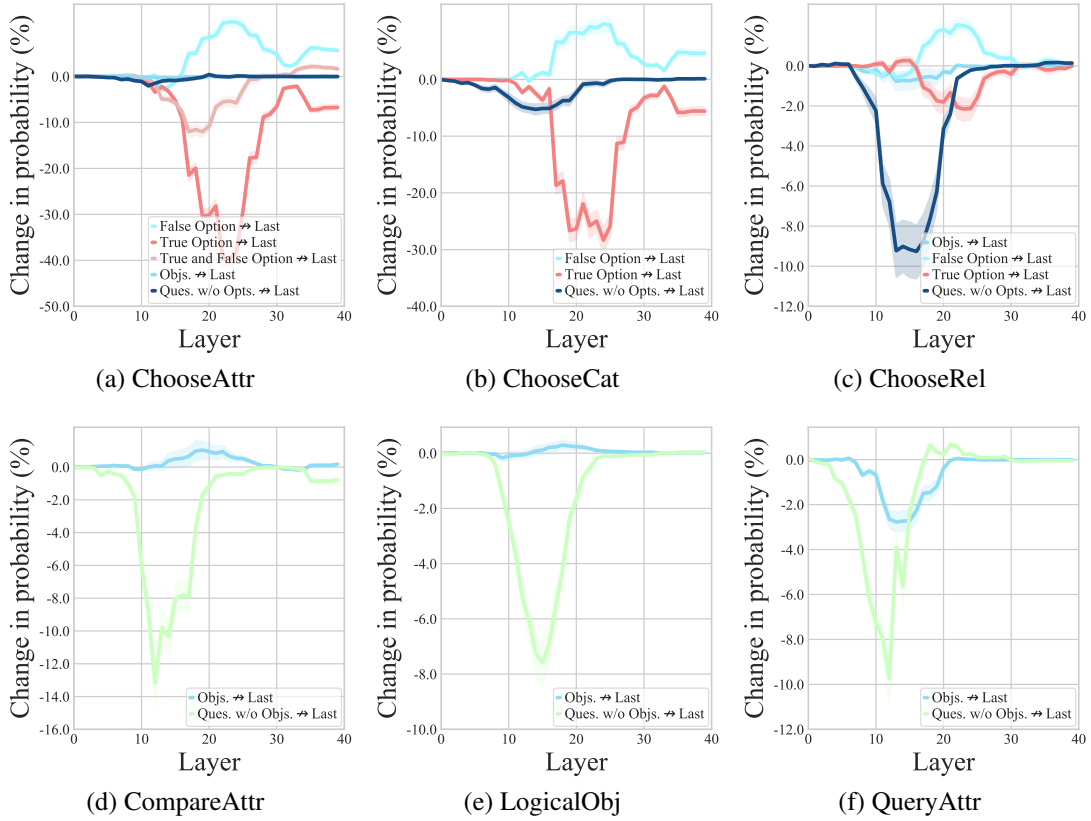


Figure 40: The relative changes in prediction probability on *LLaVA-v1.5-13b* with six VQA tasks. Preventing *Last Position* from attending to different parts of *Question*, such as *True Option*, *False Option*, *Objects* in question, *Question without Options*, *Question without Objects*, both *NTrue Option* and *False Option* together.

distinct flows are from the *image* to the *question*. Notably, however, different parts of the *question* exhibit varying magnitudes of probability change, especially in the second-time drop in probability, which may be because different kinds of questions have different attention patterns to the image. For example, during the second-time drop in probability, in the tasks of *ChooseAttr* and *ChooseCat*, the *image* information does not transfer to *false option* while it transfers much more information to *true option*. However, this pattern isn't observed in the task of *ChooseRel*, where most *image* information is transferred into *question without options* and *objects*.

Other image patches to different parts of question

In the tasks of *ChooseAttr*, *ChooseCat* and *ChooseRel*, for each layer ℓ , we block attention edge between *other image patches* and different parts of *question*, including *question without options*, *true option*, *false option*, *objects* and *question without objects*, with the same window size ($k = 9$) around the ℓ -th layer and observe the change in the

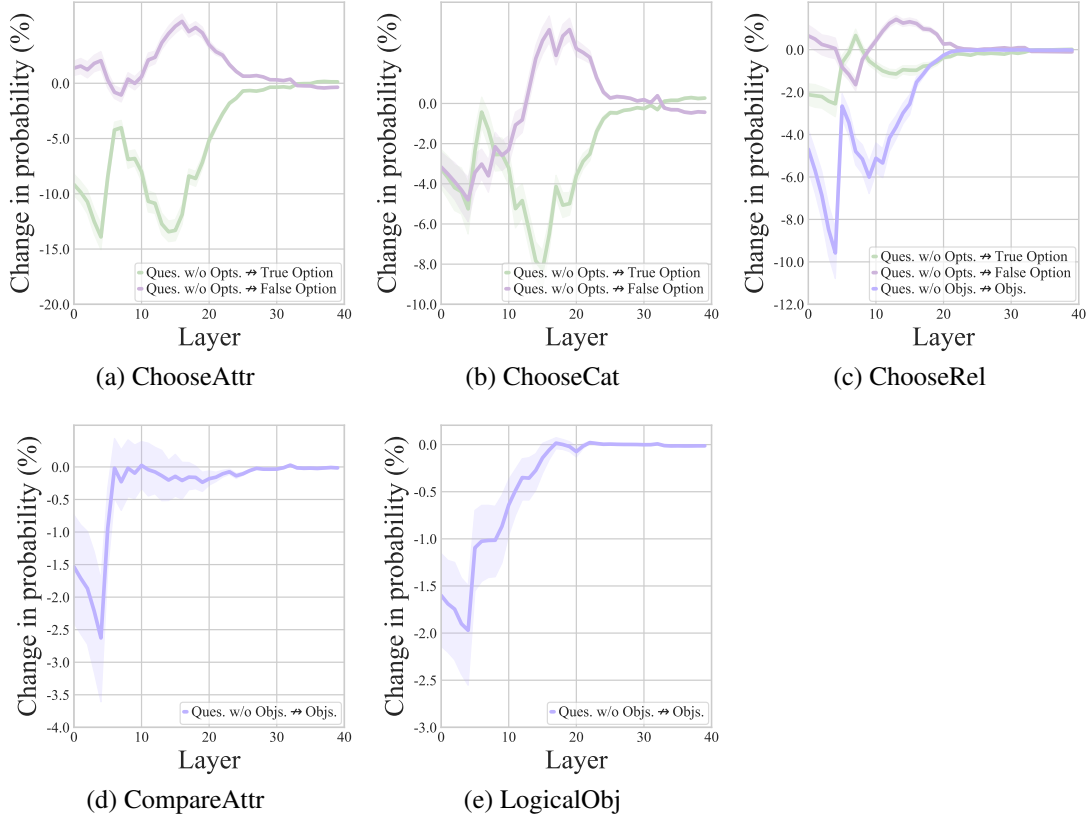


Figure 41: The relative changes in prediction probability on *LLaVA-v1.5-13b* with six VQA tasks. Preventing information flow from *Question without Option* to *Options* and from *Question without Objects* to *Objects*.

probability of the answer word. In the tasks of *CompareAttr*, *LogicalObj* and *QueryAttr*, we conduct the same operations with the above tasks except for blocking attention edge between *other image patches* and *question without objects* or *objects* respectively.

As shown in Figure 43, the information flow from *other image patches* to different parts of the *question* for all six tasks consistently aligns the flow observed from *other image patches* to the entire *question*, as illustrated in Figure 7.5 in the main body of the paper. Specifically, the information flow dominantly occurs in the first-time drop in the probability in the lower layers, regardless of which part of the *question* is being blocked.

Related image patches to different parts of question

In the tasks of *ChooseAttr*, *ChooseCat* and *ChooseRel*, for each layer ℓ , we block the attention edge between *Related image patches* and different parts of *question*, including *question without options*, *true option*, *false option*, *objects* and *question without objects*, with the same window size ($k = 9$) around the ℓ -th layer and observe the change

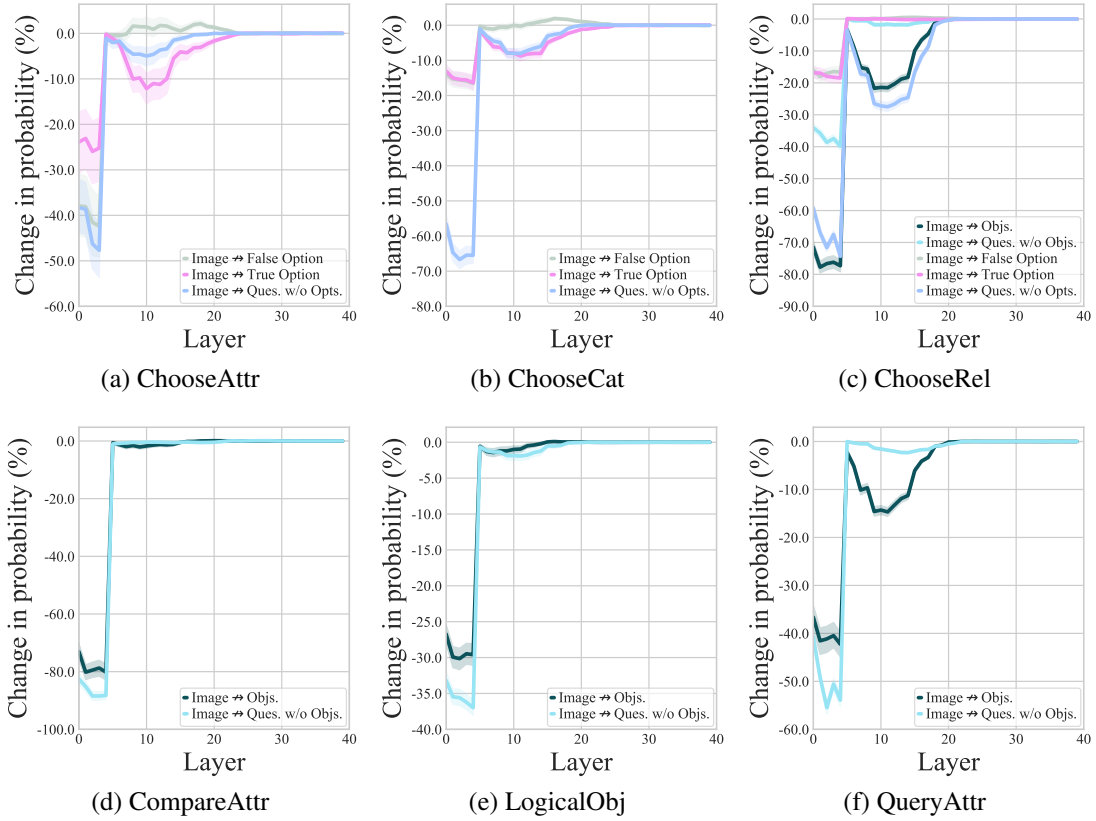


Figure 42: The relative changes in prediction probability on *LLaVA-v1.5-13b* with six VQA tasks. Blocking the information flow from *Image* to different parts of the *question*, including *True Option*, *False Option*, *Objects* in question, *Question without Objects*, *Question without Options*.

in the probability of the answer word. In the tasks of *CompareAttr*, *LogicalObj* and *QueryAttr*, we conduct the same operations with the above tasks except for blocking the attention edge between *Related image patches* and *question without objects* or *objects* respectively.

The observations of the overall information flow from *related image patches* to different parts of the *question* for all six tasks shown in Figure 44 consistently align the flow observed from *related image patches* to the entire *question*, as illustrated in Figure 7.5 in the main body of the paper. Specifically, the information flow dominantly occurs in the second-time drop in the probability in the lower-to-middle layers (around 10th). However, there are some parts of *question* that don't obtain the information followed from the *related image patches*. For example, the *objects* in the task of *ChooseCat*, or *false option* and *true option* in the task of *ChooseRel*.

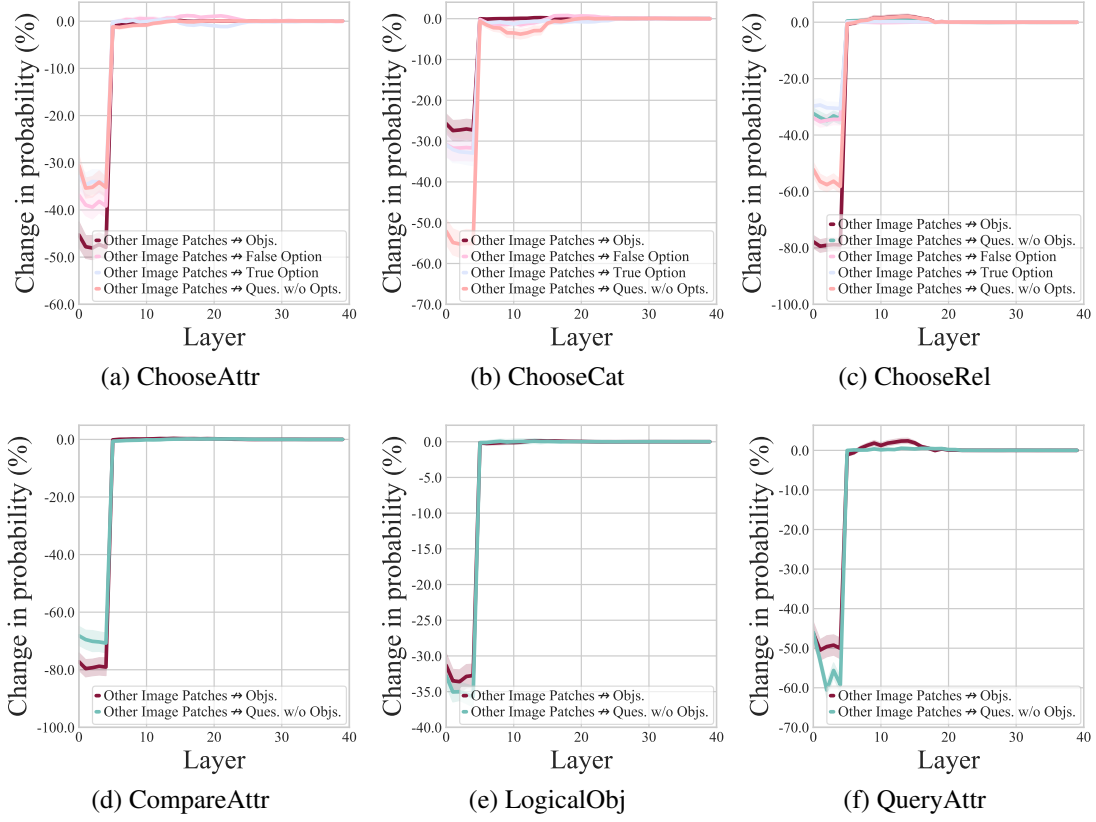


Figure 43: The relative changes in prediction probability on *LLaVA-v1.5-13b* with six VQA tasks. Blocking the information flow from *Other Image Patches* to different parts of the question, including *True Option*, *False Option*, *Objects in question*, *Question without Objects*, *Question without Options*.

E.7 The influence of *images* on the semantics of *Questions*

We already know that the *image* information is integrated into the representation corresponding to the position of *question*. To investigate whether the *image* affects the final semantics of the *question*, for each layer ℓ , we prevent the *question* from attending to the *question*, with the same window size ($k = 9$) around the ℓ -th layer and observe the change of semantics of the *question* in the final layer. The semantics of the *question* is evaluated by the *Jaccard Similarity* as in Appendix E.4.

