Walking the Graph of Language:
On a Framework for Meaning and Analogy

MSc Thesis *(Afstudeerscriptie)*

written by

Nal Emmerich Kalchbrenner
(born December 11th, 1987 in Lugano, Switzerland)

under the supervision of **Dr Reinhard Blutner** and **Dr Raquel Fernandez Rovira**, and submitted to the Board of Examiners in partial fulfillment of the requirements for the degree of

MSc in Logic

at the *Universiteit van Amsterdam*.

Date of the public defense:	Members of the Thesis Committee:
September 7th, 2012	Prof Dr Johan Bos
	Dr Reinhard Blutner
	Dr Raquel Fernandez Rovira
	Prof Dr Benedikt Löwe
	Dr Jelle Zuidema

INSTITUTE FOR LOGIC, LANGUAGE AND COMPUTATION

Acknowledgements

I am indebted to Grisha Mints, Luca Trevisan, Harry Buhrman, Bruno Loff and Ronald de Wolf for teaching me on the concepts of computation and algorithm in general; I am grateful to Ronald de Wolf for introducing me into the mathematics of the quantum.

I am indebted to Michiel van Lambalgen for spurning me on to the computational study of language, for his lectures on Kant and for his readiness to help in many a circumstance.

I am indebted to Dick de Jongh for his lectures on logic and for his attentive and steady mentoring, without which many things present and future would not be possible.

I am indebted to Raquel Fernandez and Reinhard Blutner for their supervision of the present thesis, insightful questions and remarks, and encouragement and support throughout. I am thankful to the other members of the thesis committee, Johan Bos, Benedikt Löwe and Jelle Zuidema, for reviewing the thesis.

I am grateful to my friend and companion of philosophical journeys Dāvis Ozols for innumerable walks and conversations.

Above all, I am grateful for things untold to my family and close fellows.

Abstract

We introduce a computational framework for generating *representations* of linguistic *concepts*. The concepts we consider are the *meanings* of words and the *verbal analogs* corresponding to n-tuples of words. Representations of meanings can be compared to estimate their *degree of synonymy*. Likewise, representations of verbal analogs can be compared to estimate the *strength of the analogy* between them. The framework automatically constructs from a corpus of language large graphs with *words* as vertices and *conceptual connections* as edges; these graphs are dubbed *word-graphs*. Focusing on representations of verbal analogs of word *pairs*, we present two main algorithms for the extraction of such representations from a word-graph. One algorithm relies on *path distance measures* and *random walks* over the word-graph. The other algorithm relies on *spreading activation* and *algebraic vector operations*. Tested on a standardized set of verbal analogy problems, one of the algorithms attains accuracy that is statistically not significantly different from the state-of-the-art. Further, the experiments yield a novel theoretical insight into the workings of verbal analogy and its representation.

Table of Contents

1 Preface

The computational study of language investigates the *effective* generation of *operational* mathematical representations of *linguistic information*. By effective generation, we mean that a *feasible algorithm* is specified that computes the representation of the intended linguistic information, given as input data of varying sort. By an operational representation, we mean one that may be feasibly operated on or computed with. *Types* of linguistic information include, among multiple others, the *syntactic* analysis of a sentence, the *meaning* of a word or phrase within or without a discourse, or the content of a paragraph or document.

The aim of generating such representations is twofold. On the one hand, the representations may be adopted to allow computing machines to *process* and *interpret* the linguistic information, thus allowing them to perform to a certain degree of *accuracy* a large variety of linguistic tasks. Examples range from using representations of syntactic analyses of sentences in *translating* from one language to another [8], to adopting representations of the meaning of phrases in *retrieving* documents that are relevant to the phrase [29]. Future uses of sufficiently accurate representations are potentially very extensive and may include, for instance, general linguistic interactions between users and machines.

On the other hand, the representations and the algorithms that generate them may yield *findings* and *insight* into the type of linguistic information itself. If the representations resulting from one algorithm capture, according to the experimental setting, the intended type of linguistic information *more accurately* than those resulting from another algorithm, then an appraisal of the differences between the two algorithms may increase our understanding of the particular type of linguistic information. More specifically, if an algorithm $\mathcal{A}$ is more accurate than an algorithm $\mathcal{A}'$, where the latter results from the former by a small, but significant variation, then *what* is varied and the *way* it is varied may be informative as to the linguistic phenomenon. [1]

We here present a framework in which one generates representations for the *meaning* of words and representations for the *verbal analog* of n-tuples of words. A *verbal analog* of an n-tuple of words is an agglomerate of *concepts* and of

[1] A notable example is [28].

relations between the concepts, where each concept is given by the *meaning* of one of the words in the n-tuple; a verbal analog of a single word is just its meaning.[2] One compares representations of meanings by measuring their degree of *synonymy*. One compares representations of verbal analogs by measuring their degree of *analogical strength*. The degree of analogical strength of two verbal analogs given by *single* words is just the degree of synonymy of the two words.

One generates representations in the framework in two main steps. First, on the basis of a corpus of language one constructs a *word-graph*, that is a graph with words at its vertices and *conceptual role connections* as directed weighted edges between the vertices (Sect. 4). Then, the information incorporated in certain *regions* of the word-graph is taken to correspond to the meaning or verbal analog of certain words. Focusing on the verbal analog given by a *pair* of words, we present two main algorithms $\mathcal{R}$ and $\mathcal{S}$ for extracting *vectorial representations* of the information within the selected regions (Sect. 5).[3] The vectorial representations yield in turn the desired representation of the verbal analog of a pair.

Different theoretical *insights* on analogy induce the selection of different regions for the representation of the verbal analog of the pair of words. There are two types of regions, called respectively *meaning* regions and *relation* regions (Sect. 5). If the verbal analog is taken to be a result of just the *meanings* of the two words, as e.g. in the Model of Analogical Reasoning (MAR) [35], then one selects the corresponding *meaning* regions and assigns the verbal analog representation to be an *algebraic operation* of the vectorial representations of the meaning regions. If the verbal analog is taken to be the result of the *relations* that hold between the two words, as suggested by the Structure Mapping Theory (SMT) [18] and implemented in Latent Relational Analogy (LRA) [36] and Distributional Memory (DM) [2], then one selects the *relation* region determined by the two words and lets the verbal analog representation be the vectorial representation of the relation region.

We investigate the accuracy of the framework and algorithms in a two-fold way, according to the two aims of representations suggested above (Sect. 6). The experimental setting is given by 374 verbal analogy problems from the SAT

[2] We return to this in Sect. 2.

[3] A third, hybrid algorithm $\mathcal{T}$ is also briefly presented, but no experimental evaluation is reported for it.

College Board exam [36]. On the one hand, we consider the overall accuracy of the framework across the two algorithms at the task of solving the verbal analogy problems. We see that the reported experiments yield an accuracy for the higher performing $\mathcal{S}$ algorithm that is significantly better than the $LexDM$ model (one of the three sub-models of DM, the other two being $DepDM$ and $TypeDM$), and is *not* significantly different from $DepDM$, LRA or the model with state-of-the-art accuracy $TypeDM$.[4] Since the basic way of harvesting connection weights in the present framework is rather close to that in $LexDM$ and $DepDM$, this shows that a combination of the graph-structure and semantic analysis (not present in $LexDM$ and $DepDM$) and of the $\mathcal{S}$ algorithm yields a significant improvement at least with respect to the $LexDM$ model. By contrast, LRA and $TypeDM$ have a specific way of harvesting such weights; the way of harvesting weights that is adopted in $TypeDM$ is also portable to the present framework (Sect. 7).

On the other hand, we consider the accuracies of the representations induced by the different theoretical insights on analogy. We see that the resulting accuracies do not yield a significant difference between the representations induced by MAR and those induced by SMT. Further, a novel theoretical insight emerges that yields representations that are significantly better than those induced by MAR. The novel theoretical insight coincides with the highest performing variant of the $\mathcal{S}$ algorithm. The insight incorporates both a crucial idea from MAR and one from SMT.

We proceed as follows. We start off in Sect. 2 by considering linguistic concepts such as *meanings* and *verbal analogs* with a view towards the framework. We also consider the notions of *synonymy* and *analogical strength* and how these are specified in the framework. In Sect. 3 we survey three foregoing models of verbal analogy underscoring the insights that go with them. In Sect. 4 we present the procedure for the construction of a word-graph. In Sect. 5 we present the two main algorithms $\mathcal{R}$ and $\mathcal{S}$, the former based on *path distance functions* and *random walks* and the latter on *spreading activation* and *algebraic operations*. In Sect. 6 we present the accuracy values for various specifications of the algorithms and consider the significance of these values as to the theoretical insights into analogy. Finally, in Sect. 7 we end by describing possible variations and extensions of the framework.

[4] All Fischer tests are reported in Sect. 6. The state-of-the-art is relative to the size of the corpus.

2 A Prelude of Words, Concepts and Representations

We here introduce some of the more fundamental notions underlying the theoretical and computational aspects of the framework. We begin with the *syntactic constructs* of the framework that will serve as *labels* to the computational objects. We continue by expounding on the informal, theoretical *interpretation* of the syntactic constructs in terms of linguistic *concepts*; concepts that are *word meanings* turn out to be special cases of concepts that are *verbal analogs*. We further remark on some features of such concepts. Then, we move on to the general computational *representation* of the constructs, thus relating central notions such as *vector representation* and *measure of vector similarity*. We conclude by extending the theoretical interpretation to the latter.

2.i Words and n-tuples of words

The syntactic *constructs* that will be assigned representations in the framework are *words* and *n-tuples of words*. By *word* we will mean the main form of a lexeme of the English language, also called a *lemma*; examples of *words* are 'aardvark', 'bright' or 'blossom'. By an *n-tuple of words* we mean a finite sequence of *words*, such as $\langle$ 'aardvark', 'nightly' $\rangle$ or $\langle$ 'bright', 'blossom', 'blue' $\rangle$. We identify a *word* w with the 1-tuple $\langle w \rangle$. Further, the notion of *word* is to be distinguished from that of *word token*, that is a possibly inflected instance of a corresponding word; thus, 'aardvarks' and 'blossomed' are word tokens of respectively 'aardvark' and 'blossom'. Let us briefly remark on these notions.

Any expression that appears in language and purports to convey linguistic information counts as a *word token*. For instance, expressions such as "probableee" and "lol" that occur in written language are word tokens that one may view as instances of the homograph *words*. Any indeterminateness is resolved by automatic procedures that extract words corresponding to word tokens occurring in a given corpus of language, as we see below. For now let us fix a collection $\mathcal{W}$ of all the relevant *words* and in turn determine precisely the syntactic constructs *word* and *n-tuple of words*. With this in mind, let us consider the informal, theoretical interpretation of the constructs.

2.ii Concepts of a linguistic ilk

The *constructs* stand for their *conceptual* counterparts. One naturally interprets a word w to stand for its *meaning*, i.e. the information that an ordinary speaker of the language deploys in concordance with the speaker's deployment of the word w.[5] We write $\mathbf{w}$ to point to the meaning of w and to distinguish meaning from the sequence of symbols w itself. Thus, 'aardvark' stands for **aardvark** and 'blossom' for **blossom**. We view the meaning of a word as a type of linguistic *concept*. Let us consider next the interpretation of word n-tuples.

An n-tuple $\langle w_1, ..., w_n \rangle$ stands for the corresponding *verbal analog*. A general *analog* is constituted of a commonly small collection of, possibly non-linguistic, *concepts* and *conceptual relations* between such concepts.[6] An example of a general analog is a *tree* with its *roots*, *trunk*, *branches*, *flowers*, *fruits* and *leaves* playing different roles in relation to each other. Another example is a set of *particles* with the relevant *equations* describing the relations between them. We call a *verbal analog* one in which the constituent concepts are given by the *meanings* of a specified collection of words. Thus, given two words in the form of a pair, e.g. ⟨ 'aardvark', 'Africa' ⟩, one considers the verbal analog given by the meanings of the two words, here **aardvark** and **Africa**, and the conceptual relations occurring between them, e.g. **being native to**. In this way we take a word n-tuple $\langle w_1, ..., w_n \rangle$ to stand for the corresponding *verbal analog* that we write $\langle \mathbf{w_1}, ..., \mathbf{w_n} \rangle$, where the conceptual relations occurring between the concepts are not explicitly given. We note that one may view *conceptual relations* themselves as just concepts, linguistic or otherwise, that happen to play a linking role in the context of a particular analog. More generally, we view an analog itself as a concept that in turn is an agglomerate of other concepts. Let us briefly comment on the connection between *meaning* and *verbal analog*.