As illustrated in Figure 45, the *Jaccard Similarity* demonstrates a significant decline in the lower layers, resembling the behaviour observed in layers where information flows from the *image* to the *question*. This pattern highlights the critical role of *image* information in constructing the final multimodal semantic representation.

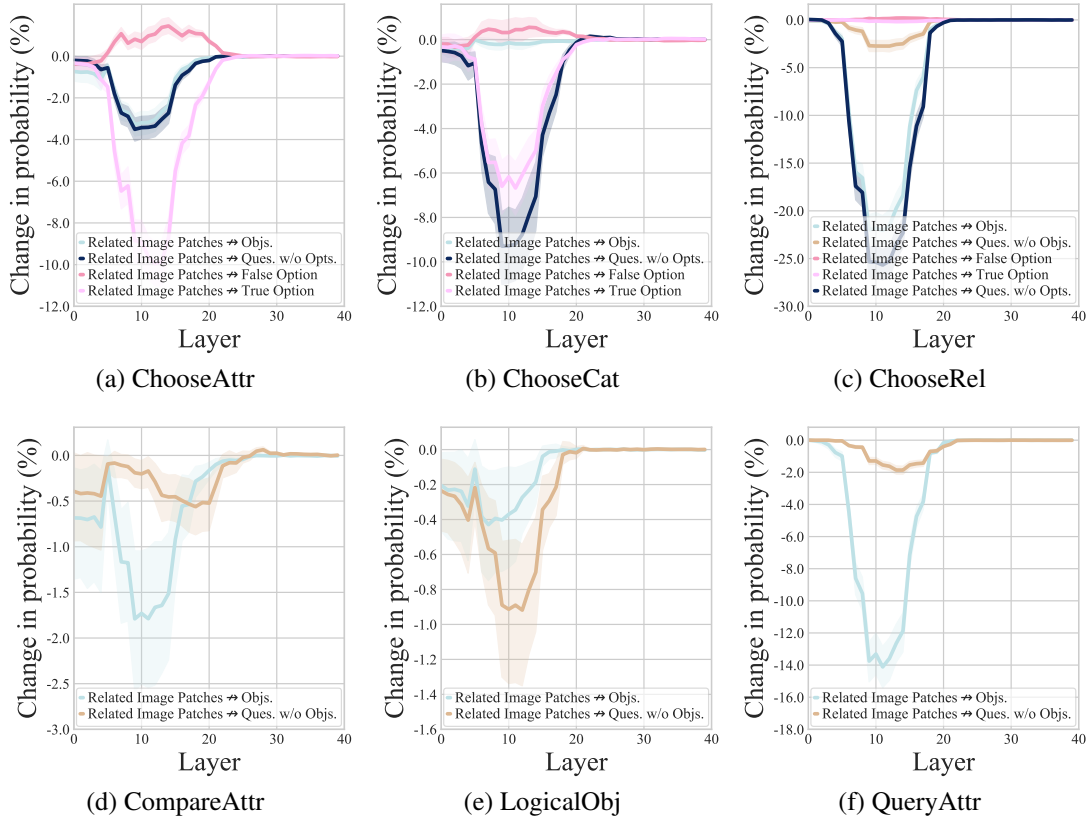


Figure 44: The relative changes in prediction probability on *LLaVA-v1.5-13b* with six VQA tasks. Blocking the information flow from *Related Image Patches* to different parts of the question, including *True Option*, *False Option*, *Objects in question*, *Question without Objects*, *Question without Options*.

E.8 Difference with unimodal LLM

To compare the information flow in LLMs and MLLMs, we replace the image with dense captions (obtained from VG dataset (Krishna et al., 2017)) and analyze the resulting information flow. As shown in Figure 46 (first two figures), we observe that information flow from the question to last position primarily occurs in middle layers which is consistent with MLLMs. However, in LLMs, information flow from dense captions to the last position in lower layers differs from MLLMs where almost no information flow is observed from the image. This suggests that captions and questions follow distinct processing stages, aligning with findings from Geva et al. (2023), which identify separate stages of information flow in attribute extraction tasks within LLMs.

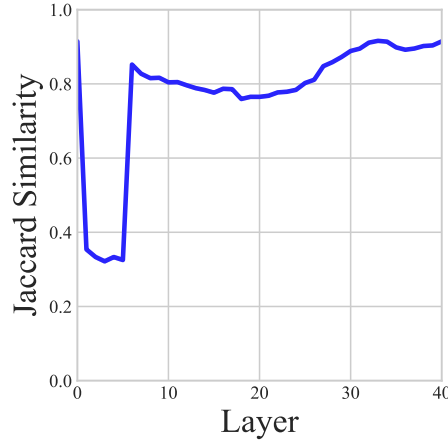


Figure 45: The Jaccard similarity between the predicted words of the original model *LLaVA-1.5-13b* and those of the intervened model blocking *question* from attending to *image* on the task of *ChooseAttr*.

E.9 More complex tasks

Since the main focus of our paper is the interaction between modalities, we evaluate tasks where cross-modal alignment plays a crucial role, while excluding those requiring external knowledge for reasoning to minimize confounding factors and ensure a more precise analysis. Nevertheless, to further validate our findings, we extend our experiments to two additional factual multimodal tasks, OKVQA (Marino et al., 2019) and AOKVQA (Schwenk et al., 2022) involving more complex, fact-based reasoning requiring external commonsense and world knowledge. Figure 47 shows the results align with our findings, *i.e.* from lower to higher layers: a two-stage multimodal integration, information flow from question to last positions and semantic generation and syntactic refinement.

E.10 Attention sink

The work in (Darcet et al., 2024) shows high-norm image patch tokens hold global rather than local information, we analyze the distribution of token norms over our dataset and identify a threshold of 57, as shown in Figure 48. Excluding patches exceeding 57 (3.3 out of 576 patches per image on average), we split the remaining patches into two groups and replicate the experiments from our paper (Section 7.6). As shown in Figure 49, the results across six tasks remain consistent with our original findings.

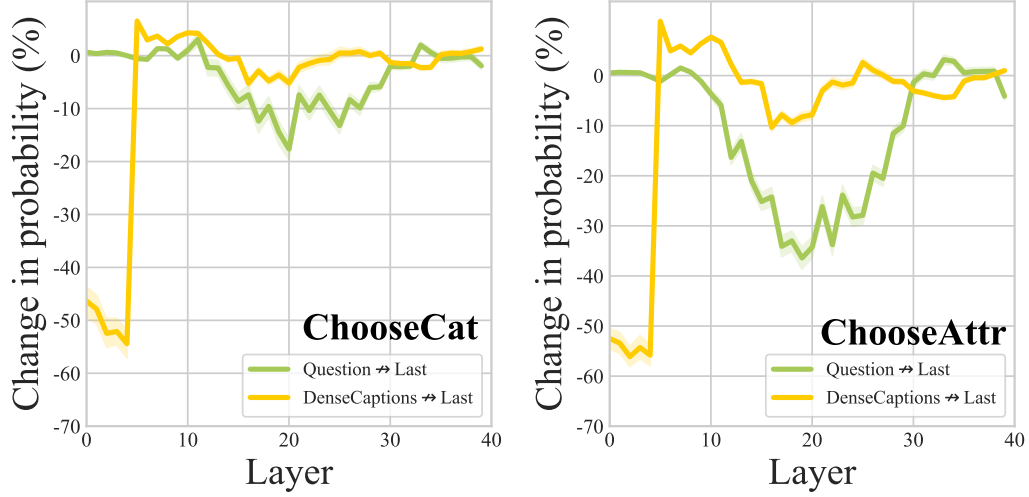


Figure 46: The information flow from dense captions or questions to the last position in unimodal LLM: *vicuna1.5-13b*, the initial LLM in *LLaVA1.5-13b* (MLLM). They are tested on *ChooseCat* (left) and *ChooseAttr* (right) tasks.

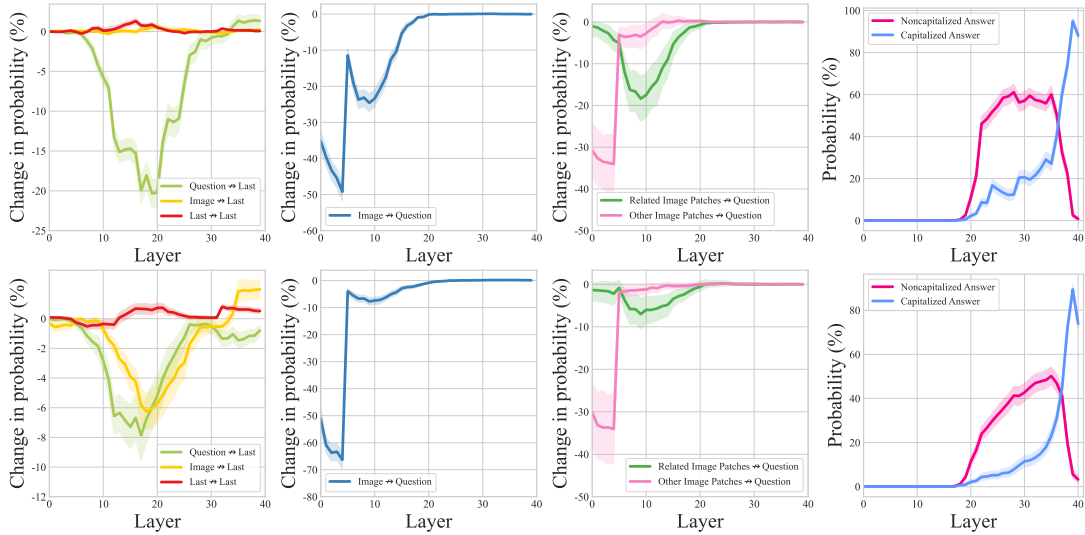


Figure 47: The information flow and answer word tracking on factual datasets: AOKVQA (first line), OKVQA (second line)

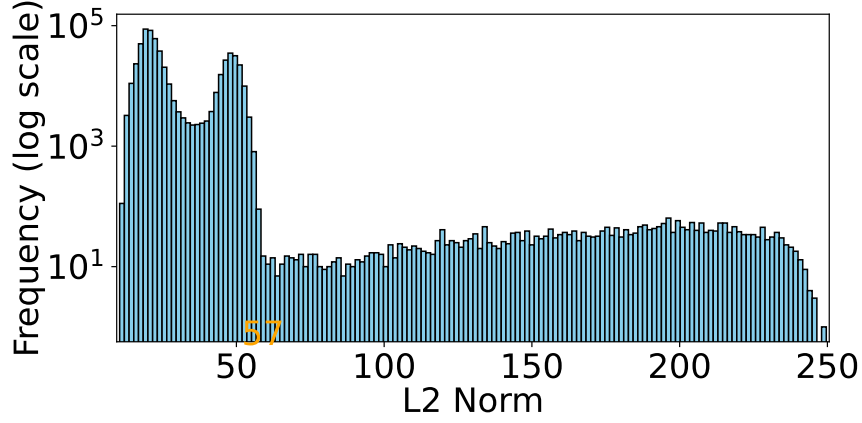


Figure 48: Distribution of image patch norms.

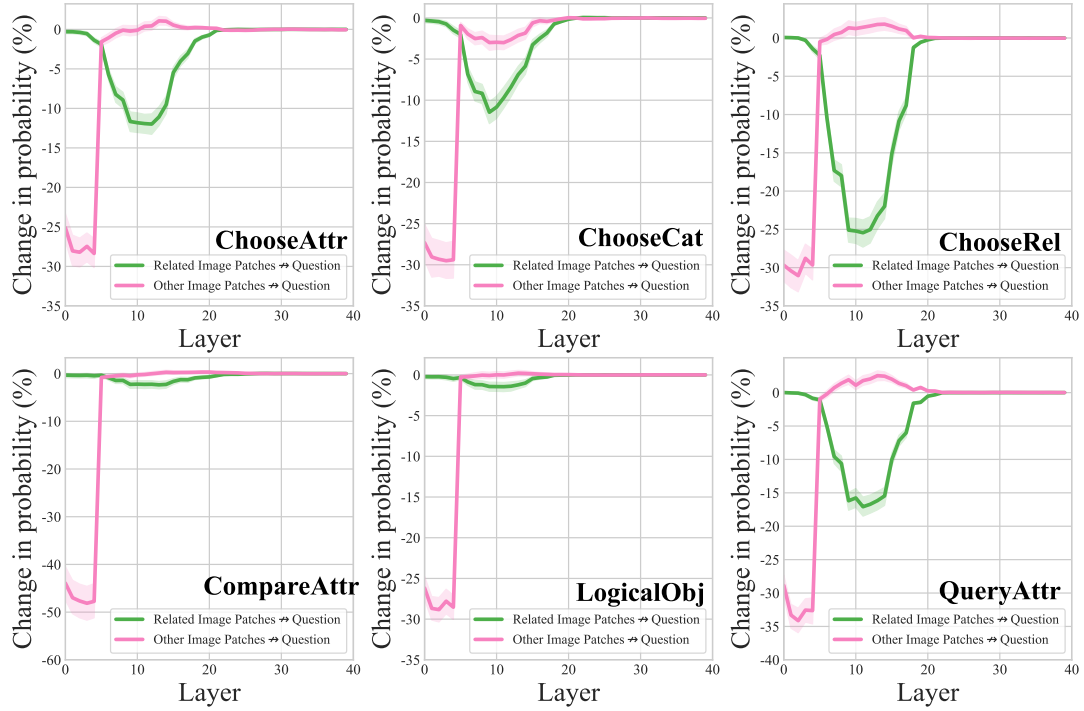


Figure 49: Information flow from image patches (related/other), excluding those with high norms, to question position on six VQA tasks.

Bibliography

2024. Imms-lab/llama3-llava-next-8b ‘ hugging face. <https://huggingface.co/lmms-lab/llama3-llava-next-8b>. Accessed: 2024-11-13.
- Amina Adadi. 2021. A survey on data-efficient algorithms in big data era. *Journal of Big Data*, 8(1):24.
- Estelle Aflalo, Meng Du, Shao-Yen Tseng, Yongfei Liu, Chenfei Wu, Nan Duan, and Vasudev Lal. 2022. VI-interpret: An interactive visualization tool for interpreting vision-language transformers. In *Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition*, pages 21406–21415.
- Rishabh Agarwal, Levi Melnick, Nicholas Frosst, Xuezhou Zhang, Ben Lengerich, Rich Caruana, and Geoffrey E Hinton. 2021. Neural additive models: Interpretable machine learning with neural nets. *Advances in neural information processing systems*, 34:4699–4711.
- Arash Ahmadian, Saurabh Dash, Hongyu Chen, Bharat Venkitesh, Zhen Stephen Gou, Phil Blunsom, Ahmet Üstün, and Sara Hooker. 2023. Intriguing properties of quantization at scale. *Advances in Neural Information Processing Systems*, 36:34278–34294.
- Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebrón, and Sumit Sanghai. 2023. Gqa: Training generalized multi-query transformer models from multi-head checkpoints. *arXiv preprint arXiv:2305.13245*.
- Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. 2022. Flamingo: a visual language model for few-shot learning. *Advances in Neural Information Processing Systems*, 35:23716–23736.
- David Alvarez-Melis and Tommi S Jaakkola. 2017. A causal framework for explaining the predictions of black-box sequence-to-sequence models. *arXiv preprint arXiv:1707.01943*.

- Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. 2018. Bottom-up and top-down attention for image captioning and visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6077–6086.
- Alan Ansell, Ivan Vulić, Hannah Sterz, Anna Korhonen, and Edoardo M Ponti. 2024. Scaling sparse fine-tuning to large language models. *arXiv preprint arXiv:2401.16405*.
- Anna Arias-Duart, Victor Gimenez-Abalos, Ulises Cortés, and Dario Garcia-Gasulla. 2023. Assessing biases through visual contexts. *Electronics*, 12(14):3066.
- Shahin Atakishiyev, Mohammad Salameh, Hengshuai Yao, and Randy Goebel. 2024. Explainable artificial intelligence for autonomous driving: A comprehensive overview and field guide for future research directions. *IEEE Access*.
- Matan Avitan, Ryan Cotterell, Yoav Goldberg, and Shauli Ravfogel. 2024. What changed? converting representational interventions to natural language. *arXiv e-prints*, pages arXiv–2402.
- Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.
- Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. 2017. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 39(12):2481–2495.
- Nooshin Bahador. 2025. Mechanistic interpretability of fine-tuned vision transformers on distorted images: Decoding attention head behavior for transparent and trustworthy ai. *arXiv preprint arXiv:2503.18762*.
- Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. 2023a. [Qwen-VL: A Versatile Vision-Language Model for Understanding, Localization, Text Reading, and Beyond](#). ArXiv:2308.12966 [cs].
- Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. 2023b. [Qwen-vl: A versatile vision-language model for understanding, localization, text reading, and beyond](#).
- Sriram Balasubramanian, Samyadeep Basu, and Soheil Feizi. 2024. Decomposing and interpreting image representations via text in vits beyond clip. *arXiv preprint arXiv:2406.01583*.
- Yamini Bansal, Preetum Nakkiran, and Boaz Barak. 2021. Revisiting model stitching to compare neural representations. *Advances in neural information processing systems*, 34:225–236.

- Hang Bao, Li Dong, and Furu Wei. 2021. Beit: Bert pre-training of image transformers. In *International Conference on Learning Representations (ICLR)*.
- Ankur Bapna, Naveen Arivazhagan, and Orhan Firat. 2019. Simple, scalable adaptation for neural machine translation. *arXiv preprint arXiv:1909.08478*.
- Brian R Bartoldson, Bhavya Kailkhura, and Davis Blalock. 2023. Compute-efficient deep learning: Algorithmic trends and opportunities. *Journal of Machine Learning Research*, 24(122):1–77.
- Samyadeep Basu, Martin Grayson, Cecily Morrison, Besmira Nushi, Soheil Feizi, and Daniela Massiceti. 2024. Understanding information storage and transfer in multi-modal large language models. *arXiv preprint arXiv:2406.04236*.
- David Bau, Bolei Zhou, Aditya Khosla, Aude Oliva, and Antonio Torralba. 2017. Network dissection: Quantifying interpretability of deep visual representations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6541–6549.
- Charles Beattie, Joel Z Leibo, Denis Teplyashin, Tom Ward, Marcus Wainwright, Heinrich Küttler, Andrew Lefrancq, Simon Green, Víctor Valdés, Amir Sadik, et al. 2016. Deepmind lab. *arXiv preprint arXiv:1612.03801*.
- Nora Belrose, Zach Furman, Logan Smith, Danny Halawi, Igor Ostrovsky, Lev McKinney, Stella Biderman, and Jacob Steinhardt. 2023a. Eliciting latent predictions from transformers with the tuned lens. *arXiv preprint arXiv:2303.08112*.
- Nora Belrose, David Schneider-Joseph, Shauli Ravfogel, Ryan Cotterell, Edward Raff, and Stella Biderman. 2023b. Leace: Perfect linear concept erasure in closed form. *Advances in Neural Information Processing Systems*, 36:66044–66063.
- Elad Ben Zaken, Yoav Goldberg, and Shauli Ravfogel. 2022. [BitFit: Simple Parameter-efficient Fine-tuning for Transformer-based Masked Language-models](#). In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 1–9, Dublin, Ireland. Association for Computational Linguistics.
- Leonard Bereska and Efstratios Gavves. 2024. [Mechanistic Interpretability for AI Safety – A Review](#). ArXiv:2404.14082.
- Kartikeya Bhardwaj, Nilesh Pandey, Sweta Priyadarshi, Viswanath Ganapathy, Shreya Kadambi, Rafael Esteves, Shubhankar Borse, Paul Whatmough, Risheek Garrepalli, Mart van Baalen, et al. 2024. Sparse high rank adapters. *Advances in Neural Information Processing Systems*, 37:13685–13715.

- Ning Bian, Xianpei Han, Bo Chen, and Le Sun. 2021. Benchmarking knowledge-enhanced commonsense question answering via knowledge-to-text transformation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 12574–12582.
- Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, et al. 2020. [PIQA: Reasoning about physical commonsense in natural language](#). In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7432–7439.
- Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.
- Collin Burns, Haotian Ye, Dan Klein, and Jacob Steinhardt. 2022. Discovering latent knowledge in language models without supervision. *arXiv preprint arXiv:2212.03827*.
- Daniele Calandriello, Alessandro Lazaric, and Marcello Restelli. 2014. Sparse multi-task reinforcement learning. *Advances in neural information processing systems*, 27.
- Jiezhong Cao, Jincheng Li, Xiping Hu, Xiangmiao Wu, and Minghui Tan. 2022. Towards interpreting deep neural networks via layer behavior understanding. *Machine Learning*, 111:1159–1179.
- Jize Cao, Zhe Gan, Yu Cheng, Licheng Yu, Yen Chun Chen, and Jingjing Liu. 2020. [Behind the Scene: Revealing the Secrets of Pre-trained Vision-and-Language Models](#). *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 12351 LNCS:565–580. ArXiv: 2005.07310 ISBN: 9783030585389.
- Nicolas Carion et al. 2020. End-to-end object detection with transformers. In *European Conference on Computer Vision (ECCV)*.
- Rich Caruana, Yin Lou, Johannes Gehrke, Paul Koch, Marc Sturm, and Noemie Elhadad. 2015. Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1721–1730.
- Niladri S Chatterji, Behnam Neyshabur, and Hanie Sedghi. 2019. The intriguing role of module criticality in the generalization of deep networks. *arXiv preprint arXiv:1912.00528*.
- Hila Chefer, Shir Gur, and Lior Wolf. 2021a. [Generic Attention-model Explainability for Interpreting Bi-Modal and Encoder-Decoder Transformers](#). pages 397–406. ArXiv: 2103.15679.

- Hila Chefer, Shir Gur, and Lior Wolf. 2021b. Transformer interpretability beyond attention visualization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 782–791.
- Jun Chen, Han Guo, Kai Yi, Boyang Li, and Mohamed Elhoseiny. 2022. Visualgpt: Data-efficient adaptation of pretrained language models for image captioning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18030–18040.
- Liang Chen, Haozhe Zhao, Tianyu Liu, Shuai Bai, Junyang Lin, Chang Zhou, and Baobao Chang. 2024. [An Image is Worth 1/2 Tokens After Layer 2: Plug-and-Play Inference Acceleration for Large Vision-Language Models](#). ArXiv:2403.06764 [cs].
- Tianlong Chen, Jonathan Frankle, Shiyu Chang, Sijia Liu, Yang Zhang, Zhangyang Wang, and Michael Carbin. 2020a. The lottery ticket hypothesis for pre-trained bert networks. *ArXiv*, abs/2007.12223.
- Tianrun Chen, Lanyun Zhu, Chaotao Ding, Runlong Cao, Shangzhan Zhang, Yan Wang, Zejian Li, Lingyun Sun, Papa Mao, and Ying Zang. 2023. [Sam fails to segment anything? – sam-adapter: Adapting sam in underperformed scenes: Camouflage, shadow, and more](#).
- Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020b. A simple framework for contrastive learning of visual representations. In *Proceedings of ICML*.
- Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. 2020c. Uniter: Universal image-text representation learning. In *European conference on computer vision*, pages 104–120. Springer.
- Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. 2020d. Uniter: Universal image-text representation learning. In *European Conference on Computer Vision*, pages 104–120.
- Zhao Chen, Jiquan Ngiam, Yanping Huang, Thang Luong, Henrik Kretzschmar, Yuning Chai, and Dragomir Anguelov. 2020e. Just pick a sign: Optimizing deep multitask models with gradient sign dropout. *Advances in Neural Information Processing Systems*, 33:2039–2050.
- Gong Cheng, Junwei Han, and Xiaoqiang Lu. 2017. Remote sensing image scene classification: Benchmark and state of the art. *Proceedings of the IEEE*, 105(10):1865–1883.
- Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. In *Proceedings of EMNLP*.

- Rochelle Choenni, Dan Garrette, and Ekaterina Shutova. 2022. [Data-Efficient Cross-Lingual Transfer with Language-Specific Subnetworks](#). ArXiv:2211.00106 [cs].
- Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. 2014. Describing textures in the wild. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3606–3613.
- Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019a. [BoolQ: Exploring the surprising difficulty of natural yes/no questions](#). In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2924–2936, Minneapolis, Minnesota. Association for Computational Linguistics.
- Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D Manning. 2019b. What does bert look at? an analysis of bert’s attention. *arXiv preprint arXiv:1906.04341*.
- Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. [Think you have solved question answering? Try ARC, the AI2 reasoning challenge](#). *arXiv:1803.05457*.
- Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. 2021. [Training verifiers to solve math word problems](#). *arXiv:2110.14168*.
- Arthur Conmy, Augustine N. Mavor-Parker, Aengus Lynch, Stefan Heimersheim, and Adrià Garriga-Alonso. 2023. [Towards Automated Circuit Discovery for Mechanistic Interpretability](#). ArXiv:2304.14997 [cs].
- Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. 2016. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3213–3223.
- Camille Couprie, Clément Farabet, Laurent Najman, and Yann LeCun. 2013. [Indoor semantic segmentation using depth information](#).
- Ian Covert, Scott Lundberg, and Su-In Lee. 2021. Explaining by removing: A unified framework for model explanation. *Journal of Machine Learning Research*, 22(209):1–90.
- Y. L. Cun, J. S. Denker, and S. A. Solla. 1990. *Optimal brain damage*. Advances in neural information processing systems 2.

- Hoagy Cunningham, Aidan Ewart, Logan Riggs, Robert Huben, and Lee Sharkey. 2023. Sparse autoencoders find highly interpretable features in language models. *arXiv preprint arXiv:2309.08600*.
- Leonardo L Custode and Giovanni Iacca. 2023. Evolutionary learning of interpretable decision trees. *IEEE Access*, 11:6169–6184.
- Adam Dahlgren Lindström, Johanna Björklund, Suna Bensch, and Frank Drewes. 2020. [Probing multimodal embeddings for linguistic properties: the visual-semantic case](#). In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 730–744, Barcelona, Spain (Online). International Committee on Computational Linguistics.
- Damai Dai, Li Dong, Yaru Hao, Zhifang Sui, Baobao Chang, and Furu Wei. 2022. [Knowledge Neurons in Pretrained Transformers](#). ArXiv:2104.08696 [cs].
- Wenliang Dai, Junnan Li, Dongxu Li, Anthony Meng Huat Tiong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale Fung, and Steven Hoi. 2023. [Instruct-BLIP: Towards General-purpose Vision-Language Models with Instruction Tuning](#). ArXiv:2305.06500 [cs].
- Guy Dar, Mor Geva, Ankit Gupta, and Jonathan Berant. 2022. Analyzing transformers in embedding space. *arXiv preprint arXiv:2209.02535*.
- Timothée Darcet, Maxime Oquab, Julien Mairal, and Piotr Bojanowski. 2024. [Vision Transformers Need Registers](#). ArXiv:2309.16588 [cs].
- Joe Davison, Joshua Feldman, and Alexander Rush. 2019. [Commonsense knowledge mining from pretrained models](#). In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1173–1178, Hong Kong, China. Association for Computational Linguistics.
- Nicola De Cao, Wilker Aziz, and Ivan Titov. 2021. Editing factual knowledge in language models. *arXiv preprint arXiv:2104.08164*.
- Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee.
- Jiaqi Deng, Zonghan Wu, Huan Huo, and Guandong Xu. 2025. A comprehensive survey of knowledge-based vision question answering systems: The lifecycle of knowledge in visual reasoning task. *arXiv preprint arXiv:2504.17547*.
- Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2024. Qlora: Efficient finetuning of quantized llms. *Advances in Neural Information Processing Systems*, 36.

- Tim Dettmers and Luke Zettlemoyer. 2019. Sparse networks from scratch: Faster training without losing performance. *arXiv preprint arXiv:1907.04840*.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*.
- Bhuwan Dhingra, Jeremy R Cole, Julian Martin Eisenschlos, Daniel Gillick, Jacob Eisenstein, and William W Cohen. 2022. Time-aware language models as temporal knowledge bases. *Transactions of the Association for Computational Linguistics*, 10:257–273.
- Ning Ding, Shengding Hu, Weilin Zhao, Yulin Chen, Zhiyuan Liu, Hai-Tao Zheng, and Maosong Sun. 2021. Openprompt: An open-source framework for prompt-learning. *arXiv preprint arXiv:2111.01998*.
- Ning Ding, Yujia Qin, Guang Yang, Fuchao Wei, Zonghan Yang, Yusheng Su, Shengding Hu, Yulin Chen, Chi Min Chan, and Weize and Chen. 2023. Parameter-efficient fine-tuning of large-scale pre-trained language models. *Nature Machine Intelligence*, 5(3):220–235.
- Xiao Ding, Kuo Liao, Ting Liu, Zhongyang Li, and Junwen Duan. 2019. Event representation learning enhanced with external commonsense knowledge. *arXiv preprint arXiv:1909.05190*.
- Saket Dingliwa, Ashish Shenoy, Sravan Bodapati, Ankur Gandhe, Ravi Teja Gadde, and Katrin Kirchhoff. 2022. Domain prompts: Towards memory and compute efficient domain adaptation of asr systems.
- Yinpeng Dong, Hang Su, Jun Zhu, and Bo Zhang. 2017. Improving interpretability of deep neural networks with semantic information. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4306–4314.
- Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2020a. [An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale](#). ArXiv: 2010.11929.
- Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. 2020b. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Alexey Dosovitskiy and Thomas Brox. 2016. Inverting visual representations with convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4829–4837.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Olivier Duchenne, Onur ‘elebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collet, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vladan Petrovic, Wei-

wei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaoqing Ellen Tan, Xinfeng Xie, Xuchao Jia, Xuwei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papanikolaou, Aaditya Singh, Aaron Grattafiori, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alex Vaughan, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Franco, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Changan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, Danny Wyatt, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkan Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Firat Ozgenel, Francesco Caggioni, Francisco Guzmán, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Govind Thattai, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Ibrahim Damla, Igor Molybog, Igor Tufanov, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Karthik Prasad, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kun Huang, Kunal Chawla, Kushal Lakhotia, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabza, Manav Avalani, Manish Bhatt, Maria Tsimpoukelli, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Greshnev, Maxim Naumov, Maya Lathi, Meghan Keneally, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikolay Pavlovich Laptev, Ning Dong, Ning Zhang, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji,

- Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Rohan Maheswari, Russ Howes, Rutu Rinott, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Kohler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vitor Albiero, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaofang Wang, Xiaojian Wu, Xiaolan Wang, Xide Xia, Xilun Wu, Xinbo Gao, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yuchen Hao, Yundi Qian, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, and Zhiwei Zhao. 2024a. [The Llama 3 Herd of Models](#). ArXiv:2407.21783.
- Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024b. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.
- Nelson Elhage, Tristan Hume, Catherine Olsson, Nicholas Schiefer, Tom Henighan, Shauna Kravec, Zac Hatfield-Dodds, Robert Lasenby, Dawn Drain, Carol Chen, Roger Grosse, Sam McCandlish, Jared Kaplan, Dario Amodei, Martin Wattenberg, and Christopher Olah. 2022. Toy models of superposition. *Transformer Circuits Thread*. https://transformer-circuits.pub/2022/toy_model/index.html.
- Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Nova DasSarma, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. 2021. A mathematical framework for transformer circuits. *Transformer Circuits Thread*. <https://transformer-circuits.pub/2021/framework/index.html>.
- Allyson Ettinger. 2020. What bert is not: Lessons from a new suite of psycholinguistic diagnostics for language models. *Transactions of the Association for Computational Linguistics*, 8:34–48.