The notion of *verbal analog* encompasses that of *meaning*. In Sect. 2.i we have identified the word w with the 1-tuple $\langle w \rangle$. Lifting the identity to that of the respective interpretations, one considers the meaning $\mathbf{w}$ to be just the verbal analog $\langle \mathbf{w} \rangle$ as implied by $\langle w \rangle = w$, where in $\langle \mathbf{w} \rangle$ no specific conceptual

[5] We do not attempt to give a definition of *meaning*; the present explication or the ordinary understanding of the notion are sufficient.

[6] Concepts may be of a visual, behavioral or musical sort, among multiple others. The use of the term *analog* here for a general collection of concepts foreshadows the study of *analogy* below.

relations are elicited. A verbal analog $\langle \mathbf{w_1}, ..., \mathbf{w_n} \rangle$ may thus be usefully viewed as a more encompassing notion than meaning, i.e. as a generalization of meaning to multiple words $\langle w_1, ..., w_n \rangle$; here the joint consideration of multiple meanings elicits the consideration of conceptual relations between them.

A final remark concerns analogs in connection to *analogy*. *Analogy* is the identification of correspondences between concepts and conceptual relations in two or more analogs [19,18,7]. *Verbal analogy* is similarly explicated with respect to *verbal* analogs. For example, one may find compelling the verbal analogy between ⟨**aardvark**, **Africa**⟩ and ⟨**emu**, **Australia**⟩. We return to analogy below. Let us consider next certain features of the connections between words and concepts that turn out to be significant in the framework.

2.iii Facets of conceptual behavior

Words may be *ambiguous* as to the concept they stand for. When looking at the map from words to meanings, it surely strikes one as one-to-many at best.[7] Notorious are instances such as 'bear', that may signify either the *woodland creature of drably fur* or the *enduring act of support*, among others. We frame this as the following remark:

> *Ambiguity of words*: A word w can stand for multiple distinct concepts $\mathbf{w^1}, ..., \mathbf{w^n}$.

Can one say more about the relation of w with respect to a single one of its meanings $\mathbf{w^i}$?

An additional remark concerns the *variableness* in the meanings of words. Let us illustrate the remark first with a parable.[8] Consider the Druids, a people, the parable goes, who left the shores of Great Britain some three centuries ago and settled on a desolate island in the South Sea. The Druids spoke a form of antique English as they left and kept speaking it throughout their generations of maritime isolation. On a sunny day not too long ago, some Druids started noticing grey and white shiny objects floating in the air as modern settlers were exploring the area. Having never heard of airplanes and having never envisaged engine-driven machines, the lucky few Druids cried out, "Look at those large birds flying over the tall trees". Further observations followed and word spread

[7] The map is also many-to-one as the same concept can be meant by differing words.
[8] The parable is adapted from [40].

through the jungly villages. As the awe of the news placated over the following days, ordinary Druids' conversations would comprise utterances such as, "birds have soft, brightly colored feathers and birds gain height by forcefully flapping their feathered wings" and "fetch some eggs from these birds' nests!", as well as, "some birds have drab but shiny bodies that seem to be made of steel" and "I wouldn't try lunching on one of those birds!". What is interesting about the parable is, to wit, the flawless and unnoticed expansion of the word 'bird' and its meaning **bird** into unforeseen linguistic territory.

A more contemporary example would involve the word 'phone'. A dictionary entry for 'phone' explicates its meaning **phone** in terms of a device for transmitting and receiving sounds.[9] Nonetheless, we all too often hear nowadays, "I texted her with my phone", "I used my phone to take a photograph of the event", "he played chess with his phone and it beat him". The meaning **phone** has clearly enlarged its patches of linguistic usage to cover devices whose main functions go beyond sound manipulations and the expansion has apparently gone unnoticed by dictionaries. A wealth of instances of expansions and modifications of meaning may be found not just in the recent history of gadgets, but also in the domains of colors, biology and engineering, among others.[10] We do not relate here further the quirky proclivities of meanings; we frame instead a corresponding remark as follows:

> *Variableness of meanings:* For a word w and one of the concepts $\mathbf{w^i}$ that w stands for, the concept $\mathbf{w^i}$ may vary, i.e. expand, shrink or modify, its patches of linguistic usage.

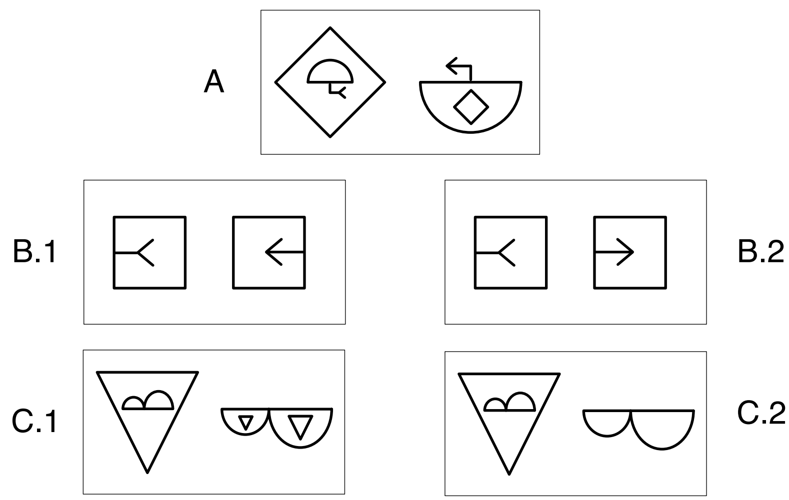
It is worthwhile to briefly compare the *ambiguity of words* with the *variableness of meanings*. One picture is as follows: as some meaning $\mathbf{w^i}$ of a word w varies over time and crystalizes over, say, two recognizably distinct patches of usage, $\mathbf{w^i}$ is "split" into $\mathbf{w^j}$ and $\mathbf{w^k}$ and the latter two are identified as distinct concepts that the word w ambiguously stands for. Thus, the Druids may eventually determine that the steel, shiny airplanes are usefully distinct in kind from the jungle avians and stipulate the word 'bird' as ambiguous between the two meanings; one may eventually do the same with the word 'phone'. This ends our

[9] The *Oxford Dictionary* entry for 'telephone' reads, "a system for transmitting voices over a distance using wire or radio, by converting acoustic vibrations to electrical signals."

[10] For a rich survey, see [40].

considerations of the relations between words and meanings. We inquire finally into the behavior of conceptual relations within verbal analogs.

Given an analog, the elicitation of a conceptual relation between two of the constituent concepts depends on the surrounding background. Let us first illustrate with a rather simple pictorial analog.



Concentrating on the motif labelled A, we notice it portrays what is an analog with the two patterns as main constituents. Each pattern in turn is made up of a few simple shapes, a diamond and an umbrella-like figure. A number of different relations may occur between the left and the right patterns in A and some of these may not be immediately discerned. Compare this independent appraisal of A with the *task* of *choosing* among the analogs $B.1$ and $B.2$ the one that most closely matches A. As one finds $B.1$ to be the most closely matching analog, the relation of **inverting the arrow markings** and, possibly, that of **rotating the inner figure by** $\frac{\pi}{2}$ are elicited. On the other hand, against the background of the task of choosing between $C1$ and $C2$, the relations **rotating the inner figure by** $\frac{\pi}{2}$ and **switching the outer figure with the inner one** are now elicited yielding a preference for $C.1$ as the most nearly matching analog. Thus, different tasks will occasion different relations to become manifest between the patterns.

The situation is that more remarkable for the more complex *verbal* analogs. Thus, take for instance the analog ⟨ **ostrich**, **bird** ⟩. To resolve a choice for the analog that most nearly matches ⟨ **ostrich**, **bird** ⟩ among the following two analogs,

⟨ **panda**, **bear** ⟩
⟨ **car**, **vehicle** ⟩

one may need to discern the relation **being an animal species of**, a rather immediate one. To choose the most matching analog between the two other analogs,

⟨ **panda**, **bear** ⟩
⟨ **cheetah**, **cat** ⟩

one may need to elicit the relation **being the fastest runner of**. To further select one of the following two analogs as the closest one,

⟨ **car**, **vehicle** ⟩
⟨ **skyscraper**, **building** ⟩

one may need to elicit the relation **being the largest of**. To do the same with these other two analogs,

⟨ **panda**, **bear** ⟩
⟨ **giraffe**, **ruminant** ⟩

one may need to discern the relation **having the largest neck-body ratio of**, and so on *ad libitum*. Hence, the presence of different *backgrounds* in the form of *selection tasks* occasions the elicitation of respectively different conceptual relations between the constituent concepts in the verbal analog.[11] This is likewise true for a wide array of different backgrounds and purposes [19,25,24]. Let us formulate the remark thusly:

> *Background-dependence of relations:* For a verbal analog $\langle \mathbf{w_1}, ..., \mathbf{w_n} \rangle$ and two constituent concepts $\mathbf{w_i}$, $\mathbf{w_j}$, what conceptual relations are elicited between $\mathbf{w_i}$ and $\mathbf{w_j}$ depends on the background against which the analog $\langle \mathbf{w_1}, ..., \mathbf{w_n} \rangle$ is considered.

One may wonder about the connection between the elicitation of relations and the previous two remarks. Words are as significant in verbal analogs as they are in

[11] Additional examples of background-dependence can be found in [7].

word meanings; thus, *ambiguity* carries over. As for *variableness of meanings*, one finds the analogy between ⟨ **phone**, **photocamera** ⟩ and ⟨ **hammer**, **weapon** ⟩ supported rather well by a relation such as **being sometimes used as**; this might not have been so just a few decades ago.

The three remarks encapsulate some of the more pervasive phenomena marking linguistic concepts and their behavior. Attempting a representation of such concepts might strike one as alike to the effort of building a house over marshy terrain. As lore tells us, the solution will involve plunging the pillars of representation deep below the watery surface.

2.iv Representing constructs

The theoretical interpretation given so far of the syntactic constructs in terms of certain linguistic concepts will serve as a guiding light to the endeavor that is the primary concern of the present essay, i.e. to provide the constructs with a mathematical *representation*. We here lay out a general *form* for such representations that encompasses more than the particular constructions given later in Sect. 4-5.[12] The general form is specified in terms of a *context function* δ and an *agglomerate function* Σ.

Fix $\mathcal{L}$ to be a *corpus* of language. We view $\mathcal{L}$ as a sequence of *word tokens* $\langle t_i \rangle_{i \leq k}$ that count as instances of particular *words*. Fix $\mathcal{C}$ to be a *collection* of selected *structures*; these are commonly *vectors* or *graphs*. One first defines a *context function* $\delta : \langle t_i \rangle_{i \leq k} \to \mathcal{C}$ that assigns to each word token t_j in $\langle t_i \rangle_{i \leq k}$ the *structure* $\delta(t_j)$; $\delta(t_j)$ is thought of as capturing the *verbal context* of the token t_j in $\mathcal{L}$. An example follows.

Let t_j be the token "aardvarks' " and let the immediate sequence of tokens around t_j in $\langle t_i \rangle_{i \leq k}$ be "the biologists found the aardvarks' tale thrilling". Let the chosen *structures* in $\mathcal{C}$ be *vectors* of m dimensions where $m = |\mathcal{W}|$ and let the dimensions be tagged by words in $\mathcal{W}$.[13] Then, a simple example of a *context function* δ - much simpler than the one we adopt in Sect. 4 - is one that assigns to t_j the vector $\delta(t_j)$ with a value of 0 everywhere, except for a value of 1 in the dimensions tagged with the corresponding words 'biologist', 'find', 'tale',

[12] However, the representation does not purport to be completely general, not even within the domain of all computational models of semantics.

[13] The set of *words* $\mathcal{W}$ and, consequently, its size may depend on $\mathcal{L}$ itself.

‘thrilling’; ‘aardvark’ itself is here given a value of 0.[14] Fig. 1 gives a sample $\delta(t_j)$. Let us now look at a further notion.

biologist	philosopher	find	aardvark	cassowary	tale	vase	thrilling
1	0	1	0	0	1	0	1

Fig. 1. A section of a sample vector $\delta(t_j)$ for the token “aardvarks’ ”.

Further, one defines an *agglomerate function* $\Sigma : \mathcal{C}^{<\omega} \to \mathcal{C}'$ that assigns to a finite sequence of *structures* $\langle\delta(t_i)\rangle_{i\leq k}$ a further *structure* $\Sigma(\langle\delta(t_i)\rangle_{i\leq k})$ in $\mathcal{C}'$, where $\mathcal{C}'$ might possibly contain structures of a different type from that of $\mathcal{C}$. In fact, we assume here that $\Sigma(\langle\delta(t_i)\rangle_{i\leq k})$ is always a *vector* in a *n-dimensional real inner-product space* $\mathcal{H}$; thus, $\mathcal{C}' = \mathcal{H}$. Let us extend the aforementioned example.