- Fenglei Fan, Jinjun Xiong, Mengzhou Li, and Ge Wang. 2020. On interpretability of artificial neural networks: A survey. *IEEE Transactions on Radiation and Plasma Medical Sciences*, 5:741–760.
- Yuxin Fang, Wen Wang, Binhui Xie, Quan Sun, Ledell Wu, Xinggang Wang, Tiejun Huang, Xinlong Wang, and Yue Cao. 2023. Eva: Exploring the limits of masked visual representation learning at scale. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19358–19369.
- Li Fei-Fei, Robert Fergus, and Pietro Perona. 2006. One-shot learning of object categories. *IEEE transactions on pattern analysis and machine intelligence*, 28(4):594–611.
- Stella Frank, Emanuele Bugliarello, and Desmond Elliott. 2021a. [Vision-and-Language or Vision-for-Language? On Cross-Modal Influence in Multimodal Transformers](#). pages 9847–9857. ArXiv: 2109.04448 ISBN: 9781955917094.
- Stella Frank, Emanuele Bugliarello, and Desmond Elliott. 2021b. Vision-and-language or vision-for-language? on cross-modal influence in multimodal transformers. *arXiv preprint arXiv:2109.04448*.
- Jonathan Frankle and Michael Carbin. 2018. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*.
- Jonathan Frankle and Michael Carbin. 2019. [The lottery ticket hypothesis: Finding sparse, trainable neural networks](#). *7th International Conference on Learning Representations, ICLR 2019*. ArXiv: 1803.03635.
- Zihao Fu, Haoran Yang, Anthony Man-Cho So, Wai Lam, Lidong Bing, and Nigel Collier. 2023. On the effectiveness of parameter-efficient fine-tuning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 12799–12807.
- Trevor Gale, Erich Elsen, and Sara Hooker. 2019. The state of sparsity in deep neural networks. *arXiv preprint arXiv:1902.09574*.
- Jingying Gao, Qi Wu, Alan Blair, and Maurice Pagnucco. 2023a. Lora: A logical reasoning augmented dataset for visual question answering. *Advances in Neural Information Processing Systems*, 36:30579–30591.
- Peng Gao*, Shijie Geng*, Renrui Zhang*, Teli Ma, Rongyao Fang, Yongfeng Zhang, Hongsheng Li, and Yu Qiao. 2021. Clip-adapter: Better vision-language models with feature adapters. *IJCV 2023*.
- Peng Gao, Jiaming Han, Renrui Zhang, Ziyi Lin, Shijie Geng, Aojun Zhou, Wei Zhang, Pan Lu, Conghui He, Xiangyu Yue, et al. 2023b. Llama-adapter v2: Parameter-efficient visual instruction model. *arXiv preprint arXiv:2304.15010*.

- Tianyu Gao, Adam Fisch, and Danqi Chen. 2020. Making pre-trained language models better few-shot learners. *arXiv preprint arXiv:2012.15723*.
- Tianyu et al. Gao. 2021. Simcse: Simple contrastive learning of sentence embeddings. *Proceedings of EMNLP*.
- Manas Gaur, Kalpa Gunaratna, Vijay Srinivasan, and Hongxia Jin. 2022. Iseeq: Information seeking question generation using dynamic meta-information retrieval and knowledge graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 10672–10680.
- Timnit Gebru, Jonathan Krause, Yilun Wang, Duyun Chen, Jia Deng, and Li Fei-Fei. 2017. Fine-grained car detection for visual census estimation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31.
- Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. 2013. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237.
- Mor Geva, Jasmijn Bastings, Katja Filippova, and Amir Globerson. 2023. [Dissecting Recall of Factual Associations in Auto-Regressive Language Models](#). ArXiv:2304.14767 [cs].
- Mor Geva, Avi Caciularu, Kevin Ro Wang, and Yoav Goldberg. 2022. Transformer feed-forward layers build predictions by promoting concepts in the vocabulary space. *arXiv preprint arXiv:2203.14680*.
- Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. 2021. [Transformer feed-forward layers are key-value memories](#). In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.
- Nicholas Goldowsky-Dill, Chris MacLeod, Lucas Sato, and Aryaman Arora. 2023. Localizing model behavior with path patching. *arXiv preprint arXiv:2304.05969*.
- Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations (ICLR)*.
- Ben Graham. 2015. Kaggle diabetic retinopathy detection competition report. *University of Warwick*, pages 24–26.
- Benjamin Graham et al. 2021. Levit: A vision transformer in convnet’s clothing for faster inference. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.

- Jian Guan, Fei Huang, Zhihao Zhao, Xiaoyan Zhu, and Minlie Huang. 2020. A knowledge-enhanced pretraining model for commonsense story generation. *Transactions of the Association for Computational Linguistics*, 8:93–108.
- Yang Guan, Yangang Ren, Qi Sun, Shengbo Eben Li, Haitong Ma, Jingliang Duan, Yifan Dai, and Bo Cheng. 2022. Integrated decision and control: Toward interpretable and computationally efficient driving intelligence. *IEEE transactions on cybernetics*, 53(2):859–873.
- Demi Guo, Alexander M Rush, and Yoon Kim. 2020a. Parameter-efficient transfer learning with diff pruning. *arXiv preprint arXiv:2012.07463*.
- Pengsheng Guo, Chen-Yu Lee, and Daniel Ulbricht. 2020b. Learning to branch for multi-task learning. In *International conference on machine learning*, pages 3854–3863. PMLR.
- Yunhui Guo, Honghui Shi, Abhishek Kumar, Kristen Grauman, Tajana Rosing, and Rogerio Feris. 2019. Spottune: transfer learning through adaptive fine-tuning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4805–4814.
- Song Han, Huizi Mao, and William J Dally. 2015a. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*.
- Song Han, Jeff Pool, John Tran, and William Dally. 2015b. Learning both weights and connections for efficient neural network. *Advances in neural information processing systems*, 28.
- Michael Hanna, Ollie Liu, and Alexandre Variengien. 2023. How does gpt-2 compute greater-than?: Interpreting mathematical abilities in a pre-trained language model. *Advances in Neural Information Processing Systems*, 36:76033–76060.
- Trevor J Hastie. 2017. Generalized additive models. *Statistical models in S*, pages 249–307.
- Haoyu He, Jianfei Cai, Jing Zhang, Dacheng Tao, and Bohan Zhuang. 2023a. Sensitivity-aware visual parameter-efficient tuning. *arXiv preprint arXiv:2303.08566*.
- Haoyu He, Jianfei Cai, Jing Zhang, Dacheng Tao, and Bohan Zhuang. 2023b. [Sensitivity-Aware Visual Parameter-Efficient Tuning](#). ArXiv:2303.08566 [cs].
- Haoze He, Juncheng Billy Li, Xuan Jiang, and Heather Miller. 2024. [Sparse Matrix in Large Language Model Fine-tuning](#). ArXiv:2405.15525 [cs].

- Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. 2021a. Towards a unified view of parameter-efficient transfer learning. *ArXiv*, abs/2110.04366.
- Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. 2021b. Towards a unified view of parameter-efficient transfer learning. *arXiv preprint arXiv:2110.04366*.
- Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020a. Momentum contrast for unsupervised visual representation learning. In *Proceedings of CVPR*.
- Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. 2017. Mask r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.
- Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Kaiming He et al. 2022. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020b. Deberta: Decoding-enhanced bert with disentangled attention. *arXiv preprint arXiv:2006.03654*.
- Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. 2019. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12(7):2217–2226.
- Roe Hendel, Mor Geva, and Amir Globerson. 2023. In-context learning creates task vectors. *arXiv preprint arXiv:2310.15916*.
- Lisa Anne Hendricks and Aida Nematzadeh. 2021. [Probing image-language transformers for verb understanding](#). In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3635–3644, Online. Association for Computational Linguistics.
- Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, et al. 2021a. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8340–8349.
- Dan Hendrycks and Thomas Dietterich. 2019. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*.

- Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. 2021b. Natural adversarial examples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15262–15271.
- Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, pages 6840–6851.
- Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.
- Mohammad Javad Hosseini, Hannaneh Hajishirzi, Oren Etzioni, and Nate Kushman. 2014. [Learning to solve arithmetic word problems with verb categorization](#). In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 523–533, Doha, Qatar. Association for Computational Linguistics.
- Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Larousilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR.
- Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Annual Meeting of the Association for Computational Linguistics*.
- Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021a. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.
- Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022a. Lora: Low-rank adaptation of large language models. *International Conference on Learning Representations*.
- Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022b. [LoRA: Low-rank adaptation of large language models](#). In *The Tenth International Conference on Learning Representations, ICLR 2022*, Virtual Event.
- Ronghang Hu, Jacob Andreas, Trevor Darrell, and Kate Saenko. 2018. Explainable neural computation via stack neural module networks. In *Proceedings of the European conference on computer vision (ECCV)*, pages 53–69.
- Ronghang Hu, Marcus Rohrbach, Jacob Andreas, Trevor Darrell, and Kate Saenko. 2017. Modeling relationships in referential expressions with compositional modular networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1115–1124.

- Ronghang Hu, Huazhe Xu, Marcus Rohrbach, Jiashi Feng, Kate Saenko, and Trevor Darrell. 2016. Natural language object retrieval. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4555–4564.
- Shengding Hu, Ning Ding, Huadong Wang, Zhiyuan Liu, Jingang Wang, Juanzi Li, Wei Wu, and Maosong Sun. 2021b. Knowledgeable prompt-tuning: Incorporating knowledge into prompt verbalizer for text classification. *arXiv preprint arXiv:2108.02035*.
- Shengding Hu, Zhen Zhang, Ning Ding, Yadao Wang, Yasheng Wang, Zhiyuan Liu, and Maosong Sun. 2022c. [Sparse structure search for parameter-efficient tuning](#).
- Zhiqiang Hu, Lei Wang, Yihuai Lan, Wanyu Xu, Ee-Peng Lim, Lidong Bing, Xing Xu, Soujanya Poria, and Roy Lee. 2023a. [LLM-Adapters: An Adapter Family for Parameter-Efficient Fine-Tuning of Large Language Models](#). In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5254–5276, Singapore. Association for Computational Linguistics.
- Zhiqiang Hu, Lei Wang, Yihuai Lan, Wanyu Xu, Ee-Peng Lim, Lidong Bing, Xing Xu, Soujanya Poria, and Roy Lee. 2023b. [LLM-adapters: An adapter family for parameter-efficient fine-tuning of large language models](#). In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5254–5276, Singapore. Association for Computational Linguistics.
- Drew A Hudson and Christopher D Manning. 2019. Gqa: A new dataset for real-world visual reasoning and compositional question answering. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6700–6709.
- Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, and Aleksander Madry. 2019. Adversarial examples are not bugs, they are features. In *Advances in neural information processing systems*, pages 125–136.
- Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pages 448–456.
- Ahmet Iscen, Jeffrey Zhang, Svetlana Lazebnik, and Cordelia Schmid. 2020. Memory-efficient incremental learning through feature adaptation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVI 16*, pages 699–715. Springer.
- Hamish Ivison, Noah A Smith, Hannaneh Hajishirzi, and Pradeep Dasigi. 2022. Data-efficient finetuning using cross-task nearest neighbors. *arXiv preprint arXiv:2212.00196*.
- Saachi Jain, Hannah Lawrence, Ankur Moitra, and Aleksander Madry. 2022. Distilling model failures as directions in latent space. *arXiv preprint arXiv:2206.14754*.

- Sarthak Jain and Byron C Wallace. 2019. Attention is not explanation. *arXiv preprint arXiv:1902.10186*.
- Debesh Jha, Steven A. Hicks, Krister Emanuelsen, Håvard Johansen, Dag Johansen, Thomas de Lange, Michael A. Riegler, and Pål Halvorsen. 2020a. [Medico multimedia task at mediaeval 2020: Automatic polyp segmentation](#).
- Debesh Jha, Pia H Smedsrud, Michael A Riegler, Pål Halvorsen, Thomas de Lange, Dag Johansen, and Håvard D Johansen. 2020b. Kvasir-seg: A segmented polyp dataset. In *International Conference on Multimedia Modeling*, pages 451–462. Springer.
- Chao Jia, Yinfei Yang, Yi-Ting Xia, et al. 2021. Scaling up visual and vision-language representation learning with noisy text supervision. In *International Conference on Machine Learning (ICML)*.
- Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. 2022a. Visual prompt tuning. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXIII*, pages 709–727. Springer.
- Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. 2022b. [Visual Prompt Tuning](#). ArXiv:2203.12119 [cs].
- Zhengbao Jiang, Frank F Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438.
- Justin Johnson, Bharath Hariharan, Laurens Van Der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. 2017. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2901–2910.
- Sonia Joseph. 2023. Vit prisma: A mechanistic interpretability library for vision transformers. <https://github.com/soniajoseph/vit-prisma>.
- Chen Ju, Tengda Han, Kunhao Zheng, Ya Zhang, and Weidi Xie. 2022. Prompting visual-language models for efficient video understanding. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXV*, pages 105–124. Springer.
- Rabeeh Karimi Mahabadi, James Henderson, and Sebastian Ruder. 2021. Compacter: Efficient low-rank hypercomplex adapter layers. *Advances in Neural Information Processing Systems*, 34:1022–1035.
- Adilbek Karmanov, Dayan Guan, Shijian Lu, Abdulmotaleb El Saddik, and Eric Xing. 2024. Efficient test-time adaptation of vision-language models. In *Proceedings of the*

- IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14162–14171.
- Nora Kassner and Hinrich Schütze. 2020. [Negated and misprimed probes for pretrained language models: Birds can talk, but cannot fly](#). In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7811–7818, Online. Association for Computational Linguistics.
- Sahar Kazemzadeh, Vicente Ordonez, Mark Matten, and Tamara Berg. 2014. Referitgame: Referring to objects in photographs of natural scenes. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 787–798.
- Alex Kendall, Yarin Gal, and Roberto Cipolla. 2018. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7482–7491.
- Aditya Khosla, Nityananda Jayadevaprakash, Bangpeng Yao, and Fei-Fei Li. 2011. Novel dataset for fine-grained image categorization: Stanford dogs. In *Proc. CVPR workshop on fine-grained visual categorization (FGVC)*, volume 2. Citeseer.
- Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschiot, Ce Liu, and Dilip Krishnan. 2020. Supervised contrastive learning. *ArXiv*, abs/2004.11362.
- Been Kim, Martin Wattenberg, Justin Gilmer, Carrie Cai, James Wexler, Fernanda Viegas, and Rory Sayres. 2018. [Interpretability Beyond Feature Attribution: Quantitative Testing with Concept Activation Vectors \(TCAV\)](#). *ArXiv:1711.11279 [stat]*.
- Siwon Kim, Jinoh Oh, Sungjin Lee, Seunghak Yu, Jaeyoung Do, and Tara Taghavi. 2023. Grounding counterfactual explanation of image classifiers to textual concept space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10942–10950.
- Yoon Kim. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*.
- Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.
- Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. 2023a. Segment anything. *arXiv:2304.02643*.
- Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al.