Specify a simple diagonal map τ, where $\tau(i,j) = 1$ if the word corresponding to token t_i is the same as the word corresponding to token t_j; and $\tau(i,j) = 0$ otherwise. Then, for the token t_j, an *agglomerate function* may be specified by,

$$\Sigma(\langle\delta(t_i)\rangle_{i\leq k}) = \sum_i^k \tau(i,j) * \delta(t_i) \ . \tag{1}$$

The specified Σ just sums the vectors of the verbal contexts of all tokens t_i that correspond to the same word w as t_j does. Clearly for every token t_l corresponding to the particular word w, Σ will yield the same sum of vectors. Thus, Σ is thought as yielding for each *word* a unique sum of vectors, that is itself a vector in $\mathcal{H}$. Fig. 2 illustrates such a Σ. Let us finally consider the form of representations.

	biologist	philosopher	find	aardvark	cassowary	tale	vase	thrilling
aardvark	6	2	2	0	3	3	0	1

Fig. 2. A section of a sample vector $\Sigma(\langle\delta(t_i)\rangle_{i\leq k})$ for the word ‘aardvark’ obtained from vectors $\delta(t_j)$ for tokens of ‘aardvark’.

[14] Notice that here a procedure as mentioned in Sect. 2.i is assumed that extracts words from word tokens; some of the tokens are disregarded (e.g. “the”).

Let $\mathcal{W}^*$ be the set of syntactic constructs, i.e. n-tuples of words from $\mathcal{W}$ for $n \geq 1$. Given some n, and some *context* and *agglomerate* functions δ^n and Σ^n possibly dependent on n, for some n-tuple $\langle w_1, ..., w_n \rangle$ its *representation* $\langle \boldsymbol{w_1, ..., w_n} \rangle$ is defined by the following general form:

$$\langle \boldsymbol{w_1, ..., w_n} \rangle = \Sigma^n(\langle \delta^n(t_i) \rangle_{i \leq k}) \ . \tag{2}$$

That is, the *representation* $\langle \boldsymbol{w_1, ..., w_n} \rangle$ of the construct will generally be the result of an *agglomerate function* Σ^n applied to *verbal contexts* extracted by δ^n from the *word tokens* t_i that form a corpus of language $\mathcal{L}$. For instance, the functions δ and Σ mentioned in the above example yield a simple representation for the single word construct ⟨ 'aardvark' ⟩ (Fig. 2).

A consequence of Eq. 2 is that the representation is a function of the *verbal contexts* in which the *word tokens* occur. In terms of the interpretation of constructs as *concepts*, the representation of the corresponding concept, be it a word meaning or a verbal analog, is a function of the verbal contexts in which the concept occurs in the form of a corresponding word token. In short, conceptual representations are borne out of verbal contexts.

It is also worth attempting an initial appraisal of the general consequences of the three remarks of Sect. 2.iii for representations of the type of Eq. 2. In regard to the *ambiguity* of words, if one wishes to represent one particular meaning $\mathbf{w^i}$ of a word w, it will not likely do to simply take into consideration (in the procedure Σ) *all* the *verbal contexts* $\delta(t_i)$ where t_i counts as a token of w; one would need to select only those word *tokens* and corresponding verbal *contexts* that involve an occurrence and use of the concept $\mathbf{w^i}$.[15] In the absence of such a procedure, the resulting *representation* $\langle \boldsymbol{w} \rangle$ will likely superimpose, in accordance with the operations specified by Σ, contextual information from *any* of the distinct meanings of w; the result is likely to be an average, somewhat noisy, but nevertheless still rather effective, representation of the most frequently occurring meanings of w. Similar remarks also apply in the case of the *variableness* of meanings, as ambiguity and variableness have similar effects within verbal contexts.

In regard to the *background-dependence* of conceptual relations in verbal analogs, in representing the relations between the concepts that two words stand for, one would similarly need to select only those word tokens and corresponding

[15] This is not straightforward. Search-based techniques such as those adopted in Sect. 6 may alleviate the effects of ambiguity somewhat.

contexts that are *relevant* to the *background* of the verbal analog under consideration. In the absence of such a selection, we are likely to likewise obtain an average and somewhat noisy representation of the most immediate conceptual relations within the verbal analog. Besides representations themselves, we also need an apparatus for *comparing* the representations with each other. To this we turn next.

2.v Degrees of synonymy and of analogical strength

A *degree of similarity* between *vectors* in $\mathcal{H}$ may be computed by a variety of functions. Let us mention three such functions [39]. Consider two vectors $\boldsymbol{p}, \boldsymbol{q} \in \mathcal{H}$. To begin with, we may consider the *Euclidean distance* between $\boldsymbol{p}$ and $\boldsymbol{q}$, which is equivalent to the *L2-norm* $||\boldsymbol{q} - \boldsymbol{p}||_2$ computed by,

$$||\boldsymbol{q} - \boldsymbol{p}||_2 = \sqrt{\sum_i^n (\boldsymbol{q}_i - \boldsymbol{p}_i)^2} \ . \tag{3}$$

To turn the *distance* between $\boldsymbol{p}$ and $\boldsymbol{q}$ into a degree of *similarity*, one may take for instance the inverse value $\frac{1}{||\boldsymbol{q}-\boldsymbol{p}||_2}$.

A second possibility is to use the *L1-norm* instead of the *L2-norm* as a measure of distance, which is simply given by

$$||\boldsymbol{q} - \boldsymbol{p}||_1 = \sum_i^n |\boldsymbol{q}_i - \boldsymbol{p}_i| \ . \tag{4}$$

An inversion will turn this into a similarity measure as well.

Finally, a commonly used measure that directly yields a degree of similarity is the value of the *cosine* of the *angle* θ between $\boldsymbol{p}$ and $\boldsymbol{q}$. If $\cdot$ is the *inner product* operation, then the cosine measure $\sigma(\boldsymbol{p}, \boldsymbol{q})$ is computed by,

$$\sigma(\boldsymbol{p}, \boldsymbol{q}) = \frac{\boldsymbol{p} \cdot \boldsymbol{q}}{||\boldsymbol{p}||_2 \ ||\boldsymbol{q}||_2}. \tag{5}$$

All such functions *intuitively* satisfy that the more $\boldsymbol{p}$ and $\boldsymbol{q}$ have in "common", the higher their degree of similarity. We use the L_1-*norm* and *cosine* measures in Sect. 6.

It is interesting to interpret the *degree of similarity* between two vectors in $\mathcal{H}$ when the latter are in fact *representations* of syntactic constructs. A degree of *similarity* between vectors $\langle \boldsymbol{w_1} \rangle$ and $\langle \boldsymbol{w_2} \rangle$ that *represent* single words and that are *interpreted* as capturing the meanings of such words corresponds to the *degree of synonymy* between the two words w_1 and w_2. In other words, if $\langle \boldsymbol{w_1} \rangle$ and $\langle \boldsymbol{w_2} \rangle$ do capture to a significant extent the meanings of respectively w_1 and w_2, then the more *similar* the vectors, the more *synonymous* the words.

A respective interpretation holds for *pairs* $\langle \boldsymbol{w_1}, \boldsymbol{w_2} \rangle$ and $\langle \boldsymbol{w_3}, \boldsymbol{w_4} \rangle$. If the vectors $\langle \boldsymbol{w_1}, \boldsymbol{w_2} \rangle$ and $\langle \boldsymbol{w_3}, \boldsymbol{w_4} \rangle$ representing the pairs $\langle w_1, w_2 \rangle$ and $\langle w_3, w_4 \rangle$ do capture reliably the conceptual content of the constituents and the conceptual relations between them, then the more *similar* the representing vectors, the more *analogous* the analog of $\langle w_1, w_2 \rangle$ is to that of $\langle w_3, w_4 \rangle$. In other words, a degree of similarity between such representations corresponds to a *degree of analogical strength* between the constructs. An equivalent statement holds for arbitrary n-tuples. Thus, we notice that, on the interpretational side, just as a *verbal analog* encompasses the notion of *word meaning*, so does *analogical strength* encompass the notion of *synonymy*.

* * *

We have specified the syntactic constructs that serve as labels to the representational objects of the framework and we have detailed an interpretation of the constructs that views word meanings as a special case of verbal analogs and, respectively, views synonymy as a special case of analogical strength. On the side of representations, we have presented a somewhat more general Σ-δ-form for vectorial representations corresponding to the constructs and have specified similarity measures for comparing vectorial representations. After an interluding Sect. 3 in which we describe an experimental setting to test our framework, we return in Sect. 4 to detail the computational aspects of the framework yielding a particular δ function; in Sect. 5 we then detail the algorithms giving rise to particular Σ functions and to the desired representations.

3 Experimental Layout and Foregoing Models

The primary aim of the specification and algorithms in Sect. 4-5 and of the experimental evaluations in Sect. 6 concerns *representations* of *pairs* of words $\langle w_1, w_2 \rangle$. We here describe the setting of the experimental evaluation for such representations. Further, we survey first a few, previously proposed, models that have a similar aim, but a somewhat different specification and, secondly, a few models that have a related specification, but a somewhat distinct aim.

3.i A collection of analogy problems

Representations of the form $\langle \boldsymbol{w_1}, \boldsymbol{w_2} \rangle$ are evaluated on their efficacy at solving *multiple-choice verbal analogy problems*. The collection of analogy problems has been compiled from a selection of 374 problems from past College Board SAT entrance examinations; the collection is produced and maintained by P. Turney [36,38]. [16] A verbal analogy problem consists of a *source* (S) verbal analog and five possible *target* (T) verbal analogs; the problem requires one to select the target analog that creates the *strongest* analogy with the source analog. A typical problem thus has the following form:

S : ⟨ 'lull', 'trust' ⟩
T_1 : ⟨ 'balk', 'fortitude' ⟩
T_2 : ⟨ 'betray', 'loyalty' ⟩
T_3 : ⟨ 'cajole', 'compliance' ⟩
T_4 : ⟨ 'hinder', 'destination' ⟩
T_5 : ⟨ 'soothe', 'passion' ⟩

Here the strongest analogy is between the verbal analog of S and that of T_3. Another example is as follows:

[16] The College Board SAT exams do not contain word analogy questions since 2005; but word analogy questions remain an important component of other examinations, such as the Graduate Record Examination GRE.

S : ⟨ 'ostrich', 'bird' ⟩
T_1 : ⟨ 'lion', 'cat' ⟩
T_2 : ⟨ 'goose', 'flock' ⟩
T_3 : ⟨ 'ewe', 'sheep' ⟩
T_4 : ⟨ 'cub', 'bear' ⟩
T_5 : ⟨ 'primate', 'monkey' ⟩

Here the strongest analogy is with the verbal analog T_1. As suggested by the examples, even though most of the words that constitute the word pairs in the collection of analogy problems are indeed *nouns*, words that are *verbs*, *adjectives* and *adverbs* appear rather frequently as well. It is worthwhile mentioning the estimate of human performance on verbal analogy problems. A high-school student about to enter university taking the SAT examination on average obtains an accuracy of about 57% on verbal analogy problems [36]; a baseline given by random guessing yields an accuracy of 20%. Keeping in mind the verbal analogy problems making up the evaluative setting, let us now describe three previously proposed models for such problems.

3.ii Insights from foregoing models of verbal analogy

We survey the early Rumelhart and Abrahamson's psychological *Model for Analogical Reasoning* (*MAR*) [35], Turney's *Latent Relational Analysis* (*LRA*) [36] and the more recent Baroni and Lenci's *Distributional Memory* (*DM*) [2] that is related to *LRA*. A distinction arises between a more *meaning-centered* and a more *relation-centered* view of the *conceptual relations* elicited between the concepts $\mathbf{w_1}$ and $\mathbf{w_2}$ in a verbal analog $\langle \mathbf{w_1}, \mathbf{w_2} \rangle$. We see that *MAR* incorporates the former, whereas *LRA* and *DM* incorporate the latter.

Model for Analogical Reasoning Consider a source word pair $\langle w_1, w_2 \rangle$ and a target word pair $\langle w_3, w_4 \rangle$. Let $\boldsymbol{w_1}, \boldsymbol{w_2}, \boldsymbol{w_3}, \boldsymbol{w_4} \in \mathcal{H}$ be representations of the respective *meanings* of the corresponding words as vectors in a Hilbert space. *MAR* defines an ideal vector

$$\boldsymbol{i} = \boldsymbol{w_3} + (\boldsymbol{w_2} - \boldsymbol{w_1}) \tag{6}$$

such that the *strength* of the verbal analogy is a monotone decreasing function of the Euclidean distance between $\boldsymbol{i}$ and $\boldsymbol{w_4}$ in $\mathcal{H}$. More specifically, let f be such a monotone decreasing function. Then, the analogical strength is given by,

$$f(||\boldsymbol{w_4} - \boldsymbol{i}||_2) = f(||\boldsymbol{w_4} - (\boldsymbol{w_3} + (\boldsymbol{w_2} - \boldsymbol{w_1}))||_2) = f(||(\boldsymbol{w_4} - \boldsymbol{w_3}) - (\boldsymbol{w_2} - \boldsymbol{w_1})||_2)\ . \tag{7}$$

That is, the analogical strength is given by a monotone decreasing function of the Euclidean distance between the vectors $(\boldsymbol{w_2} - \boldsymbol{w_1})$ and $(\boldsymbol{w_4} - \boldsymbol{w_3})$. Analogously, let σ be any similarity measure of those mentioned in Sect. 2.v. Then, extending the insight of MAR to these measures, the analogical strength is given by,

$$\sigma(\boldsymbol{w_2} - \boldsymbol{w_1}, \boldsymbol{w_4} - \boldsymbol{w_3})\ . \tag{8}$$

By taking $(\boldsymbol{w_2} - \boldsymbol{w_1})$ as *de facto* a representation $\langle \boldsymbol{w_1}, \boldsymbol{w_2} \rangle$ of the verbal analog, and likewise for $(\boldsymbol{w_4} - \boldsymbol{w_3})$, after constructing the *meaning* vectors $\boldsymbol{w_1}, \boldsymbol{w_2}, \boldsymbol{w_3}, \boldsymbol{w_4}$ we evaluate the efficacy of these particular representations in Sect. 6.

Notice the significant part that *meaning* plays in representing $\langle \boldsymbol{w_1}, \boldsymbol{w_2} \rangle = (\boldsymbol{w_2} - \boldsymbol{w_1})$. The conceptual relations of the verbal analog $\langle \mathbf{w_1}, \mathbf{w_2} \rangle$ are *implicitly* represented in terms of a simple algebraic function $(-)$ of the *meanings* of the constituents. This meaning-centered view contrasts with the *explicit* representation of the conceptual relations that is harvested in models such as *LRA* and *DM*. Let us examine the latter in turn.

Latent Relational Analysis The *Structure Mapping Theory* of analogy [18] underscores, among others, the centrality of the role that *conceptual relations* play in the formation of an analogy. *LRA* makes this role explicit by searching in a corpus of language for *short phrases* that occur between two words from a word pair. Given a collection of word pairs C and a corpus of language $\mathcal{L}$, *LRA*'s core algorithms involves the following steps:

1. For each word pair $\langle w_1, w_2 \rangle \in C$, form alternate word pairs by combining one of the words w_1, w_2 with a *synonym* of the other word; the synonyms are obtained from a thesaurus.
2. For the original pair $\langle w_1, w_2 \rangle$ and each alternate pair, search in $\mathcal{L}$ for short phrases of less than $k = 5$ words such that the first word in the phrase is one of w_1, w_2 and the second word is the other.
3. Sort the alternate word pairs by the number of short phrases found for each of them; keep the topmost three alternate word pairs with most phrases, in addition to the original word pair; add alternate word pairs to C.

4. For each phrase occurring between the kept word pairs, exclude the first and last word, and from the remaining at most $k-2$ words, build 2^{k-2} *patterns* by replacing every subset of the $k-2$ words with respective *wildcards*. Filter the top $l = 4000$ most occurring patterns between the word pairs.
5. Build a matrix M where rows are indexed with word pairs from C and columns are indexed by patterns. Apply *log* and *entropy transformations* and smooth the matrix with *singular value decomposition* [36].

For each word pair $\langle w_1, w_2 \rangle \in C$, the result is a vector $\boldsymbol{r}$ encapsulating the weighted counts of explicit phrase *patterns* from $\mathcal{L}$; these phrase patterns are seen as explicit instantiations of the various *conceptual relations* between w_1 and w_2. We thus see that the representation $\boldsymbol{r}$ given to the verbal analog $\langle \mathbf{w_1}, \mathbf{w_2} \rangle$ by *LRA* is a *relation-centered* one that is not specified in terms of representations $\boldsymbol{w_1}$, $\boldsymbol{w_2}$ for *meanings*.

It is notable that while representations alike to those in *MAR* have not, to our knowledge, so far been tested on the aforementioned collection of analogy problems, *LRA* has been tested and achieves state-of-the-art performance on the problems. With a corpus of language $\mathcal{L}$ consisting of about $5 * 10^{10}$ word tokens, *LRA* achieved an accuracy of 56%, not significantly different from average human performance [36]. With a corpus $\mathcal{L}$ consisting of about $2.83 * 10^9$ word tokens, an order of magnitude smaller, *LRA* achieved an accuracy of 37.8% [2]. Let us finally consider the *DM* model.

Distributional Memory The *DM* model extracts *triples* $\langle w_1, l, w_2 \rangle$, like $\langle bird, \textsf{as}, ostrich \rangle$, from a dependency parsed corpus $\mathcal{L}$, where l is the *type* of the link connecting w_1 to w_2.[17] Each triple is given a weight t that depends, among others, on the frequency of the triple in $\mathcal{L}$. The triples and their weights give rise to the following two matrices:[18]

- *word* by *link-word* matrix M_1: a word w_1 is given a representation $\boldsymbol{w_1}$ where each value corresponds to the weight of a triple $\langle w_1, l, w_k \rangle$, for some link l and word w_k;

[17] The extraction of such triples is not unique to DM and is commonly adopted in *structured* vector space models [11].

[18] DM actually includes two additional matrices. All four matrices are naturally derived from a labelled third order tensor [2].

– *word-word* by *link* matrix M_2: a pair of words $\langle w_1, w_2 \rangle$ is given a representation $\langle \boldsymbol{w_1}, \boldsymbol{w_2} \rangle$ where each value corresponds to the weight of a triple $\langle w_1, l, w_2 \rangle$, for some link l.

Note that M_1 aims at capturing the *meaning* of w_1 whereas M_2 aims at the *relations* between w_1 and w_2.[19]

There are three kinds of matrices M_1 and M_2 in DM depending on the kind of link types considered. In short, the $DepDM$ model considers semantic types of links obtained from dependency paths, such as sb_intr (subject of an intransitive verb), obj (direct object), and prepositions themselves such as with and as. Examples of triples in $DepDM$ are $\langle \textit{book}, \textsf{obj}, \textit{read} \rangle$ and $\langle \textit{bird}, \textsf{as}, \textit{ostrich} \rangle$. The $LexDM$ model includes the types of links in $DepDM$ and adds many additional types in such a way that almost any verb or adjective, tagged with suffixes encoding additional information, constitutes a type. An example is $\langle \textit{soldier}, \textsf{use}\textit{+n-the+n-a}, \textit{gun} \rangle$. Finally, $TypeDM$ uses the same types of links as $LexDM$, but it drops the suffixes, and instead of counting the *frequency* of a triple in $\mathcal{L}$, it counts the number of different suffixes that a link type has. Thus, if $LexDM$ also included the triple $\langle \textit{soldier}, \textsf{use}\textit{+n-the+n-the-j}, \textit{gun} \rangle$, then $TypeDM$ would include the triple $\langle \textit{soldier}, \textsf{use}, \textit{gun} \rangle$ counting the two former triples as two occurrences of the latter, independently of the frequencies of the two former triples. Each of $DepDM$, $LexDM$ and $TypeDM$ gives rise to a pair of matrices M_1 and M_2.

$DepDM$, $LexDM$ and $TypeDM$ using the corresponding matrices M_2 achieve respectively an accuracy of 29.3%, 31.4% and 42.4% on the 374 analogy questions, given the *same* smaller corpus of $2.83 * 10^9$ tokens [2]. $TypeDM$ achieves the highest accuracy to date on a corpus of that size. As suggested above, these M_2 matrices and respective models incorporate a *relation-centered* view of the relations in verbal analogs.

This concludes the description of DM and of some of the more relevant foregoing models of verbal analogy. The framework described in Sect. 4 extracts links and counts frequencies in a way that is similar to the extraction of links in $DepDM$ and $LexDM$ (though not similar to that in $TypeDM$ or LRA); but the tools used in the construction of the framework and the resulting graph structure are different. Let us thus mention models with a different purpose than ours, but which adopt similar tools and resulting graph structure.

[19] Similarly, M_1 is said to capture the *attributional similarity* of words, whereas M_2 is said to capture the *relational similarity* between pairs of words [37,2].

3.iii Models of a related specification

The framework that we present in Sect. 4-5 consists of a core graph structure, a *word-graph*, and of *algorithms* applied to word-graphs. The construction of a *word-graph* is similar to the construction of the semantic network underlying the *ASKNet* system [23] and of that in [42]. The same tools *C&C* and *Boxer* are adopted in all cases; the construction of a word-graph differs somewhat in the processing of the *Boxer* output (Sect. 4). The algorithms presented in Sect. 5 include, among others, the use of *spreading activation* over word-graphs; spreading activation over semantic networks is also used in [23,22,42].

* * *

We have described the experimental setting under consideration involving 374 verbal analogy problems. We have seen two primary ways of understanding representations of verbal analogs, a *meaning-centered* one and a *relation-centered* one. Keeping these views and respective models in mind, let us proceed to specify the construction of the graph structure underlying the framework.

4 Word-Graphs and their Assemblage

A *word-graph* is a graph with *words* at its *vertices* and *conceptual role connections* as *weighted directed edges* between the vertices. The connections result from the semantic analyses of sentences containing the words. We here set out to describe the construction of a *word-graph* pointing out the free *parameters* on which the construction depends.[20] One of the parameters, the *merging function* μ, plays a crucial role as it determines whether or not certain types of words are to be merged into a single vertex; this affects the extent to which vertices have *paths connecting* one another.

Preliminaries to the construction The first major parameter $\mathcal{P}_1$ that affects precisely the set of vertices $\mathcal{V}$ and the set of edges $\mathcal{E}$ in a word-graph $\mathcal{G} = \langle \mathcal{V}, \mathcal{E} \rangle$ is the corpus of language $\mathcal{L}$ itself from which $\mathcal{G}$ is constructed. That is, the first parameter is the following:

$$\mathcal{P}_1 :: \textit{corpus of language } \mathcal{L}$$

Given some $\mathcal{L}$, we assume not only a procedure for separating $\mathcal{L}$ into distinct *word tokens* t_i, thus viewing $\mathcal{L}$ as a sequence of tokens $\langle t_i \rangle_{i \leq k}$, but also a procedure for separating $\mathcal{L}$ into a sequence of *sentences* $\langle s_i \rangle_{i \leq l}$; each sentence s_i in turn corresponds to a small sequence of word tokens.

The construction of $\mathcal{G}$ goes through three stages. First, each sentence s is converted into a *discourse representation structure* $\mathcal{B}_s$ that encodes a *semantic analysis* of s [26]. Secondly, $\mathcal{B}_s$ is converted into a *context graph* $\mathcal{G}_s$ that yields a graph representation of the semantic analysis. Finally, a merging procedure incorporates $\mathcal{G}_s$ into $\mathcal{G}$. The construction of $\mathcal{G}$ also defines a *context function* δ assigning a verbal context to each word token t_j in s for every sentence s in $\mathcal{L}$. Let us thus fix a sentence s, say,

"Large birds such as cassowary, emu, and ostrich are displayed in separate compounds."

and proceed to describe the three stages of its conversion.

[20] Advance to Fig. 5 for a picture of the word-graph that we are going to construct.

4.i From sentence to logical structure

A *discourse representation structure* (DRS) $\mathcal{B}_s$ of a sentence s is a pair $\mathcal{B}_s = \langle \mathcal{R}_s, \mathcal{C}_s \rangle$, where $\mathcal{R}_s$ is a set of *discourse referents* and $\mathcal{C}_s$ is a set of DRS-*conditions* [26]. *Discourse referents* are thought of as standing for the objects that s or the discourse preceding and surrounding s refer to. DRS-*conditions* encode the information that s or the surrounding discourse convey about such objects. We obtain $\mathcal{B}_s$ by applying the following two processes to s:

- the *C&C* tools robustly and efficiently *tag* and *parse* the sentence s using *categorial combinatory grammar CCG* ;
- the *Boxer* analyzer interprets the *CCG* parse tree and outputs a semantic analysis in the form of a DRS [4].[21]

We here detail the syntax of *Boxer*'s output DRSs; the latter are defined similarly to the standard first-order DRSs [26].