- 2023b. Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4015–4026.
- Iasonas Kokkinos. 2017. Ubernet: Training a universal convolutional neural network for low-, mid-, and high-level vision using diverse datasets and limited memory. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6129–6138.
- Rik Koncel-Kedziorski, Hannaneh Hajishirzi, Ashish Sabharwal, Oren Etzioni, and Siena Dumas Ang. 2015. [Parsing algebraic word problems into equations](#). *Transactions of the Association for Computational Linguistics*, 3:585–597.
- Rik Koncel-Kedziorski, Subhro Roy, Aida Amini, Nate Kushman, and Hannaneh Hajishirzi. 2016. [MAWPS: A math word problem repository](#). In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1152–1157, San Diego, California. Association for Computational Linguistics.
- Dawid Jan Kopiczko, Tijmen Blankevoort, and Yuki Markus Asano. 2023. Vera: Vector-based random matrix adaptation. *arXiv preprint arXiv:2310.11454*.
- Mathias Kraus, Daniel Tschernutter, Sven Weinzierl, and Patrick Zschech. 2024. Interpretable generalized additive neural networks. *European Journal of Operational Research*, 317(2):303–316.
- Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. 2016. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *arXiv preprint arXiv:1602.07332*.
- Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. 2017. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International journal of computer vision*, 123:32–73.
- Alex Krizhevsky, Geoffrey Hinton, et al. 2009. Learning multiple layers of features from tiny images.
- Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems*.
- John K Kruschke and Javier R Movellan. 1991. Benefits of gain: Speeded learning and minimal hidden layers in back-propagation networks. *IEEE Transactions on systems, Man, and Cybernetics*, 21(1):273–280.

- Ananya Kumar, Aditi Raghunathan, Robbie Jones, Tengyu Ma, and Percy Liang. 2022. Fine-tuning can distort pretrained features and underperform out-of-distribution. *arXiv preprint arXiv:2202.10054*.
- Biagio La Rosa, Roberto Capobianco, and Daniele Nardi. 2023. A self-interpretable module for deep image classification on small data. *Applied Intelligence*, 53(8):9115–9147.
- François Lagunas, Ella Charlaix, Victor Sanh, and Alexander M Rush. 2021. Block pruning for faster transformers. *arXiv preprint arXiv:2109.04838*.
- Anna Langedijk, Hosein Mohebbi, Gabriele Sarti, Willem Zuidema, and Jaap Jumelet. 2023. Decoderlens: Layerwise interpretation of encoder-decoder transformers. *arXiv preprint arXiv:2310.03686*.
- Sebastian Lapuschkin, Stefan Wäldchen, Alexander Binder, Grégoire Montavon, Wojciech Samek, and Klaus-Robert Müller. 2019. Unmasking clever hans predictors and assessing what machines really learn. *Nature communications*, 10(1):1096.
- Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.
- Yann LeCun, Fu Jie Huang, and Leon Bottou. 2004. Learning methods for generic object recognition with invariance to pose and lighting. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, volume 2, pages II–104. IEEE.
- Hector Levesque, Ernest Davis, and Leora Morgenstern. 2012. [The Winograd Schema Challenge](#). In *Proceedings of the Thirteenth International Conference on Principles of Knowledge Representation and Reasoning*.
- Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *Proceedings of ACL*.
- Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. 2016a. Pruning filters for efficient convnets. *arXiv preprint arXiv:1608.08710*.
- Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. 2016b. Pruning filters for efficient convnets. *ArXiv*, abs/1608.08710.
- Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. 2023a. [BLIP-2: Bootstrapping Language-Image Pre-training with Frozen Image Encoders and Large Language Models](#). ArXiv: 2301.12597.

- Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. [BLIP: Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation](#). Technical report.
- Junnan Li, Ramprasaath Selvaraju, Shuang Zhang, et al. 2022a. Blip: Bootstrapped language-image pretraining for unified vision-language understanding and generation. In *NeurIPS*.
- Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. 2023b. Inference-time intervention: Eliciting truthful answers from a language model. *Advances in Neural Information Processing Systems*, 36:41451–41530.
- Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. 2019. Visualbert: A simple and performant baseline for vision and language. *arXiv preprint arXiv:1908.03557*.
- Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*.
- Yanwei Li, Yuechen Zhang, Chengyao Wang, Zhisheng Zhong, Yixin Chen, Ruihang Chu, Shaoteng Liu, and Jiaya Jia. 2024. [Mini-gemini: Mining the potential of multi-modality vision language models](#).
- Yizhe Li, Huazheng Zhang, Yuwei Wang, and Dong Guo. 2022b. Diffusion-lm improves controllable text generation. *arXiv preprint arXiv:2211.15089*.
- Dongze Lian, Daquan Zhou, Jiashi Feng, and Xinchao Wang. 2022. Scaling & shifting your features: A new baseline for efficient model tuning. *arXiv preprint arXiv:2210.08823*.
- Dongze Lian, Daquan Zhou, Jiashi Feng, and Xinchao Wang. 2023. [Scaling & Shifting Your Features: A New Baseline for Efficient Model Tuning](#). ArXiv:2210.08823 [cs].
- Baohao Liao, Yan Meng, and Christof Monz. 2023a. Parameter-efficient fine-tuning without introducing new latency. *arXiv preprint arXiv:2305.16742*.
- Baohao Liao, Shaomu Tan, and Christof Monz. 2023b. Make pre-trained model reversible: From parameter to memory efficient fine-tuning. *Advances in Neural Information Processing Systems*, 36.
- Tom Lieberum, Matthew Rahtz, János Kramár, Neel Nanda, Geoffrey Irving, Rohin Shah, and Vladimir Mikulik. 2023. Does circuit analysis interpretability scale? evidence from multiple choice capabilities in chinchilla. *arXiv preprint arXiv:2307.09458*.
- Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer.

- Xinyu Lin, Wenjie Wang, Yongqi Li, Shuo Yang, Fuli Feng, Yinwei Wei, and Tat-Seng Chua. 2024. Data-efficient fine-tuning for llm-based recommendation. In *Proceedings of the 47th international ACM SIGIR conference on research and development in information retrieval*, pages 365–374.
- Zhaojiang Lin, Andrea Madotto, and Pascale Fung. 2020. Exploring versatile generative language model via parameter-efficient transfer learning. *arXiv preprint arXiv:2004.03829*.
- Zihao Lin, Samyadeep Basu, Mohammad Beigi, Varun Manjunatha, Ryan A. Rossi, Zichao Wang, Yufan Zhou, Sriram Balasubramanian, Arman Zarei, Keivan Rezaei, Ying Shen, Barry Menglong Yao, Zhiyang Xu, Qin Liu, Yuxiang Zhang, Yan Sun, Shilong Liu, Li Shen, Hongxuan Li, Soheil Feizi, and Lifu Huang. 2025. [A Survey on Mechanistic Interpretability for Multi-Modal Foundation Models](#). ArXiv:2502.17516 [cs].
- Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. 2017. [Program induction by rationale generation: Learning to solve and explain algebraic word problems](#). *arXiv:1705.04146*.
- Bo Liu, Yihao Feng, Peter Stone, and Qiang Liu. 2023a. [Famo: Fast adaptive multitask optimization](#).
- Bo Liu, Xingchao Liu, Xiaojie Jin, Peter Stone, and Qiang Liu. 2021a. Conflict-averse gradient descent for multi-task learning. *Advances in Neural Information Processing Systems*, 34:18878–18890.
- Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin Raffel. 2022a. [Few-Shot Parameter-Efficient Fine-Tuning is Better and Cheaper than In-Context Learning](#). ArXiv:2205.05638 [cs].
- Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. 2024a. Improved baselines with visual instruction tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 26296–26306.
- Haotian Liu, Chunyuan Li, Yuheng Li, Bo Li, Yuanhan Zhang, Sheng Shen, and Yong Jae Lee. 2024b. [Llava-next: Improved reasoning, ocr, and world knowledge](#).
- Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2023b. [Visual Instruction Tuning](#). ArXiv:2304.08485 [cs].
- Haotian Liu et al. 2023c. Llava: Large language and vision assistant. *ArXiv*.
- Jiacheng Liu, Alisa Liu, Ximing Lu, Sean Welleck, Peter West, Ronan Le Bras, Yejin Choi, and Hannaneh Hajishirzi. 2021b. Generated knowledge prompting for commonsense reasoning. *arXiv preprint arXiv:2110.08387*.

- Liyang Liu, Yi Li, Zhanghui Kuang, J Xue, Yimin Chen, Wenming Yang, Qingmin Liao, and Wayne Zhang. 2021c. Towards impartial multi-task learning. In *International Conference on Learning Representations*.
- Nelson F Liu, Matt Gardner, Yonatan Belinkov, Matthew E Peters, and Noah A Smith. 2019a. Linguistic knowledge and transferability of contextual representations. *arXiv preprint arXiv:1903.08855*.
- Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2023d. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55(9):1–35.
- Sheng Liu, Haotian Ye, Lei Xing, and James Zou. 2023e. In-context vectors: Making in context learning more effective and controllable through latent space steering. *arXiv preprint arXiv:2311.06668*.
- Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. 2024c. [DoRA: Weight-decomposed low-rank adaptation](#). *arXiv:2402.09353*.
- Shikun Liu, Edward Johns, and Andrew J Davison. 2019b. End-to-end multi-task learning with attention. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1871–1880.
- Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2022b. P-tuning: Prompt tuning can be comparable to fine-tuning across scales and tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 61–68.
- Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019c. [RoBERTa: A robustly optimized BERT pretraining approach](#). *arXiv:1907.11692*.
- Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. 2021d. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022.
- Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. 2022c. A convnet for the 2020s. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11976–11986.
- Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. 2022d. [A ConvNet for the 2020s](#). ArXiv:2201.03545 [cs].

- Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. 2015. Deep learning face attributes in the wild. In *Proceedings of the IEEE international conference on computer vision*, pages 3730–3738.
- Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.
- Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *arXiv preprint arXiv:1908.02265*.
- Adriano Lucieri, Muhammad Naseer Bajwa, Stephan Alexander Braun, Muhammad Imran Malik, Andreas Dengel, and Sheraz Ahmed. 2020. On interpretability of deep learning based skin lesion classifiers using concept activation vectors. In *2020 international joint conference on neural networks (IJCNN)*, pages 1–10. IEEE.
- Kexin Luo, Guanci Yang, Yang Li, Shangen Lan, Yang Wang, Ling He, and Binqi Hu. 2024. Croup and pertussis cough sound classification algorithm based on channel attention and multiscale mel-spectrogram. *Biomedical Signal Processing and Control*, 91:106073.
- Maria Lymperaioi and Giorgos Stamou. 2024. A survey on knowledge-enhanced multimodal learning. *Artificial Intelligence Review*, 57(10):284.
- Yiwei Lyu, Paul Pu Liang, Zihao Deng, Ruslan Salakhutdinov, and Louis-Philippe Morency. 2022. Dime: Fine-grained interpretations of multimodal models via disentangled local explanations. In *Proceedings of the 2022 AAAI/ACM Conference on AI, Ethics, and Society*, pages 455–467.
- Rabeeh Karimi Mahabadi, Sebastian Ruder, Mostafa Dehghani, and James Henderson. 2021. Parameter-efficient multi-task fine-tuning for transformers via shared hypernetworks. *arXiv preprint arXiv:2106.04489*.
- Aravindh Mahendran and Andrea Vedaldi. 2015. Understanding deep image representations by inverting them. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5188–5196.
- Kevis-Kokitsi Maninis, Ilija Radosavovic, and Iasonas Kokkinos. 2019. Attentive single-tasking of multiple tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1851–1860.
- Junhua Mao, Jonathan Huang, Alexander Toshev, Oana Camburu, Alan L Yuille, and Kevin Murphy. 2016. Generation and comprehension of unambiguous object descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 11–20.

- Kenneth Marino, Mohammad Rastegari, Ali Farhadi, and Roozbeh Mottaghi. 2019. [OK-VQA: A Visual Question Answering Benchmark Requiring External Knowledge](#). ArXiv:1906.00067 [cs].
- Samuel Marks, Can Rager, Eric J Michaud, Yonatan Belinkov, David Bau, and Aaron Mueller. 2024. Sparse feature circuits: Discovering and editing interpretable causal graphs in language models. *arXiv preprint arXiv:2403.19647*.
- Loic Matthey, Irina Higgins, Demis Hassabis, and Alexander Lerchner. 2017. dsprites: Disentanglement testing sprites dataset.
- Kevin Meng, David Bau, Alex J Andonian, and Yonatan Belinkov. 2022. [Locating and editing factual associations in GPT](#). In *Advances in Neural Information Processing Systems*.
- Otniel-Bogdan Mercea, Alexey Gritsenko, Cordelia Schmid, and Anurag Arnab. 2024. Time-memory-and parameter-efficient visual adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5536–5545.
- Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. [Can a suit of armor conduct electricity? A new dataset for open book question answering](#). *arXiv:1809.02789*.
- Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.
- Marvin Minsky. 2000. Commonsense-based ai. In *Communications of the ACM*, volume 43, pages 65–66.
- Ishan Misra, Abhinav Shrivastava, Abhinav Gupta, and Martial Hebert. 2016. Cross-stitch networks for multi-task learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3994–4003.
- Sameh K Mohamed, Aayah Nounu, and Vít Nováček. 2021. Biological applications of knowledge graph embedding models. *Briefings in bioinformatics*, 22(2):1679–1693.
- Milad Moradi and Matthias Samwald. 2021. Evaluating the robustness of neural language models to input perturbations. *arXiv preprint arXiv:2108.12237*.
- Marius Mosbach, Vagrant Gautam, Tomás Vergara-Browne, Dietrich Klakow, and Mor Geva. 2024. From insights to actions: The impact of interpretability and analysis research on nlp. *arXiv preprint arXiv:2406.12618*.
- Hesham Mostafa and Xin Wang. 2019. Parameter efficient training of deep convolutional neural networks by dynamic sparse reparameterization. In *International Conference on Machine Learning*, pages 4646–4655. PMLR.