A DRS D is a *pair* $D = \langle R, C \rangle$, where R is a set of *discourse referents* and C is a set of DRS-*conditions*. *Discourse referents* R are simply given by a set of variables $x_1, ...x_n$. DRS-*conditions* C are in turn inductively defined by the following clauses:

a. if $P_\pi(\cdot)$ is a *one-place predicate symbol* and if $x \in R$, then $P_\pi(x) \in C$, where $\pi \in \{\mathsf{n}, \mathsf{v}, \mathsf{a}, \mathsf{d}, \mathsf{f}\}$ indicates a *part-of-speech type*;

b. if $N_\pi(\cdot)$ is a *named-entity symbol* and if $x \in R$, then $N_\pi(x) \in C$, for $\pi \in \{\mathsf{n}, \mathsf{v}, \mathsf{a}, \mathsf{d}\}$;

c. if $R_\pi(\cdot, \cdot)$ is a *two-place relation symbol* and if $x_1, x_2 \in R$, then $R_\pi(x_1, x_2) \in C$, for $\pi \in \{\mathsf{i}, \mathsf{f}\}$;

d. if $x_1, x_2 \in R$, then $(x_1 = x_2) \in C$;

e. if D is a DRS and $x \in R$, then $(x : D) \in C$; $(x : D)$ stands for a *propositional attitude*;

f. if D_1, D_2 are DRSs, then $(\neg D_1)$, $(D_1 \vee D_2)$, $(D_1 \to D_2)$, and possibly others, are in C.

Finally, one defines a *merge* operation $\uplus$ on pairs of DRSs $D_1 = \langle R_1, C_1 \rangle$, $D_2 = \langle R_2, C_2 \rangle$ by,

$$D_1 \uplus D_2 = \langle R_1 \cup R_2, C_1 \cup C_2 \rangle$$

[21] Other possible formats include *first-order logic* formulas and *segmented* DRS. For a full description, see http://svn.ask.it.usyd.edu.au/trac/candc/wiki/DRSs

yielding a further DRS.[22]

A few explanatory comments are in place. The symbols that π in $(a - c)$ ranges over indicate the following *part-of-speech* (*pos*) *types*:

n : *noun*
v : *verb*
a : *adjective*
d : *adverb*
i : *preposition*
f : *fixed*

Fixed predicate symbols include symbols such as thing, $\mathsf{proposition}$, neuter along with multiple others and are designated by the set $\mathsf{P_f}$. Likewise, *fixed* relation symbols include agent, $\mathsf{patient}$, rel, nn among others and are designated by the set $\mathsf{R_f}$. DRS-conditions of the form defined in clause (f) are designated by $c(D_1, D_2)$.

Let us illustrate the outcome of the *C&C* tools and the *Boxer* analyzer applied to the sentence s. *Boxer*'s output for s in easy-to-read *box* format is depicted in Fig. 3. The resulting DRS $\mathcal{B}_s$ involves the discourse referents $x_0, ..., x_7$. Most of the word tokens in s are turned into DRS-conditions where the symbols are given by the corresponding *words*. Thus, $\mathcal{B}_s$ includes non-fixed predicate symbols such as 'ostrich$_{\mathsf{n}}$' and 'display$_{\mathsf{v}}$', and non-fixed relation symbols such as 'as$_{\mathsf{i}}$' and 'in$_{\mathsf{i}}$'. Other word tokens like "such", "and", and "are" are analyzed away and do not appear in $\mathcal{B}_s$. Further, $\mathcal{B}_s$ contains the *fixed* predicate symbol event that is introduced by the main verb 'display' and the *fixed* relation $\mathsf{patient}(x_6, x_0)$ indicating that the referent x_0 is the *direct object* of the event x_6. $\mathcal{B}_s$ also contains *additional* DRSs as its own conditions. Let us then continue to the next stage and see how $\mathcal{B}_s$ is transformed into a context graph $\mathcal{G}_s$.

[22] There are minor technical differences with Boxer's full output syntax, but any additional information is derived from it. The differences are: (i) we consider four part-of-speech types for predicates, and consider the part-of-speech type of named-entities; (ii) we do not consider presently time-expression and cardinality conditions; (iii) any additional (complex) conditions are included in clause f, but are not treated specially; (iv) DRSs with *alpha-types* are always resolved and thus do not occur in the output.

```
 _________________________
| x0 x1 x2 x3 x4 x5 x6 x7 |
|_________________________|
| large(x0)               |
| bird(x0)                |
| cassowary(x1)           |
| emu(x2)                 |
| ostrich(x4)             |
| display(x6)             |
| separate(x7)            |
| compound(x7)            |
| event(x6)               |
| as(x0,x1)               |
| patient(x6,x0)          |
| in(x6,x7)               |
|        _________        |
|       |         |       |
|    x3:|_________|       |
|       | x1 = x2 |       |
|       |_________|       |
|        _________        |
|       |         |       |
|    x5:|_________|       |
|       | x1 = x4 |       |
|       |_________|       |
|_________________________|
```

Fig. 3. *Boxer*'s analysis $\mathcal{B}_s$ of sentence s.

4.ii From logical structure to context graph

The *context graph* $\mathcal{G}_s = \langle \mathcal{V}_s, \mathcal{E}_s \rangle$ for the sentence s incorporates the linguistic information derived from the *verbal contexts* $\delta(t_i)$ of the word *tokens* t_i in s in terms of *semantic* or *conceptual role connections* between the corresponding *words*; the connections are dictated by the *semantic analysis* $\mathcal{B}_s$. The algorithm for the construction of $\mathcal{G}_s$ thus concludes the *defining procedure* of the *context function* δ; the latter procedure is made explicit below.

Let us begin by explicating the algorithm for the constuction of $\mathcal{G}_s$ from the DRS $\mathcal{B}(s) = \langle \mathcal{R}_s, \mathcal{C}_s \rangle$ obtained for the sentence s. The algorithm has 5 steps.

1. **Processing DRS-conditions** We process and separate the DRS-*conditions* in $\mathcal{B}(s)$ into a set $\mathbf{1}_s$ of *unary conditions* and a set $\mathbf{2}_s$ of *binary conditions*;

the recursive procedure $Q(D)$ for doing so is specified over an arbitrary DRS $D = \langle R, C\rangle$ by the following clauses:

a. for a *predicate* $P_\pi(x) \in C$, $\langle P, \pi, x\rangle \in \mathbf{1}_s$;
b. for a *named-entity* $N_\pi(x) \in C$, $\langle N, \pi, x\rangle \in \mathbf{1}_s$;
c. for a *relation* $R_\pi(x_1, x_2) \in C$, $\langle R, \pi, x_1, x_2\rangle \in \mathbf{2}_s$;
d. for an *equality* $(x_1 = x_2) \in C$, $\langle \mathsf{rel}, \mathsf{f}, x_1, x_2\rangle \in \mathbf{2}_s$ and $\langle \mathsf{rel}, \mathsf{f}, x_2, x_1\rangle \in \mathbf{2}_s$;
e. for a *propositional attitude* $(x : D_1) \in C$, $\langle \mathsf{rel}, \mathsf{f}, x, y\rangle \in \mathbf{2}_s$, where y is the discourse referent of $P_\pi(y) := main(D_1)$;
f. for a DRS-*condition* $c(D_1, D_2)$, apply $Q(D_1)$ and $Q(D_2)$;
g. for a *merged* DRS $D_1 \uplus D_2$, apply $Q(D_1)$ and $Q(D_2)$.

We must define the selection function *main* from clause (e). For a non-empty set of *predicates* or *named-entities* A, order the elements of A by their *pos-types* according to ranking $\mathsf{v} < \mathsf{n} < \mathsf{a} < \mathsf{d} < \mathsf{f}$; then let *priority*$(A)$ be the first element of the resulting ordering.[23] Then, for a DRS $D' = \langle R', C'\rangle$, we simply have $main(D') = priority(\{\ P_\pi(x) \mid P_\pi(x)$ a *predicate* or *named-entity* in $C'\ \})$. Thus, clause (e) in Q heuristically chooses one main predicate or named-entity from the DRS D' giving precedence first to *verbs*, then to *nouns*, and so on through the other *pos-types*; then it relates the discourse referent x of the propositional attitude to the discourse referent of the chosen predicate. In sum, the first step of the algorithm involves separating all the DRS-*conditions* into *unary conditions* that refer to just *one* discourse referent and *binary conditions* that refer to *two* discourse referents; by way of the fixed relation symbol rel, one treats *equalities* as a pair of binary conditions and *propositional attitudes* as one binary condition. The operations in DRS-*conditions* with form $c(D_1, D_2)$ and in merged DRSs are not directly processed, only the DRSs D_1, D_2 themselves are.

2. **Mapping referents to priority unary conditions** The next step involves computing a map u that maps each referent $x \in \mathcal{R}_s$ to a *unary condition* in $\mathbf{1}_s$ that is given by,

$$u(x) := priority'(\{\langle U, \pi, y\rangle \in \mathbf{1}_s \mid y = x\}).$$

[23] We do not impose special conditions on the ordering of predicates with same *pos-type* π.

Here, *priority*$'(A)$ is similar to *priority*, except that it now orders a set A of *unary conditions* and it orders them according to the slightly different pos-type ranking $\mathsf{n} < \mathsf{v} < \mathsf{a} < \mathsf{d} < \mathsf{f}$, that gives precedence first to *nouns*. If the set $\{\langle U, \pi, y\rangle \in \mathbf{1}_s \mid y = x\}$ is empty, we let $u(x) := \langle \mathsf{thing}, \mathsf{f}, x\rangle$ for the *fixed* predicate symbol thing, here used as a temporary place-holder.

3. **Incorporating unary conditions into the context graph** The first building blocks of $\mathcal{G}_s$ are given by *priority unary conditions*. That is, for each referent $x \in \mathcal{R}_s$, we consider first the priority unary condition $u(x) = \langle U, \pi, x\rangle$ and create a vertex $v = \langle U, \pi\rangle$ keeping only the *symbol* U and its *pos-type* π; v is added to the vertices $\mathcal{V}_s$. Further, for every *other* unary condition in $\{\langle U', \pi', y\rangle \in \mathbf{1}_s \mid y = x\}$, we add to $\mathcal{V}_s$ the corresponding vertex $v' = \langle U', \pi'\rangle$ and connect v to v' by adding a directed edge $\langle v, v', \mathbf{e}\rangle$ to $\mathcal{E}_s$ with the default real-valued *weight* $\mathbf{e}$ (we let $\mathbf{e} = 1$). For a referent $x \in \mathcal{R}_s$, we designate by $v(x)$ the vertex (here v) corresponding to $u(x)$.

4. **Incorporating binary conditions** For each *binary condition* $\langle B, \pi, x_1, x_2\rangle \in \mathbf{2}_s$ with $\pi \neq \mathsf{f}$, we form the vertex $v' = \langle B, \pi\rangle$ and consider the vertices $v(x_1)$ and $v(x_2)$; then to $\mathcal{E}_s$ we add $\langle v(x_1), v', \mathbf{e}\rangle$ and $\langle v', v(x_2), \mathbf{e}\rangle$. The remaining fixed binary conditions $\langle B, \pi, x_1, x_2\rangle \in \mathbf{2}_s$ where $\pi = \mathsf{f}$ are treated as follows:
   - for $\langle \mathsf{agent}, \mathsf{f}, x_1, x_2\rangle \in \mathbf{2}_s$, one adds $\langle v(x_2), v(x_1), \mathbf{e}\rangle$ to $\mathcal{E}_s$;
   - for $\langle \mathsf{nn}, \mathsf{f}, x_1, x_2\rangle \in \mathbf{2}_s$, one adds $\langle v(x_1), v(x_2), \mathbf{e}\rangle$ and $\langle v(x_2), v(x_1), \mathbf{e})\rangle$ to $\mathcal{E}_s$;
   - for any other fixed binary condition $\langle B, \mathsf{f}, x_1, x_2\rangle \in \mathbf{2}_s$, including $\langle \mathsf{rel}, \mathsf{f}, x_1, x_2\rangle$, one adds $\langle v(x_1), v(x_2), \mathbf{e})\rangle$ to $\mathcal{E}_s$.