- Keerthiram Murugesan and Jaime Carbonell. 2017. Self-paced multitask learning with shared knowledge. *arXiv preprint arXiv:1703.00977*.
- Moin Nadeem, Anna Bethke, and Siva Reddy. 2020. Stereoset: Measuring stereotypical bias in pretrained language models. *arXiv preprint arXiv:2004.09456*.
- Neel Nanda, Lawrence Chan, Tom Lieberum, Jess Smith, and Jacob Steinhardt. 2023. Progress measures for grokking via mechanistic interpretability. *arXiv preprint arXiv:2301.05217*.
- Muhammad Muzammal Naseer, Kanchana Ranasinghe, Salman H Khan, Munawar Hayat, Fahad Shahbaz Khan, and Ming-Hsuan Yang. 2021. Intriguing properties of vision transformers. *Advances in Neural Information Processing Systems*, 34:23296–23308.
- Aviv Navon, Aviv Shamsian, Idan Achituve, Haggai Maron, Kenji Kawaguchi, Gal Chechik, and Ethan Fetaya. 2022. [Multi-task learning as a bargaining game](#).
- Daniel Neil, Joss Briody, Alix Lacoste, Aaron Sim, Paidi Creed, and Amir Saffari. 2018. Interpretable graph convolutional neural networks for inference on noisy knowledge graphs. *arXiv preprint arXiv:1812.00279*.
- Clement Neo, Luke Ong, Philip Torr, Mor Geva, David Krueger, and Fazl Barez. 2024. Towards interpreting visual information processing in vision-language models. *arXiv preprint arXiv:2410.07149*.
- Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. 2011. Reading digits in natural images with unsupervised feature learning.
- Maria-Elena Nilsback and Andrew Zisserman. 2008. Automated flower classification over a large number of classes. In *2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing*, pages 722–729. IEEE.
- Yulei Niu, Hanwang Zhang, Zhiwu Lu, and Shih-Fu Chang. 2019. Variational context: Exploiting visual and textual context for grounding referring expressions. *IEEE transactions on pattern analysis and machine intelligence*, 43(1):347–359.
- Nostalgebraist. interpreting GPT: the logit lens. <https://www.lesswrong.com/posts/AcKRB8wDpdaN6v6ru/interpreting-gpt-the-logit-lens>.
- Chris Olah. 2024. Mechanistic interpretability, variables, and the importance of interpretable bases. <https://www.transformer-circuits.pub/2022/mech-interp-essay>. Accessed: 2024-10-20.
- OpenAI. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.
- Vicente Ordonez, Girish Kulkarni, and Tamara Berg. 2011. Im2text: Describing images using 1 million captioned photographs. *Advances in neural information processing systems*, 24:1143–1151.

- Koyena Pal, Jiuding Sun, Andrew Yuan, Byron C Wallace, and David Bau. 2023. Future lens: Anticipating subsequent tokens from a single hidden state. *arXiv preprint arXiv:2311.04897*.
- Vedant Palit, Rohan Pandey, Aryaman Arora, and Paul Pu Liang. Towards Vision-Language Mechanistic Interpretability: A Causal Tracing Tool for BLIP.
- Emilio Parisotto, Lei Jimmy Ba, and Ruslan Salakhutdinov. 2016. Actor-mimic: Deep multitask and transfer reinforcement learning. In *International Conference on Learning Representations*.
- Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, and CV Jawahar. 2012. Cats and dogs. In *2012 IEEE conference on computer vision and pattern recognition*, pages 3498–3505. IEEE.
- Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. [Are NLP models really able to solve simple math word problems?](#) In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2080–2094, Online. Association for Computational Linguistics.
- Fabio Petroni, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H Miller, and Sebastian Riedel. 2019a. Language models as knowledge bases? *arXiv preprint arXiv:1909.01066*.
- Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019b. [Language models as knowledge bases?](#) In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.
- Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. 2020a. Adapterfusion: Non-destructive task composition for transfer learning. *arXiv preprint arXiv:2005.00247*.
- Jonas Pfeiffer, Andreas Rücklé, Clifton Poth, Aishwarya Kamath, Ivan Vulić, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. 2020b. Adapterhub: A framework for adapting transformers. *arXiv preprint arXiv:2007.07779*.
- Sai Prasanna, Anna Rogers, and Anna Rumshisky. 2020. When bert plays the lottery, all tickets are winning. In *Conference on Empirical Methods in Natural Language Processing*.
- John G. Proakis and Dimitris G. Manolakis. 1992. Digital signal processing: Principles, algorithms, and applications.

- Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021a. [CLIP: Learning Transferable Visual Models From Natural Language Supervision](#). ArXiv: 2103.00020.
- Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021b. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR.
- Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. *OpenAI*.
- Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.
- Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*.
- Maithra Raghu, Thomas Unterthiner, Simon Kornblith, Chiyuan Zhang, and Alexey Dosovitskiy. 2021. Do vision transformers see like convolutional neural networks? *Advances in Neural Information Processing Systems*, 34:12116–12128.
- Amrutha Varshini Ramesh, Vignesh Ganapathiraman, Issam H Laradji, and Mark Schmidt. 2024. Blockllm: Memory-efficient adaptation of llms by selecting and optimizing the right coordinate blocks. *arXiv preprint arXiv:2406.17296*.
- Jiahua Rao, Zifei Shan, Longpo Liu, Yao Zhou, and Yuedong Yang. 2023. Retrieval-based knowledge augmented vision language pre-training. In *Proceedings of the 31st ACM International Conference on Multimedia*, pages 5399–5409.
- Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. 2017. Learning multiple visual domains with residual adapters. *Advances in neural information processing systems*, 30.
- Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28:91–99.
- Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. ” why should i trust you?” explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144.
- Robin Rombach et al. 2022. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

- Subhro Roy and Dan Roth. 2015. [Solving general arithmetic word problems](#). In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1743–1752, Lisbon, Portugal. Association for Computational Linguistics.
- Andreas Rücklé, Gregor Geigle, Max Glockner, Tilman Beck, Jonas Pfeiffer, Nils Reimers, and Iryna Gurevych. 2020. Adapterdrop: On the efficiency of adapters in transformers. *arXiv preprint arXiv:2010.11918*.
- David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. 1986. Learning representations by back-propagating errors. *Nature*, 323:533–536.
- Andrei A Rusu, Sergio Gomez Colmenarejo, Caglar Gulcehre, Guillaume Desjardins, James Kirkpatrick, Razvan Pascanu, Volodymyr Mnih, Koray Kavukcuoglu, and Raia Hadsell. 2015. Policy distillation. *arXiv preprint arXiv:1511.06295*.
- Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. 2017. Dynamic routing between capsules. *Advances in neural information processing systems*, 30.
- Devendra Singh Sachan and Graham Neubig. 2018. Parameter sharing methods for multilingual self-attentional translation models. *arXiv preprint arXiv:1809.00252*.
- Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. [Wino-Grande: An adversarial Winograd Schema Challenge at scale](#). *Communications of the ACM*, 64(9):99–106.
- Mansi Sakarvadia, Arham Khan, Aswathy Ajith, Daniel Grzenda, Nathaniel Hudson, André Bauer, Kyle Chard, and Ian Foster. 2023. Attention lens: A tool for mechanistically interpreting the attention head information retrieval mechanism. *arXiv preprint arXiv:2310.16270*.
- Emmanuelle Salin, Badreddine Farah, Stéphanie Ayache, Benoit Favre, Emmanuelle Salin, Badreddine Farah, Stéphanie Ayache, Benoit Favre Are Vision-language Trans, and Probing Perspective. 2022. Are Vision-Language Transformers Learning Multimodal Representations? A probing perspective. *Proceedings of the 36th AAAI Conference on Artificial Intelligence*.
- Victor Sanh, Thomas Wolf, and Alexander Rush. 2020. Movement pruning: Adaptive sparsity by fine-tuning. *Advances in neural information processing systems*, 33:20378–20389.
- Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. 2019. [Social IQa: Commonsense reasoning about social interactions](#). In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4463–4473, Hong Kong, China. Association for Computational Linguistics.

- Dustin Schwenk, Apoorv Khandelwal, Christopher Clark, Kenneth Marino, and Roozbeh Mottaghi. 2022. [A-OKVQA: A Benchmark for Visual Question Answering using World Knowledge](#). ArXiv:2206.01718 [cs].
- Sarah Schwettmann, Neil Chowdhury, Samuel Klein, David Bau, and Antonio Torralba. 2023. [Multimodal Neurons in Pretrained Text-Only Transformers](#). In *2023 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*, pages 2854–2859, Paris, France. IEEE.
- Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. 2017. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626.
- Ozan Sener and Vladlen Koltun. 2018. Multi-task learning as multi-objective optimization. *Advances in Neural Information Processing Systems*, 31.
- Uri Shaham, Maha Elbayad, Vedanuj Goswami, Omer Levy, and Shruti Bhosale. 2022. Causes and cures for interference in multilingual translation. *arXiv preprint arXiv:2212.07530*.
- Claude E. Shannon. 1948. A mathematical theory of communication. *The Bell System Technical Journal*, 27:379–423, 623–656.
- Lloyd S Shapley et al. 1953. A value for n-person games.
- Lee Sharkey, Dan Braun, and Beren Millidge. 2022. Taking features out of superposition with sparse autoencoders. In *AI Alignment Forum*, volume 8, pages 15–16.
- Lee Sharkey, Bilal Chughtai, Joshua Batson, Jack Lindsey, Jeff Wu, Lucius Bushnaq, Nicholas Goldowsky-Dill, Stefan Heimersheim, Alejandro Ortega, Joseph Bloom, Stella Biderman, Adria Garriga-Alonso, Arthur Conmy, Neel Nanda, Jessica Rumbelow, Martin Wattenberg, Nandi Schoots, Joseph Miller, Eric J. Michaud, Stephen Casper, Max Tegmark, William Saunders, David Bau, Eric Todd, Atticus Geiger, Mor Geva, Jesse Hoogland, Daniel Murfet, and Tom McGrath. 2025. [Open Problems in Mechanistic Interpretability](#). ArXiv:2501.16496 [cs].
- Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. 2018. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2556–2565.
- Shufan Shen, Junshu Sun, Xiangyang Ji, Qingming Huang, and Shuhui Wang. 2024. Expanding sparse tuning for low memory usage. *arXiv preprint arXiv:2411.01800*.

- Guangyuan Shi, Qimai Li, Wenlong Zhang, Jiaxin Chen, and Xiao-Ming Wu. 2023. Recon: Reducing conflicting gradients from the root for multi-task learning. *arXiv preprint arXiv:2302.11289*.
- Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. 2017a. Learning important features through propagating activation differences. In *International conference on machine learning*, pages 3145–3153. PMIR.
- Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. 2017b. Learning important features through propagating activation differences. In *International Conference on Machine Learning*.
- Vered Shwartz. 2021. [Commonsense reasoning for natural language processing](#). Accessed: 2025-04-25.
- Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.
- Shashwat Singh, Shauli Ravfogel, Jonathan Herzig, Roei Aharoni, Ryan Cotterell, and Ponnurangam Kumaraguru. 2024. [MiMiC: Minimally modified counterfactuals in the representation space](#). *arXiv:2402.09631*.
- Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. 2017. Smoothgrad: removing noise by adding noise. *arXiv preprint arXiv:1706.03825*.
- Weixi Song, Zuchao Li, Lefei Zhang, Hai Zhao, and Bo Du. 2023. Sparse is enough in fine-tuning pre-trained large language models. *arXiv preprint arXiv:2312.11875*.
- Robyn Speer, Joshua Chin, and Catherine Havasi. 2017a. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Proceedings of AAAI*.
- Robyn Speer, Joshua Chin, and Catherine Havasi. 2017b. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Thirty-first AAAI conference on artificial intelligence*.
- Pablo Sprechmann, Siddhant M Jayakumar, Jack W Rae, Alexander Pritzel, Adria Puigdomenech Badia, Benigno Uribe, Oriol Vinyals, Demis Hassabis, Razvan Pascanu, and Charles Blundell. 2018. Memory-based parameter adaptation. *arXiv preprint arXiv:1802.10542*.
- Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. 2015. Striving for simplicity: The all convolutional net. In *International Conference on Learning Representations (ICLR) Workshop*.
- Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958.

- Gabriela Ben Melech Stan, Raanan Yehezkel Rohekar, Yaniv Gurwicz, Matthew Lyle Olson, Anahita Bhiwandiwalla, Estelle Aflalo, Chenfei Wu, Nan Duan, Shao-Yen Tseng, and Vasudev Lal. 2024. Lvlm-intrepret: An interpretability tool for large vision-language models. *arXiv preprint arXiv:2404.03118*.
- Asa Cooper Stickland and Iain Murray. 2019. Bert and pals: Projected attention layers for efficient adaptation in multi-task learning. In *International Conference on Machine Learning*, pages 5986–5995. PMLR.
- Alessandro Stolfo, Yonatan Belinkov, and Mrinmaya Sachan. 2023. A mechanistic interpretation of arithmetic reasoning in language models using causal mediation analysis. *arXiv preprint arXiv:2305.15054*.
- Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, and Jifeng Dai. 2019. Vl-bert: Pre-training of generic visual-linguistic representations. *arXiv preprint arXiv:1908.08530*.
- Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, and Jifeng Dai. 2020. Vl-bert: Pre-training of generic visual-linguistic representations. In *arXiv preprint arXiv:1908.08530*.
- Yusheng Su, Xiaozhi Wang, Yujia Qin, Chi-Min Chan, Yankai Lin, Zhiyuan Liu, Peng Li, Juanzi Li, Lei Hou, Maosong Sun, et al. 2021. On transferability of prompt tuning for natural language understanding. *arXiv preprint arXiv:2111.06719*.
- Tianxiang Sun, Yunfan Shao, Xiaonan Li, Pengfei Liu, Hang Yan, Xipeng Qiu, and Xuanjing Huang. 2020. Learning sparse sharing architectures for multiple tasks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 8936–8943.
- Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In *International conference on machine learning*, pages 3319–3328. PMLR.
- Yi-Lin Sung, Jaemin Cho, and Mohit Bansal. 2022. Vl-adapter: Parameter-efficient transfer learning for vision-and-language tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5227–5237.
- Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2015. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9.
- Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2013. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*.

- Hao Tan and Mohit Bansal. 2019a. Lxmert: Learning cross-modality encoder representations from transformers. *arXiv preprint arXiv:1908.07490*.
- Hao Tan and Mohit Bansal. 2019b. Lxmert: Learning cross-modality encoder representations from transformers. In *arXiv preprint arXiv:1908.07490*.
- Niket Tandon, Gerard de Melo, and Gerhard Weikum. 2017a. Webchild 2.0: Fine-grained commonsense knowledge distillation. In *ACL System Demonstrations*, pages 115–120.
- Niket Tandon, Gerard De Melo, and Gerhard Weikum. 2017b. Webchild 2.0: Fine-grained commonsense knowledge distillation. In *Proceedings of ACL 2017, System Demonstrations*, pages 115–120.
- Yan Tao, Olga Viberg, Ryan S Baker, and René F Kizilcec. 2024. Cultural bias and cultural alignment of large language models. *PNAS nexus*, 3(9):pgae346.
- Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothee Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. LLaMA: Open and Efficient Foundation Language Models.
- Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023a. [Llama 2: Open Foundation and Fine-Tuned Chat Models](#). ArXiv:2307.09288 [cs].
- Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi,

- Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023b. [Llama 2: Open foundation and fine-tuned chat models](#).
- Hugo et al. Touvron. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.
- Trieu H Trinh and Quoc V Le. 2019. Do language models have common sense?
- Yi-Lin Tuan, Sajjad Beygi, Maryam Fazel-Zarandi, Qiaozi Gao, Alessandra Cervone, and William Yang Wang. 2022. Towards large-scale interpretable knowledge graph reasoning for dialogue systems. *arXiv preprint arXiv:2203.10610*.
- Tsukasa Ueno and Qiangfu Zhao. 2018. Interpretation of deep neural networks based on decision trees. In *2018 IEEE 16th Intl Conf on Dependable, Autonomic and Secure Computing, 16th Intl Conf on Pervasive Intelligence and Computing, 4th Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress (DASC/PiCom/DataCom/CyberSciTech)*, pages 256–261. IEEE.
- Grant Van Horn, Steve Branson, Ryan Farrell, Scott Haber, Jessie Barry, Panos Ipeirotis, Pietro Perona, and Serge Belongie. 2015. Building a bird recognition app and large scale dataset with citizen scientists: The fine print in fine-grained dataset collection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 595–604.
- Simon Vandenhende, Stamatios Georgoulis, Wouter Van Gansbeke, Marc Proesmans, Dengxin Dai, and Luc Van Gool. 2021. Multi-task learning for dense prediction tasks: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 44(7):3614–3633.
- Mukund Varma T, Xuxi Chen, Zhenyu Zhang, Tianlong Chen, Subhashini Venugopalan, and Zhangyang Wang. 2022. Sparse winning tickets are data-efficient image recognizers. *Advances in Neural Information Processing Systems*, 35:4652–4666.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017a. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017b. [Attention is All you Need](#). In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