   Note that, if on the one hand, non-fixed binary conditions correspond to vertices bridging between the vertices of the referents, on the other, one does not add any vertices for *fixed* binary conditions. One interprets them instead by adding possibly new edges between existing vertices.

5. **Circumventing fixed unary conditions** At this stage, vertices $\langle V, \pi\rangle \in \mathcal{V}_s$ that have $\pi = \mathsf{f}$ are those corresponding to fixed *unary* conditions. Let $f = \langle V, \mathsf{f}\rangle$ be any such vertex. For any other distinct vertices v, v' such that $\langle v, f, \mathbf{e}\rangle$ and $\langle f, v', \mathbf{e}\rangle$ are edges in $\mathcal{E}_s$, we incorporate into $\mathcal{E}_s$ the transitive closure $\langle v, v', \mathbf{e}\rangle$ of the two edges. After the transitive edges have been added

for every such pair of vertices v, v' and for every vertex $f = \langle V, \mathsf{f} \rangle$, we finally drop from $\mathcal{G}_s$ the vertices with form $f = \langle V, \mathsf{f} \rangle$ and all the edges that begin or end at such vertices. This ensures that the linguistic information encoded by fixed unary conditions is retained in the form of possibly novel edges between the relevant vertices, while at the same time the resulting vertices in $\mathcal{G}_s$ have symbols that are *words* extracted from tokens in s. This concludes the generation of $\mathcal{G}_s$.

If we use the algorithm to generate the context graph $\mathcal{G}_s$ for our sentence s from the DRS $\mathcal{B}_s$, we obtain the graph depicted in Fig. 3. Notice for instance how the equalities $(x_1 = x_2)$ and $(x_1 = x_4)$ in $\mathcal{B}_s$ have been resolved to bi-directed arrows from 'cassowary' to 'ostrich' and to 'emu'. Notice also how the fixed binary condition with symbol `patient` has given rise to an arrow between 'display' and its direct object 'bird'. With $\mathcal{G}_s$ having been constructed, let us look at the resulting *context function* δ.

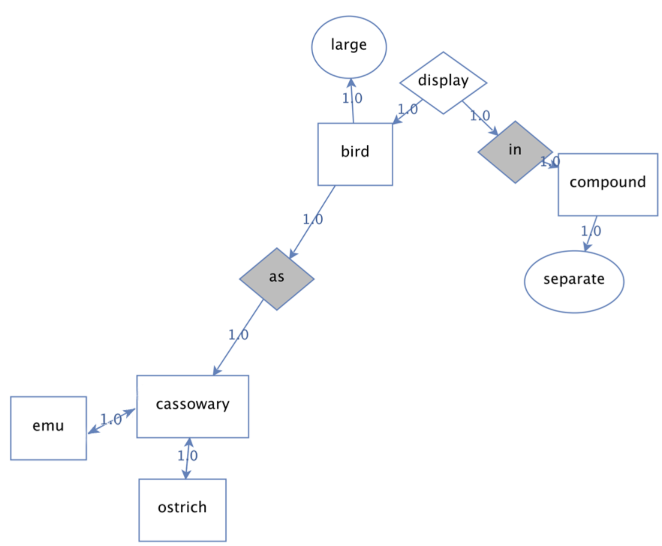


Fig. 4. Context graph $\mathcal{G}_s$ for sentence s. Vertices of pos-type n are rectangles, those of pos-type v are rhombuses, those of pos-type a are ellipses, and those of pos-type i are shaded rhombuses. Vertices of pos-type d do not occur in $\mathcal{G}_s$.

The context function is defined on *word tokens* t_i. For a word token t_i, we designate by $v^*(t_i)$ the vertex $\langle w, \pi \rangle$ in $\mathcal{G}_s$ where the symbol w is the *word* extracted from t_i during the construction and π is the extracted pos-type. We then define $\delta(t_i)$ as a vector whose values are given by:[24]

$$\delta(t_i)_j := \begin{cases} e' & \text{if } \langle v^*(t_i), v^*(t_j), e' \rangle \in \mathcal{E}_s \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

The present δ function might be undefined for a word token t_i if $v^*(t_i)$ is undefined in turn, i.e. if the token t_i does not ultimately correspond to a *word* and a vertex in $\mathcal{G}_s$. The central aspect of $\delta(t_i)$ is that the resulting vector encoding t_i's *verbal context* is non-zero only on those dimensions j for which there exists an edge from $v^*(t_i)$ into $v^*(t_j)$. Thus, δ("birds") is non-zero on dimensions corresponding to the vertices $\langle$'as$'$, $\mathsf{i}\rangle$ and $\langle$'large$'$, $\mathsf{a}\rangle$ that have an outgoing edge from $\langle$'bird$'$, $\mathsf{n}\rangle$ connected to them; in other words, the verbal context of "birds" and of the resulting *word* 'bird' is captured in terms of conceptual role connections such as *being the argument of the adjective or attribute* 'large', and *being the first argument of the preposition or relation* 'as'. Similarly, δ("are_displayed") is non-zero on dimensions corresponding to $\langle$'in', $\mathsf{i}\rangle$ and $\langle$'bird', $\mathsf{n}\rangle$, its context being captured by conceptual role connections to these *words*. Thus, the present context function δ is very different from the 'bag-of-surrounding-tokens' context function illustrated in Sect. 2.iv; the present δ defines the *verbal context* of a token in terms of conceptual role connections obtained from the syntactic and semantic processing of the sentence s in which the token occurs. Let us lastly examine how $\mathcal{G}_s$ is merged into $\mathcal{G}$.

4.iii Merging context graphs into a word-graph

The operation of merging a context graph $\mathcal{G}_s$ into a word-graph $\mathcal{G}$ is straightforward. For every edge $\langle v, v', e \rangle \in \mathcal{E}_s$, where e is the weight of the edge, one selects a vertex $x_v \in \mathcal{V}$, another vertex $x_{v'} \in \mathcal{V}$ and, if no edge from x_v to $x_{v'}$ exists, one adds an edge $\langle x_v, x_{v'}, e \rangle$ to $\mathcal{E}$; otherwise, if such an edge $g \in \mathcal{E}$ exists, one just augments the weight of g by the value e. What remains to be explained is how x_v and $x_{v'}$ are actually selected.

[24] This definition is the not only possible one; one may define the context function also in terms of ingoing edges at the expense of double-counting edges; ingoing edges play a significant role below.

We parametrize the selection of vertices x_v and $x_{v'}$ that are made to correspond to v, v' on a *merging function* μ. μ specifies whether identical vertices of a given *pos-type* $\pi \in \{\mathsf{n}, \mathsf{v}, \mathsf{a}, \mathsf{d}, \mathsf{i}\}$ should be all merged together in $\mathcal{WG}$ or whether they should all be kept distinct. Thus, if v is a vertex with $\pi = \mathsf{n}$, and vertices of *pos-type* n are to be merged according to μ, then one simply finds the vertex $x_v = v$ in $\mathcal{G}_s$, if such a vertex x_v already exists; if v is not to be merged or if it does not exist in $\mathcal{G}_s$, a new vertex x_v is created in $\mathcal{G}_s$. We frame the function parameter as follows:

$$\mathcal{P}_2 :: \textit{merging function } \mu$$

Finally, an additional parameter indicates the *maximum* number of vertices in $\mathcal{G}$:

$$\mathcal{P}_3 :: \textit{maximum size of } \mathcal{G}$$

The parameter may be set to *unlimited*, in which case the construction of $\mathcal{G}$ proceeds unaffected as described. If, on the other hand, $\mathcal{G}$ reaches its maximum size, the construction procedure is altered as follows. For every edge $\langle v, v', e\rangle \in \mathcal{E}_s$, suppose that at least one of v and v' is to be merged according to μ; if not, do not consider the edge. Let v be the merged vertex. Then, if there already is an edge f in $\mathcal{G}$ connecting v to a vertex v'' in $\mathcal{G}$ that has the same *word* and *pos-type* as v', add the edge weight e to the weight of f; if there are multiple such v', choose one randomly. In all other cases, do not consider the edge $\langle v, v', e\rangle \in \mathcal{E}_s$. This strategy ensures that the number of vertices in $\mathcal{G}$ does not grow, while additional edge weights coming from unprocessed context graphs may be incorporated into $\mathcal{G}$ as long as the edges are already contained in $\mathcal{G}$.

An example of a word-graph built from seven sentences containing the tokens "ostrich" and "bird" with a merging function μ that merges all vertices except those of pos-type i is given in Fig. 5. This concludes the description of the assemblage of word-graph $\mathcal{G}$.

* * *

We have seen how each sentence from a corpus is converted first into a DRS, then into a context graph and then merged into a word-graph. We have seen how this defines a corresponding context function δ. With the word-graph having been constructed, let us now turn to examine algorithms.

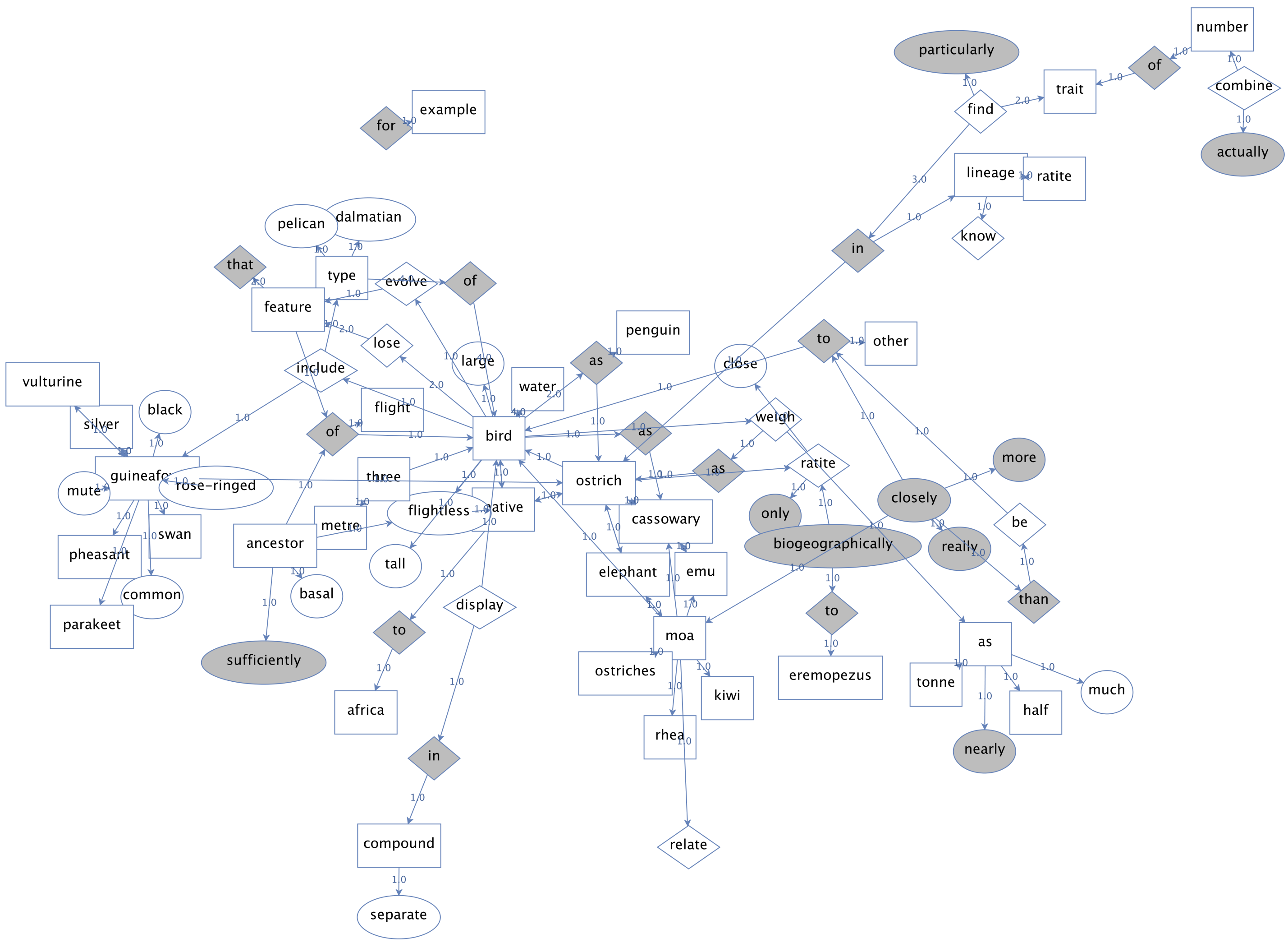


Fig. 5. A word-graph built from seven sentences. The pos-type of vertices are indicated as in Fig. 4. In addition, shaded ellipses indicate vertices with pos-type d.

5 Regions in Word-Graphs and their Algorithms

Having constructed a word-graph, we now aim to extract the linguistic information captured by certain *regions* of the word-graph. A region is simply a *subgraph* of the word-graph. We consider *meaning regions* that are viewed as capturing the information pertaining to the *meaning* of a word. We further consider *relation regions* that are viewed as capturing the information underlying the *conceptual relations* between a pair of words w_1 and w_2. In Sect. 5.i, we see that these regions are given by considering not just *single* links or connections as in models such as DM, but by also considering both sequences of *multiple* links or connections and the *graph structure* itself underlying the connections.

The rest of Sect. 5 is devoted to the algorithms for extracting the information from such regions in the word-graph. We present two family of algorithms $\mathcal{R}$ and $\mathcal{S}$, as well as a hybrid family $\mathcal{T}$. Each of $\mathcal{R}$, $\mathcal{S}$ and $\mathcal{T}$ can be used to extract both meaning and relation regions from a word-graph. The $\mathcal{R}$ family is based on *path distances* and *random walks*. The $\mathcal{S}$ family is based on *spreading activation* and *algebraic operations* on vectors. The hybrid $\mathcal{T}$ family uses *path distances*, but adopts *spreading activation* instead of *random walks*.[25]

5.i Meaning and relation regions

The *meaning region* of a word w_1 is viewed as the subgraph centered around w_1.[26] In Sect. 3.ii, we have seen how in a model such as *DM*, the *word* by *link-word* matrix M_1 yields a representation $\boldsymbol{w_1}$ of the meaning of w_1 by way of the *single* links that link w_1 to other words in the corpus. But, in a word-graph such as that of Fig. 5, one can find sequences of *multiple* connections such as the following:

$$ostrich \rightarrow bird \rightarrow lose \rightarrow feature \rightarrow of \rightarrow flight$$

[25] Thus, note that each of $\mathcal{R}$, $\mathcal{S}$ and $\mathcal{T}$ yields in turn a different instance of an agglomerate function Σ (Sect. 2.iv).

[26] We henceforth identify, when no confusion arises, the word w_1 with the corresponding vertex $\langle w_1, \pi \rangle$ in the word-graph.

It seems here that the entire sequence is informative as to the meaning of 'ostrich', even though, say, 'flight' is multiple connections removed from 'ostrich'. Further, one may find sequences such as,

$$cassowary \to ostrich \to bird \to flightless$$

even though 'cassowary' and 'flightless' do not in fact occur together in any one of the sentences that make up the word-graph in Fig. 5. But 'flightless' is informative as to the meaning of 'cassowary'.[27] More generally, we suppose that the subgraph closely *surrounding* a word w_1, with its sequences of multiple connections and the graph structure itself, may be more informative as to the meaning of w_1 than its single links with other words. We call such a small subgraph centered around w_1 a *meaning region* of w_1, as it captures information pertaining to the meaning of w_1.

The *relation region* of a pair of words w_1 and w_2 is viewed as the subgraph given by the sequences of connections between w_1 and w_2. The above sequences of connections may also be seen as explicit instances of *relations* connecting the first and the last word in the sequence. This is clear in the first case; the captured relation there between 'ostrich' and 'flight' is just,

being a bird that has lost the feature of

In the second case, the suggested relation between 'cassowary' and 'flightless' is somewhat less immediate to paraphrasis, but may naturally be put as,

being related to ostrich that is a bird that is

We thus suppose that the subgraph given by the most informative sequences of multiple connections between w_1 and w_2, and the underlying graph structure, is more informative as to the *relations* that hold between w_1 and w_2 than just single links.[28] We call such a subgraph the *relation region* determined by w_1 and w_2. Fig. 6 gives a schematic depiction of meaning and relation regions.

As we shall see, such regions allow us to obtain different kinds of concrete representations for a verbal analog $\langle \mathbf{w_1}, \mathbf{w_2} \rangle$. Thus, we may identify a *relation-centered* representation $\langle \boldsymbol{w_1}, \boldsymbol{w_2} \rangle^{\mathsf{r}}$ with the *relation* region determined by w_1

[27] Cassowaries are flightless avians.

[28] Single links of this sort are captured in the *word-word* by *link* matrix M_2 of *DM* (Sect. 3.ii).

and w_2. We may also identify a *meaning-centered* representation $\langle \boldsymbol{w_1}, \boldsymbol{w_2} \rangle^{\mathsf{m}}$ with that obtained by way of an algebraic operation, such as subtraction $-$, applied to the *meaning regions* of w_1 and w_2; this representation would be close to that in *MAR*. Let us thus proceed to describe algorithms that extract regions and informativeness values for vertices in such regions from a word-graph.

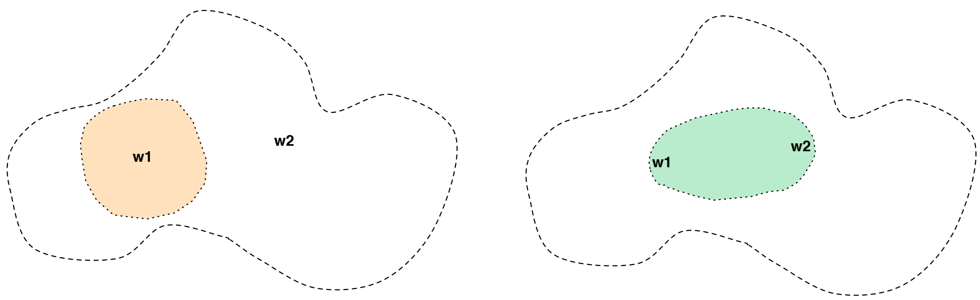


Fig. 6. Schematic depiction of the *meaning region* of w_1 in a word-graph (left) and of the *relation region* between w_1 and w_2 (right).

5.ii Preliminaries to the algorithms

We begin by describing, first, how one selects vertices in $\mathcal{G}$ from which the algorithms are initiated and, secondly, how one transforms the *weights* on the edges in $\mathcal{G}$; these two initial steps apply to all three algorithms $\mathcal{R}$, $\mathcal{S}$ and $\mathcal{T}$.

Initiating vertices Given a word-graph $\mathcal{G}$ constructed as in Sect. 4 and a particular *pair* $\langle w_1, w_2 \rangle$ of words, we need a way of selecting a *pair* of vertices in $\mathcal{G}$ that are the vertices from which the algorithms are initiated. The selection α takes a word-graph $\mathcal{G}$ and a pair $\langle w_1, w_2 \rangle$ and returns a, possibly altered, word-graph $\mathcal{G}^*$ and a pair of initiating vertices $\langle v_1, v_2 \rangle$ in $\mathcal{G}^*$:

$$\mathcal{P}_4 :: \textit{initiating vertices selection } \alpha$$

For example, given the $\mathcal{G}$ from Fig. 3 and the pair $\langle$ 'ostrich', 'bird' $\rangle$, a straightforward function α just returns $\mathcal{G}^* = \mathcal{G}$ and the two vertices $\langle$'ostrich', $\mathsf{n}\rangle$ and

$\langle$‘bird’, $\mathsf{n}\rangle$, if the pos-type n of ‘ostrich’ and ‘bird’ are known or can be determined.[29] This together with α itself concludes the first preliminary step of the algorithms.

PPMI weighting At this point the weight of an edge $\langle w_1, w_2, e\rangle$ in $\mathcal{G}^*$ is just the raw frequency count of the number of times the corresponding *connection of* w_1 *to* w_2 occurs in the sentences from the corpus $\mathcal{L}$ from which $\mathcal{G}^*$ is constructed. But such raw counts may be biased in various ways. For instance, in edges $\langle$‘ostrich’, ‘have’, $e\rangle$ and $\langle$‘ostrich’, ‘flightless’, $e'\rangle$, the raw count e might be higher than the count e' simply because the prior probability of ‘have’ occurring in $\mathcal{L}$ is higher than that of ‘flightless’. One must thus transform the weights so as to incorporate the probability of the particular connection of w_1 to w_2, as well as the prior probability of w_1 itself and that of a connection of *any* vertex to w_2. There are various weighting approaches that have proven effective.[30] Here we adopt *Positive Pointwise Mutual Information* (PPMI) [39].

PPMI is defined as follows. For a word-graph $\mathcal{G} = \langle \mathcal{V}, \mathcal{E}\rangle$, let $n = |\mathcal{V}|$ be the number of vertices and let the vertices (and corresponding words) be indexed with integers $i \leq n$. Let $e_{i,j}$ be the weight e in edge $\langle w_i, w_j, e\rangle$. The *PPMI weight* $f_{i,j}$ is given by:

$$\begin{aligned}
p_{i,j} &= \frac{e_{i,j}}{\sum_k^n \sum_l^n e_{k,l}} \\
p_{i,*} &= \frac{\sum_l^n e_{i,l}}{\sum_k^n \sum_l^n e_{k,l}} \\
p_{*,j} &= \frac{\sum_k^n e_{k,j}}{\sum_k^n \sum_l^n e_{k,l}} \\
f'_{i,j} &= log\left(\frac{p_{i,j}}{p_{i,*}p_{*,j}}\right) \\
f_{i,j} &= max(f'_{i,j}\ , 0)
\end{aligned} \tag{10}$$

A *PPMI weighting* of $\mathcal{G}$ substitutes each edge weight $e_{i,j}$ with the PPMI weight $f_{i,j}$. Here we have that $p_{i,j}$ is the estimated probability of the connection of w_i to w_j, $p_{i,*}$ is the estimated probability of w_i, and $p_{*,j}$ is the estimated probability of a connection of any vertex to w_j. Under the assumptions that informative

[29] In Sect. 6, we use a slightly more encompassing function α that sidesteps the need for pos-types.

[30] For a survey, see [39].

connections (such as that of ‘ostrich’ to ‘flightless’) have $p_{i,j} > p_{i,*} * p_{*,j}$, i.e. the probability of a connection between them is greater than if the two happened to co-occur and be connected by random chance, we expect $f_{i,j} > 0$ for informative connections. Under the assumption that uninformative connections (such as that of ‘ostrich’ to ‘have’) are statistically *independent* and thus, by definition, have $p_{i,j} = p_{i,*} * p_{*,j}$, we expect $f_{i,j} = log(1) = 0$ for uninformative connections.[31] If $p_{i,j} < p_{i,*} * p_{*,j}$, we simply take it to be statistically independent and set $f_{i,j} = 0$. Thus PPMI is designed to increase the weights of semantically informative connections and decrease the weights of the uninformative ones. A second step is thus to apply the PPMI transformation to the word-graph. Let us then proceed to the first family of algorithms $\mathcal{R}$.

5.iii Path distance measures

Having applied PPMI weighting to $\mathcal{G}^*$ and with w_1, w_2 being the two initiating vertices, we now specify how to extract *regions* that are subgraphs from $\mathcal{G}^*$ based on the distance between vertices measured by the length of certain *paths*; this constitutes the first part of $\mathcal{R}$. We focus first on the extraction of the *relation region* determined by w_1 and w_2. One important parameter is the (maximum) size κ of the subgraph to be extracted:

$$\mathcal{P}_5 :: \textit{size } \kappa \textit{ of subgraph } \mathcal{G}_{\langle w_1, w_2 \rangle}$$

The aim is to extract the κ vertices that have the most *informative* sequences of connections with w_1 and w_2, where the *informativeness* of a connection is estimated by its weight. If we let the *length* $l_{i,j}$ of a connection from w_i to w_j be inversely proportional to its weight $e_{i,j}$, i.e. $l_{i,j} = \frac{1}{e_{i,j}}$, then a connection is more informative, the shorter it is, and a sequence of connections is more informative, the shorter the sum of the lengths; finally, a vertex w_i is more informative with respect to w_j, the more informative the shortest sequence of connections from w_j to w_i. This idea underlies the three *path distance measures* presented below. The resulting $\mathcal{G}_{\langle w_1, w_2 \rangle}$ is thus an explicit representation in the form of a *word-subgraph* of the most informative sequences of connections, and of the words making up such sequences, relating w_1 and w_2 and of any additional connections between the words themselves. Let us thus specify three path distance measures that turn out to have somewhat different outcomes as to the resulting subgraph $\mathcal{G}_{\langle w_1, w_2 \rangle}$

[31] The two assumptions follow from the so-called *distributional hypothesis* [39] .