- Raja Vavekanand and Kira Sam. 2024. Llama 3.1: An in-depth analysis of the next-generation large language model. In -.
- Bastiaan S Veeling, Jasper Linmans, Jim Winkens, Taco Cohen, and Max Welling. 2018. Rotation equivariant cnns for digital pathology. In *Medical Image Computing and Computer Assisted Intervention—MICCAI 2018: 21st International Conference, Granada, Spain, September 16-20, 2018, Proceedings, Part II 11*, pages 210–218. Springer.
- Jesse Vig, Sebastian Gehrmann, Yonatan Belinkov, Sharon Qian, Daniel Nevo, Yaron Singer, and Stuart Shieber. 2020. Investigating gender bias in language models using causal mediation analysis. *Advances in neural information processing systems*, 33:12388–12401.
- Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. 2015. Show and tell: A neural image caption generator. In *CVPR*.
- Denny Vrande‘i and Markus Kr‘tztzsch. 2014. Wikidata: A free collaborative knowledge base. *Communications of the ACM*, 57(10):78–85.
- Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011. *California Institute of Technology*.
- Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019a. [Glue: A multi-task benchmark and analysis platform for natural language understanding](#).
- Haofan Wang, Zifan Wang, Mengnan Du, Fan Yang, Zijian Zhang, Xia Hu Ding, Piotr Mardziel, and Xia Hu. 2020a. Score-cam: Score-weighted visual explanations for convolutional neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 24–25.
- Haohan Wang, Songwei Ge, Zachary Lipton, and Eric P Xing. 2019b. Learning robust global representations by penalizing local predictive power. *Advances in Neural Information Processing Systems*, 32.
- Kevin Wang, Alexandre Variengien, Arthur Conmy, Buck Shlegeris, and Jacob Steinhardt. 2022. Interpretability in the wild: a circuit for indirect object identification in gpt-2 small. *arXiv preprint arXiv:2211.00593*.
- Lean Wang, Lei Li, Damai Dai, Deli Chen, Hao Zhou, Fandong Meng, Jie Zhou, and Xu Sun. 2023. [Label Words are Anchors: An Information Flow Perspective for Understanding In-Context Learning](#). ArXiv:2305.14160 [cs].
- Peng Wang, Dongyang Liu, Hui Li, and Qi Wu. 2020b. Give me something to eat: Referring expression comprehension with commonsense knowledge. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 28–36.

- Peng Wang, Qi Wu, Jiewei Cao, Chunhua Shen, Lianli Gao, and Anton van den Hengel. 2019c. Neighbourhood watch: Referring expression comprehension via language-guided graph attention networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1960–1968.
- Ruize Wang, Duyu Tang, Nan Duan, Zhongyu Wei, Xuanjing Huang, Guihong Cao, Daxin Jiang, Ming Zhou, et al. 2020c. K-adapter: Infusing knowledge into pre-trained models with adapters. *arXiv preprint arXiv:2002.01808*.
- Xiang Wang, Dingxian Wang, Canran Xu, Xiangnan He, Yixin Cao, and Tat-Seng Chua. 2019d. Explainable reasoning over knowledge graphs for recommendation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 5329–5336.
- Yuqing Wang and Yun Zhao. 2023. Gemini in reasoning: Unveiling commonsense in multimodal large language models. *arXiv preprint arXiv:2312.17661*.
- Zifeng Wang, Shao-Lun Huang, Ercan E Kuruoglu, Jimeng Sun, Xi Chen, and Yefeng Zheng. 2021a. Pac-bayes information bottleneck. *arXiv preprint arXiv:2109.14509*.
- Zifeng Wang, Shao-Lun Huang, Ercan Engin Kuruoglu, Jimeng Sun, Xi Chen, and Yefeng Zheng. 2021b. Pac-bayes information bottleneck. *ArXiv*, abs/2109.14509.
- Zirui Wang, Zachary C. Lipton, and Yulia Tsvetkov. 2020d. [On negative interference in multilingual models: Findings and a meta-learning treatment](#). *EMNLP 2020 - 2020 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, pages 4438–4450. ArXiv: 2010.03017 ISBN: 9781952148606.
- Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. [Chain-of-thought prompting elicits reasoning in large language models](#). *Advances in Neural Information Processing Systems*, 35:24824–24837.
- Nathaniel Weir, Adam Poliak, and Benjamin Van Durme. 2020. Probing neural language models for human tacit assumptions. *arXiv preprint arXiv:2004.04877*.
- Wei Wen, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. 2016. Learning structured sparsity in deep neural networks. *Advances in neural information processing systems*, 29.
- Chunyang Wu, Mark JF Gales, Anton Ragni, Penny Karanasou, and Khe Chai Sim. 2017. Improving interpretability and regularization in deep learning. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 26(2):256–265.
- Muling Wu, Wenhao Liu, Xiaohua Wang, Tianlong Li, Changze Lv, Zixuan Ling, Jianhao Zhu, Cenyuan Zhang, Xiaoqing Zheng, and Xuanjing Huang. 2024a. [Advancing Parameter Efficiency in Fine-tuning via Representation Editing](#). ArXiv:2402.15179 [cs].

- Muling Wu, Wenhao Liu, Xiaohua Wang, Tianlong Li, Changze Lv, Zixuan Ling, Jianhao Zhu, Cenyuan Zhang, Xiaoqing Zheng, and Xuanjing Huang. 2024b. [Advancing parameter efficiency in fine-tuning via representation editing](#). *arXiv:2402.15179*.
- Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google’s neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.
- Zhengxuan Wu, Aryaman Arora, Zheng Wang, Atticus Geiger, Dan Jurafsky, Christopher D. Manning, and Christopher Potts. 2024c. [ReFT: Representation Finetuning for Language Models](#). *ArXiv:2404.03592*.
- Jianxiong Xiao, James Hays, Krista A Ehinger, Aude Oliva, and Antonio Torralba. 2010. Sun database: Large-scale scene recognition from abbey to zoo. In *2010 IEEE computer society conference on computer vision and pattern recognition*, pages 3485–3492. IEEE.
- Enze Xie et al. 2021. Segformer: Simple and efficient design for semantic segmentation with transformers. *Advances in Neural Information Processing Systems*.
- Biao Xu and Guanci Yang. 2025. [Interpretability research of deep learning: A literature survey](#). *Information Fusion*, 115:102721.
- Dan Xu, Wanli Ouyang, Xiaogang Wang, and Nicu Sebe. 2018. Pad-net: Multi-tasks guided prediction-and-distillation network for simultaneous depth estimation and scene parsing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 675–684.
- Runxin Xu, Fuli Luo, Zhiyuan Zhang, Chuanqi Tan, Baobao Chang, Songfang Huang, and Fei Huang. 2021. Raise a child in large language model: Towards effective and generalizable fine-tuning. *arXiv preprint arXiv:2109.05687*.
- Shicheng Xu, Liang Pang, Yunchang Zhu, Huawei Shen, and Xueqi Cheng. 2024. Cross-modal safety mechanism transfer in large vision-language models. *arXiv preprint arXiv:2410.12662*.
- Hsiu-Yu Yang and Carina Silberer. 2022. Are visual-linguistic models commonsense knowledge bases? In *Proceedings of the 29th international conference on computational linguistics*, pages 5542–5559.
- Shenghao Yang, Weizhi Ma, Peijie Sun, Min Zhang, Qingyao Ai, Yiqun Liu, and Mingchen Cai. 2024. Common sense enhanced knowledge-based recommendation with large language model. In *International Conference on Database Systems for Advanced Applications*, pages 381–390. Springer.

- Shuo Yang, Siwen Luo, Soyeon Caren Han, and Eduard Hovy. 2025. Magic-vqa: Multimodal and grounded inference with commonsense knowledge for visual question answering. *arXiv preprint arXiv:2503.18491*.
- Guy Yariv, Idan Schwartz, Yossi Adi, and Sagie Benaim. 2024. Improving visual commonsense in language models via multiple image generation. *arXiv preprint arXiv:2406.13621*.
- Shuquan Ye, Yujia Xie, Dongdong Chen, Yichong Xu, Lu Yuan, Chenguang Zhu, and Jing Liao. 2023. Improving commonsense in vision-language models via knowledge graph riddles. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2634–2645.
- LU Ying et al. 2015. Decision tree methods: applications for classification and prediction. *Shanghai archives of psychiatry*, 27(2):130.
- Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. 2014. How transferable are features in deep neural networks? *Advances in neural information processing systems*, 27.
- Licheng Yu, Zhe Lin, Xiaohui Shen, Jimei Yang, Xin Lu, Mohit Bansal, and Tamara L Berg. 2018. Mattnet: Modular attention network for referring expression comprehension. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1307–1315.
- Licheng Yu, Patrick Poirson, Shan Yang, Alexander C Berg, and Tamara L Berg. 2016. Modeling context in referring expressions. In *European Conference on Computer Vision*, pages 69–85. Springer.
- Licheng Yu, Hao Tan, Mohit Bansal, and Tamara L Berg. 2017. A joint speaker-listener-reinforcer model for referring expressions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7282–7290.
- Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. 2020. Gradient surgery for multi-task learning. *Advances in Neural Information Processing Systems*, 33:5824–5836.
- Lu Yuan, Lijuan Chen, Weizhu Chen, et al. 2021. Florence: A new foundation model for computer vision. In *Advances in Neural Information Processing Systems*.
- Tian Yun, Chen Sun, and Ellie Pavlick. 2021. Does vision-and-language pretraining improve lexical grounding? *arXiv preprint arXiv:2109.10246*.
- Elad Ben Zaken, Shauli Ravfogel, and Yoav Goldberg. 2021. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. *arXiv preprint arXiv:2106.10199*.

- Arman Zarei, Keivan Rezaei, Samyadeep Basu, Mehrdad Saberi, Mazda Moayeri, Priyatham Kattakinda, and Soheil Feizi. 2024. Understanding and mitigating compositional issues in text-to-image generative models. *arXiv preprint arXiv:2406.07844*.
- Matthew D Zeiler and Rob Fergus. 2014. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer.
- Rowan Zellers, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019a. From recognition to cognition: Visual commonsense reasoning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6720–6731.
- Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019b. [HellaSwag: Can a machine really finish your sentence?](#) *arXiv:1905.07830*.
- Xiaohua Zhai, Alexander Kolesnikov, Neil Houlsby, and Lucas Beyer. 2022. Scaling vision transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12104–12113.
- Xiaohua Zhai, Joan Puigcerver, Alexander Kolesnikov, Pierre Ruysen, Carlos Riquelme, Mario Lucic, Josip Djolonga, Andre Susano Pinto, Maxim Neumann, Alexey Dosovitskiy, et al. 2019. A large-scale study of representation learning with the visual task adaptation benchmark. *arXiv preprint arXiv:1910.04867*.
- Biao Zhang, Ankur Bapna, Rico Sennrich, and Orhan Firat. 2021a. Share or not? learning to schedule language-specific capacity for multilingual translation. In *Ninth International Conference on Learning Representations 2021*.
- Hanwang Zhang, Yulei Niu, and Shih-Fu Chang. 2018a. Grounding referring expressions in images by variational context. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4158–4166.
- Q Zhang, M Chen, A Bukharin, P He, Y Cheng, W Chen, and T Zhao. 2023a. Adaptive budget allocation for parameter-efficient fine-tuning. preprint (2023). *arXiv preprint arXiv:2303.10512*.
- Renrui Zhang, Rongyao Fang, Wei Zhang, Peng Gao, Kunchang Li, Jifeng Dai, Yu Qiao, and Hongsheng Li. 2021b. Tip-adapter: Training-free clip-adapter for better vision-language modeling. *arXiv preprint arXiv:2111.03930*.
- Renrui Zhang, Ziyu Guo, Wei Zhang, Kunchang Li, Xupeng Miao, Bin Cui, Yu Qiao, Peng Gao, and Hongsheng Li. 2022a. Pointclip: Point cloud understanding by clip. In *CVPR 2022*.
- Renrui Zhang, Jiaming Han, Aojun Zhou, Xiangfei Hu, Shilin Yan, Pan Lu, Hongsheng Li, Peng Gao, and Yu Qiao. 2023b. Llama-adapter: Efficient fine-tuning of language models with zero-init attention. *ICLR 2024*.

- Renrui Zhang, Xiangfei Hu, Bohao Li, Siyuan Huang, Hanqiu Deng, Hongsheng Li, Yu Qiao, and Peng Gao. 2023c. Prompt, generate, then cache: Cascade of foundation models makes strong few-shot learners. *CVPR 2023*.
- Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. 2022b. [OPT: Open Pre-trained Transformer Language Models](#). ArXiv:2205.01068 [cs].
- Xiaofeng Zhang, Chen Shen, Xiaosong Yuan, Shaotian Yan, Liang Xie, Wenxiao Wang, Chaochen Gu, Hao Tang, and Jieping Ye. 2024a. [From Redundancy to Relevance: Enhancing Explainability in Multimodal Large Language Models](#). ArXiv:2406.06579 [cs].
- Yu Zhang, Peter Ti'o, Ale' Leonardis, and Ke Tang. 2021c. [A Survey on Neural Network Interpretability](#). *IEEE Transactions on Emerging Topics in Computational Intelligence*, 5(5):726–742. ArXiv:2012.14261 [cs].
- Yu Zhang and Qiang Yang. 2021. A survey on multi-task learning. *IEEE Transactions on Knowledge and Data Engineering*, 34(12):5586–5609.
- Yunlong Zhang, Nan Chen, Yonghe Wang, Xiangdong Su, and Feilong Bao. 2025. Multilingual parameter-sharing adapters: A method for optimizing low-resource neural machine translation. In *ICASSP 2025-2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE.
- Zhenyu Zhang, Zhen Cui, Chunyan Xu, Zequn Jie, Xiang Li, and Jian Yang. 2018b. Joint task-recursive learning for semantic segmentation and depth estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 235–251.
- Zhenyu Zhang, Zhen Cui, Chunyan Xu, Yan Yan, Nicu Sebe, and Jian Yang. 2019. Pattern-affinitive propagation across depth, surface normal and semantic segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4106–4115.
- Zhi Zhang, Qizhe Zhang, Zijun Gao, Renrui Zhang, Ekaterina Shutova, Shiji Zhou, and Shanghang Zhang. 2023d. Gradient-based parameter selection for efficient fine-tuning. *arXiv preprint arXiv:2312.10136*.
- Zhi Zhang, Qizhe Zhang, Zijun Gao, Renrui Zhang, Ekaterina Shutova, Shiji Zhou, and Shanghang Zhang. 2024b. Gradient-based parameter selection for efficient fine-tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 28566–28577.

- Mengjie Zhao, Tao Lin, Fei Mi, Martin Jaggi, and Hinrich Schütze. 2020. Masking as an efficient alternative to finetuning for pretrained language models. *arXiv preprint arXiv:2004.12406*.
- Qinyu Zhao, Ming Xu, Kartik Gupta, Akshay Asthana, Liang Zheng, and Stephen Gould. 2024. The first to know: How token distributions reveal hidden knowledge in large vision-language models? *arXiv preprint arXiv:2403.09037*.
- Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Li, and OTHERS. 2023. [Judging llm-as-a-judge with mt-bench and chatbot arena](#).
- Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. 2016. Learning deep features for discriminative localization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2921–2929.
- Daquan Zhou, Bingyi Kang, Xiaojie Jin, Linjie Yang, Xiaochen Lian, Zihang Jiang, Qibin Hou, and Jiashi Feng. 2021. Deepvit: Towards deeper vision transformer. *arXiv preprint arXiv:2103.11886*.
- Daquan Zhou, Zhiding Yu, Enze Xie, Chaowei Xiao, Animashree Anandkumar, Jiashi Feng, and Jose M Alvarez. 2022. Understanding the robustness in vision transformers. In *International Conference on Machine Learning*, pages 27378–27394. PMLR.
- Xiangyang Zhu, Renrui Zhang, Bowei He, Aojun Zhou, Dong Wang, Bin Zhao, and Peng Gao. 2023. Not all features matter: Enhancing few-shot clip with adaptive prior refinement. *ICCV 2023*.
- Roland S Zimmermann, Thomas Klein, and Wieland Brendel. 2023. Scale alone does not improve mechanistic interpretability in vision models. *Advances in Neural Information Processing Systems*, 36:57876–57907.

Samenvatting

Op het gebied van multimodaal leren is de afgelopen jaren aanzienlijke vooruitgang geboekt, zowel wat betreft bereikte prestaties, als de verscheidenheid van taken. Echter blijven er nog steeds veel belangrijke uitdagingen over, zoals het integreren van externe kennis in multimodale modellen, wat snelle en efficiënte aanpassing aan nieuwe taken mogelijk maakt, en negatieve interferentie verminderd tijdens gezamenlijk leren. Bovendien, ondanks de brede toepasbaarheid van deze modellen, ontbreekt er inzicht in hun interne mechanisme, met name in hoe verschillende modaliteiten interageren in multimodale systemen. Dit proefschrift onderzoekt deze uitdagingen vanuit het perspectief van twee verschillende modaliteiten: taal en beeld, en hun combinatie.

Ten eerste ontwikkelen we een methode om multimodaal redeneren te verbeteren door algemene kennis te integreren in de modellen. Dat wil zeggen, we integreren algemene kennis over objecten in een afbeelding in de representaties van die objecten. We evalueren deze techniek op basis van de "referring expression comprehension" taak, waarbij het doel is om de positie van het object in de afbeelding te vinden op basis van een taalbeschrijving en een afbeelding. Door externe kennis op te nemen, kunnen de modellen objectfuncties en contextuele relaties afleiden, waardoor ze kunnen redeneren over complexe scènes, en niet alleen visuele en ruimtelijke aanwijzingen waar nemen. Dit verbetert de toepasbaarheid van modellen voor realistische scenario's die "gezond verstand" vereisen.

Ten tweede stellen we een nieuwe "parameter-efficient fine-tuning" (PEFT) methode voor, waarmee al getrainde modellen efficiënt aangepast kunnen worden aan nieuwe taken. In plaats van het hele model aan te passen, verfijnt onze aanpak slechts een klein, maar relevant deel van alle parameters. Hierdoor worden neuron excitaties gemoduleerd om zich effectief aan te passen aan nieuwe taken. Deze selectieve afstemming verlaagt de vereiste computerkracht, vermindert potentiële gradiëntconflicten tussen verschillende taken, en behoudt het model nuttige kennis die is opgedaan tijdens de eerste trainingsronde. We tonen aan dat onze methode de prestaties in verschillende visuele en taalkundige taken verbetert. Daarnaast introduceren we een nieuwe "sparse training"-benadering waarmee voorgetrainde modellen meerdere taken tegelijk kunnen

uitvoeren. Deze methode vergemakkelijkt het delen van relevante informatie tussen taken tijdens het leerproces en vermindert tegelijkertijd gradiëntconflicten die vaak voorkomen bij het gezamenlijk leren van meerdere taken. We tonen proefondervindelijk aan dat onze aanpak de prestaties aanzienlijk verbetert bij “dense vision prediction” taken, wat wijst op foutbestendigheid en brede toepasbaarheid.

Ten slotte onderzoeken we het interne werkingsmechanisme van multimodale “Large Language Models” (MLLMs) bij het uitvoeren van multimodale taken, met name hoe taalkundige en visuele informatie in deze modellen op elkaar inwerken. Specifiek, gegeven een afbeelding en een vraag, onderzoeken we waar in het model en hoe beeld en tekst worden gecombineerd om de uiteindelijke voorspelling te genereren. Uitgebreide experimenten tonen aan dat er twee verschillende stadia zijn in het integratie proces van deze twee modaliteiten. In de onderste lagen zet het model eerst de meer algemene visuele kenmerken van het hele beeld om in de representaties van (tekstuele) vraagwoorden. In de middelste lagen wordt visuele informatie over specifieke objecten die relevant zijn voor de vraag nogmaals overgedragen naar de juiste woordenposities van de vraag. Tot slot wordt in de hogere lagen de resulterende multimodale representatie doorgegeven aan de laatste positie van de invoerreeks voor de uiteindelijke voorspelling. Onze bevindingen bieden, over het geheel genomen, een nieuw en uitgebreid perspectief op ruimtelijke en functionele aspecten van beeld- en taalverwerking in MLLMs, wat toekomstig onderzoek naar multimodale informatie lokalisatie en bewerking zal bevorderen.

Globaal samengevat, dit proefschrift draagt bij aan de vooruitgang van beeld- en taalmodellen door fundamentele probleemstellingen in deze domeinen aan te pakken. De voorgestelde oplossingen en inzichten bieden een basis voor het ontwikkelen van meer kennisbewuste, efficiënte en interpreteerbare multimodale kunstmatige intelligentie systemen, toepasbaar in uiteenlopende realistische contexten.

Abstract

The area of multimodal learning has seen substantial advances in recent years, both in terms of performance and the variety of tasks tackled. However, many important challenges remain, including integrating external factual knowledge in multimodal models, enabling their fast and efficient adaptation to a new task and reducing negative interference in joint learning. Furthermore, despite the wide applicability of these models, we still lack an understanding of their internal mechanisms, in terms of how different modalities interact in multi-modal systems. This dissertation investigates these challenges from the perspectives of two different modalities: language and vision, as well as their combination.

First, we develop a method to enhance multimodal reasoning by integrating commonsense knowledge into the models. Specifically, we incorporate the commonsense knowledge about objects in the image into model representations of these objects. We evaluate this technique in the task of referring expression comprehension, where the aim is to find the position of the object in the image from a language description and an image. By incorporating external knowledge, the models can infer object functions and contextual relationships, enabling them to reason about complex scenes rather than merely perceiving visual and spatial cues, improving the applicability of models to real-world scenarios requiring commonsense knowledge.

Second, we propose a novel parameter-efficient fine-tuning (PEFT) method for an efficient adaptation of pre-trained models to new tasks. Instead of updating the entire model, our approach fine-tunes only a small, relevant subset of parameters, effectively modulating neuron activations to adapt to new tasks. This selective tuning reduces computational demands and potentially reduces gradient conflicts between tasks and preserves useful knowledge acquired during pretraining. We demonstrate that our method improves performance across various vision and language tasks. On the other hand, to enable pretrained models to efficiently handle multiple tasks at the same time, we propose a novel sparse training approach. This method facilitates the sharing of relevant information across tasks during the learning process while simultaneously mitigating gradient conflicts that commonly arise in multi-task learning. Empirical

evaluations demonstrate that our approach significantly improves performance in dense vision prediction tasks, demonstrating robustness and wide applicability.

Lastly, we investigate the inner working mechanism of multimodal large language models (MLLMs) when performing multimodal tasks, especially how linguistic and visual information interact within these models. Specifically, given an image-question pair as input, we investigate where in the model and how the visual and linguistic information are combined to generate the final prediction. Extensive experiments have shown that there are two distinct stages in the process of integration of the two modalities. In the lower layers, the model first transfers the more general visual features of the whole image into the representations of (linguistic) question tokens. In the middle layers, it once again transfers visual information about specific objects relevant to the question to the respective token positions of the question. Finally, in the higher layers, the resulting multimodal representation is propagated to the last position of the input sequence for the final prediction. Overall, our findings provide a new and comprehensive perspective on the spatial and functional aspects of image and language processing in the MLLMs, thereby facilitating future research into multimodal information localization and editing.

In summary, this dissertation contributes to advancing vision and language models by addressing fundamental challenges across these domains. The proposed solutions and insights provide a foundation for developing more knowledge-aware, efficient, and interpretable multi-modal AI systems, applicable to diverse tasks in real-world settings.

Titles in the ILLC Dissertation Series:

ILLC DS-2020-12: **Bastiaan van der Weij**

Experienced listeners: Modeling the influence of long-term musical exposure on rhythm perception

ILLC DS-2020-13: **Thom van Gessel**

Questions in Context

ILLC DS-2020-14: **Gianluca Grilletti**

Questions & Quantification: A study of first order inquisitive logic

ILLC DS-2020-15: **Tom Schoonen**

Tales of Similarity and Imagination. A modest epistemology of possibility

ILLC DS-2020-16: **Ilaria Canavotto**

Where Responsibility Takes You: Logics of Agency, Counterfactuals and Norms

ILLC DS-2020-17: **Francesca Zaffora Blando**

Patterns and Probabilities: A Study in Algorithmic Randomness and Computable Learning

ILLC DS-2021-01: **Yfke Dulek**

Delegated and Distributed Quantum Computation

ILLC DS-2021-02: **Elbert J. Booij**

The Things Before Us: On What it Is to Be an Object

ILLC DS-2021-03: **Seyyed Hadi Hashemi**

Modeling Users Interacting with Smart Devices

ILLC DS-2021-04: **Sophie Arnoult**

Adjunction in Hierarchical Phrase-Based Translation

ILLC DS-2021-05: **Cian Guilfoyle Chartier**

A Pragmatic Defense of Logical Pluralism

ILLC DS-2021-06: **Zoi Terzopoulou**

Collective Decisions with Incomplete Individual Opinions

ILLC DS-2021-07: **Anthia Solaki**

Logical Models for Bounded Reasoners

ILLC DS-2021-08: **Michael Sejr Schlichtkrull**

Incorporating Structure into Neural Models for Language Processing

ILLC DS-2021-09: **Taichi Uemura**

Abstract and Concrete Type Theories

- ILLC DS-2021-10: **Levin Hornischer**
Dynamical Systems via Domains: Toward a Unified Foundation of Symbolic and Non-symbolic Computation
- ILLC DS-2021-11: **Sirin Botan**
Strategyproof Social Choice for Restricted Domains
- ILLC DS-2021-12: **Michael Cohen**
Dynamic Introspection
- ILLC DS-2021-13: **Dazhu Li**
Formal Threads in the Social Fabric: Studies in the Logical Dynamics of Multi-Agent Interaction
- ILLC DS-2021-14: **Ílvaro Piedrafitra**
On Span Programs and Quantum Algorithms
- ILLC DS-2022-01: **Anna Bellomo**
Sums, Numbers and Infinity: Collections in Bolzano's Mathematics and Philosophy
- ILLC DS-2022-02: **Jan Czakowski**
Post-Quantum Security of Hash Functions
- ILLC DS-2022-03: **Sonia Ramotowska**
Quantifying quantifier representations: Experimental studies, computational modeling, and individual differences
- ILLC DS-2022-04: **Ruben Brokkelkamp**
How Close Does It Get?: From Near-Optimal Network Algorithms to Suboptimal Equilibrium Outcomes
- ILLC DS-2022-05: **Lwenn Bussi're-Carae**
No means No! Speech Acts in Conflict
- ILLC DS-2022-06: **Emma Mojet**
Observing Disciplines: Data Practices In and Between Disciplines in the 19th and Early 20th Centuries
- ILLC DS-2022-07: **Freek Gerrit Witteveen**
Quantum information theory and many-body physics
- ILLC DS-2023-01: **Subhasree Patro**
Quantum Fine-Grained Complexity
- ILLC DS-2023-02: **Arjan Cornelissen**
Quantum multivariate estimation and span program algorithms

- ILLC DS-2023-03: **Robert Paʻmann**
Logical Structure of Constructive Set Theories
- ILLC DS-2023-04: **Samira Abnar**
Inductive Biases for Learning Natural Language
- ILLC DS-2023-05: **Dean McHugh**
Causation and Modality: Models and Meanings
- ILLC DS-2023-06: **Jialiang Yan**
Monotonicity in Intensional Contexts: Weakening and: Pragmatic Effects under Modals and Attitudes
- ILLC DS-2023-07: **Yiyan Wang**
Collective Agency: From Philosophical and Logical Perspectives
- ILLC DS-2023-08: **Lei Li**
Games, Boards and Play: A Logical Perspective
- ILLC DS-2023-09: **Simon Rey**
Variations on Participatory Budgeting
- ILLC DS-2023-10: **Mario Giulianelli**
Neural Models of Language Use: Studies of Language Comprehension and Production in Context
- ILLC DS-2023-11: **Guillermo Menʻndez Turata**
Cyclic Proof Systems for Modal Fixpoint Logics
- ILLC DS-2023-12: **Ned J.H. Wontner**
Views From a Peak: Generalisations and Descriptive Set Theory
- ILLC DS-2024-01: **Jan Rooduijn**
Fragments and Frame Classes: Towards a Uniform Proof Theory for Modal Fixed Point Logics
- ILLC DS-2024-02: **Bas Cornelissen**
Measuring musics: Notes on modes, motifs, and melodies
- ILLC DS-2024-03: **Nicola De Cao**
Entity Centric Neural Models for Natural Language Processing
- ILLC DS-2024-04: **Ece Takmaz**
Visual and Linguistic Processes in Deep Neural Networks: A Cognitive Perspective
- ILLC DS-2024-05: **Fatemeh Seifan**
Coalgebraic fixpoint logic Expressivity and completeness result

- ILLC DS-2024-06: **Jana Sot'kov**
Isogenies and Cryptography
- ILLC DS-2024-07: **Marco Degano**
Indefinites and their values
- ILLC DS-2024-08: **Philip Verduyn Lunel**
Quantum Position Verification: Loss-tolerant Protocols and Fundamental Limits
- ILLC DS-2024-09: **Rene Allerstorfer**
Position-based Quantum Cryptography: From Theory towards Practice
- ILLC DS-2024-10: **Willem Feijen**
Fast, Right, or Best? Algorithms for Practical Optimization Problems
- ILLC DS-2024-11: **Daira Pinto Prieto**
Combining Uncertain Evidence: Logic and Complexity
- ILLC DS-2024-12: **Yanlin Chen**
On Quantum Algorithms and Limitations for Convex Optimization and Lattice Problems
- ILLC DS-2024-13: **Jaap Jumelet**
Finding Structure in Language Models
- ILLC DS-2025-01: **Julian Chingoma**
On Proportionality in Complex Domains
- ILLC DS-2025-02: **Dmitry Grinko**
Mixed Schur-Weyl duality in quantum information
- ILLC DS-2025-03: **Rochelle Choenni**
Multilinguality and Multiculturalism: Towards more Effective and Inclusive Neural Language Models
- ILLC DS-2025-04: **Aleksi Anttila**
Not Nothing: Nonemptiness in Team Semantics
- ILLC DS-2025-05: **Niels M. P. Neumann**
Adaptive Quantum Computers: decoding and state preparation