Vertices on shortest paths For a connection or edge $\langle w_i, w_j, e \rangle$ and its weight $e_{i,j}$, let its *length* be $l_{i,j} = \frac{1}{e_{i,j}}$; thus, an edge that has high weight and is informative is taken to have a short length. Having defined lengths for edges, for two vertices w_i, w_j, let $\lambda(w_i, w_j)$ be the *length of the shortest directed path* from w_i to w_j in $\mathcal{G}^*$, and *undefined* if no directed path exists. Given initiating vertices w_1, w_2, for a vertex w_i we may now define the first measure Λ_1 as follows:

$$\Lambda_1(w_i) = min(\lambda(w_1, w_i) + \lambda(w_i, w_2),\ \lambda(w_2, w_i) + \lambda(w_i, w_1)) \quad (11)$$

If any of the λ values is undefined, $\Lambda_1(w_i)$ is undefined as well. It is easy to see that, if $\Lambda_1(w_i)$ is defined for w_i, then $\Lambda_1(w_i)$ is the minimum of the length of the shortest directed path from w_1 to w_2 passing through w_i and of the length of the shortest directed path from w_2 to w_1 passing through w_i. Thus keeping the κ vertices with highest Λ_1 value results in a subgraph $\mathcal{G}_{\langle w_1, w_2 \rangle}$ that includes, among others, the h shortest directed *paths* between w_1 and w_2, for some $h \leq \kappa$ (Fig. 7).[32] The complexity of determining the value Λ_1 for every vertex is $O(|\mathcal{V}| * \mathbf{d})$, where $|\mathcal{V}|$ is the number of vertices in $\mathcal{G}^*$ and $\mathbf{d}$ is the running time of the shortest path algorithm.[33]

Vertices on shortest two-way paths An alternative measure Λ_2 is given as follows:

$$\Lambda_2(w_i) = min(\lambda(w_1, w_i),\ \lambda(w_i, w_1)) + min(\lambda(w_2, w_i),\ \lambda(w_i, w_2)) \quad (12)$$

As above, $\Lambda_2(w_i)$ is undefined if so is any of the λ values. If $\Lambda_2(w_i)$ is defined for w_i, then there is a sequence of vertices in $\mathcal{G}^*$ that starts with w_1 and ends at w_2 and that includes w_i. There is no guarantee that this sequence is a *directed path* from w_1 to w_2 or vice versa; guaranteed are only a directed path from w_1 to w_i or from w_i to w_1, and a directed path from w_2 to w_i or from w_i to w_2. Such a sequence we call a *two-way path* from w_1 to w_2 through w_i.[34] Hence

[32] Technically, the path on which the κth vertex resides may not be complete, unless all the remaining vertices on the path are included.

[33] The used implementation of Dijkstra's shortest path algorithm has running time $O(q|\mathcal{V}|log|\mathcal{V}|)$, where q is the average out-degree of a vertex. In word-graphs, q tends to be small at about 2-3, depending on the *merging function* μ.

[34] A two-way path from w_1 to w_2 through w_i is also a two-way path from w_2 to w_1 through w_i.

keeping the κ vertices with highest Λ_2 value results in a subgraph $\mathcal{G}_{\langle w_1,w_2\rangle}$ with $h \leq \kappa$ *shortest two-way paths* from w_1 to w_2, among possibly other two-way paths (Fig. 7). As above, the complexity of determining the value Λ_2 for every vertex is $O(|\mathcal{V}| * \mathbf{d})$.

Vertices on shortest undirected paths Finally, a third, more efficient, measure Λ_3 acts on the *undirected* variant $\mathcal{G}'$ of $\mathcal{G}^*$; the length of an undirected edge between w_i and w_j in $\mathcal{G}'$ is $l'_{i,j} = l'_{j,i} = min(l_{i,j},\ l_{j,i})$, the smallest of the lengths of the corresponding directed edges in $\mathcal{G}^*$, if both edges exist; if only one edge between w_i and w_j exists in $\mathcal{G}^*$, then $l'_{i,j} = l'_{j,i}$ is simply the length of that edge. With $\lambda'(w_i, w_j) = \lambda'(w_j, w_i)$ being the shortest *undirected* path on $\mathcal{G}'$ between w_i and w_j, the measure Λ_3 is given as follows:

$$\Lambda_3(w_i) = \lambda'(w_1, w_i) + \lambda'(w_2, w_i) \tag{13}$$

$\Lambda_3(w_i)$ is undefined if so is any of the two λ' values. If defined, $\Lambda_3(w_i)$ is the length of the *shortest undirected path* from w_1 to w_2 passing through w_i. The corresponding sequence of vertices in $\mathcal{G}^*$ that reside on the undirected path is neither guaranteed to be a directed path nor a two-way path. Thus, keeping in $\mathcal{G}^*$ the κ vertices with highest Λ_3 value results in a $\mathcal{G}_{\langle w_1,w_2\rangle}$ with $h \leq \kappa$ sequences of vertices that viewed in $\mathcal{G}'$ are the shortest undirected paths between w_1 and w_2 (Fig. 7). An important advantage of Λ_3 over Λ_1 and Λ_2 is its complexity. It turns out that to find the Λ_3 value for every vertex in $\mathcal{G}^*$ requires only $O(\mathbf{d})$ steps. In fact, merely two full runs of Dijkstra's shortest path algorithm, one starting from w_1 and the other from w_2 on $\mathcal{G}'$ are sufficient to determine the shortest undirected paths from w_1 and w_2 to every other vertex in $\mathcal{G}^*$.[35] According to our experimental considerations, the increase in efficiency makes Λ_3 a fast and feasible measure on word-graphs that have upwards 500,000 vertices; by contrast, Λ_1 and Λ_2 cease to be really feasible already on word-graphs of more than 20,000 vertices.

The first part of $\mathcal{R}$ is thus to extract a subgraph $\mathcal{G}_{\langle w_1,w_2\rangle}$ from $\mathcal{G}^*$ according to one of the three measures Λ_1, Λ_2, Λ_3. $\mathcal{G}_{\langle w_1,w_2\rangle}$ is a restriction of $\mathcal{G}^*$ to the vertices on the most informative paths, i.e. directed, undirected or two-way paths,

[35] This is also true for shortest *directed* paths. However, using *undirected* paths mimics more closely the subgraphs resulting from Λ_1 and Λ_2. In this case, for instance, a vertex that only has outgoing *directed* paths to w_1 and w_2 may still be considered by Λ_3 in the resulting subgraph $\mathcal{G}_{\langle w_1,w_2\rangle}$.

relating specifically w_1 and w_2. $\mathcal{G}_{\langle w_1, w_2 \rangle}$ is thus a relation region determined by w_1 and w_2. Having gathered the relation region between w_1 and w_2, the next crucial step is to determine the informativeness of a vertex w_j itself as it resides on the paths in $\mathcal{G}_{\langle w_1, w_2 \rangle}$. A Λ measure for w_j only gives the informativeness of w_j with respect to some shortest paths between w_j and w_1, w_2. By contrast, we would like to estimate the informativeness of w_j in $\mathcal{G}_{\langle w_1, w_2 \rangle}$ as given by the general graph structure underlying the connections and paths in $\mathcal{G}_{\langle w_1, w_2 \rangle}$. This not only allows one to determine the informativeness of a single *word* in the relation region $\mathcal{G}_{\langle w_1, w_2 \rangle}$, but also allows one to compare different relation regions with each other by comparing the informativeness values of the words in them. The next part of $\mathcal{R}$ thus uses *random walks with non-uniform jumps* to determine informativeness values for the vertices in $\mathcal{G}_{\langle w_1, w_2 \rangle}$.

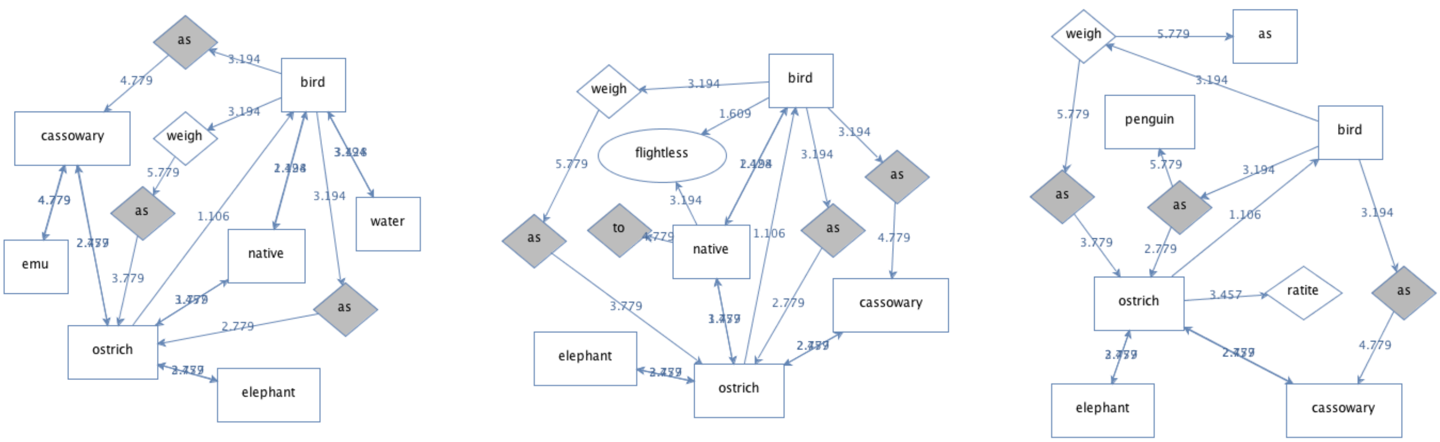


Fig. 7. Subgraphs extracted from the word-graph in Fig. 5 that include the topmost 11 vertices. The vertices are ranked according to respectively the Λ_1, Λ_2 and Λ_3 measures; the ranks are not shown. Note, for instance, that the vertex ‘flightless’ is on a two-way path that is not a path, and ‘penguin’ is on an undirected path that is not a two-way path.

5.iv Estimating informativeness by random walks

We specify a more precise criterion for the informativeness of a vertex and corresponding word w_i within the path and graph structure in $\mathcal{G}_{\langle w_1,w_2\rangle}$. The first criterion that we aim to capture is as follows:

(C_1) w_i is informative if other informative words connect to it.

Assume for the moment that $\mathcal{G}_{\langle w_1,w_2\rangle}$ that contains κ vertices satisfies the following two properties, where one sets $e_{i,j} = 0$ if there is no edge from w_i to w_j in $\mathcal{G}_{\langle w_1,w_2\rangle}$:

$$\forall i \ \forall j \ e_{i,j} \in \{0,1\} \tag{14}$$

$$\forall i \ \sum_{j}^{\kappa} e_{i,j} = 1 \tag{15}$$

A graph that satisfies Eq. 14-15 is called a *stochastic* graph. In a stochastic $\mathcal{G}_{\langle w_1,w_2\rangle}$, criterion C_1 translates as follows, where $\mathcal{I}_1(w_i)$ intuitively stands for the informativeness of w_i:

$$\mathcal{I}_1(w_i) = \sum_{j}^{k} e_{j,i} \ \mathcal{I}_1(w_j) \tag{16}$$

We see shortly below that $I(w_i)$ as in Eq. 16 can be obtained as the *stationary probability* of w_i given by a *random walk* over $\mathcal{G}_{\langle w_1,w_2\rangle}$ modeled as a *Markov chain*.

An additional criterion seems to be desirable for estimating the informativeness of a word w_i in $\mathcal{G}_{\langle w_1,w_2\rangle}$:

(C_2) w_i is informative if it connects to other informative words.

Criterion C_2 translates into the following:

$$\mathcal{I}_2(w_i) = \sum_{j}^{k} e_{i,j} \ \mathcal{I}_2(w_j) \tag{17}$$

It thus seems that the informativeness of a word $\mathcal{I}(w_i)$ may best be captured by a combination of the above two criteria:

$$\mathcal{I}(w_i) = \mathcal{I}_1(w_i) + \mathcal{I}_2(w_i) \tag{18}$$

The value $\mathcal{I}(w_2)$ may *approximately* be modeled as a random walk similar to $\mathcal{I}_1$, but with the edges in $\mathcal{G}_{\langle w_1,w_2\rangle}$ *reversed* and appropriately weighted so as to obtain a stochastic graph.[36] We return to the values $\mathcal{I}_2$ and $\mathcal{I}$ only briefly in Sect. 7; here we focus on $\mathcal{I}_1$ instead. Let us then describe some of the details behind estimating the value $\mathcal{I}_1$.

Random walks with non-uniform jumps Let $\mathcal{G}$ be a stochastic word-graph with n vertices. Consider a *random walker* placed at a vertex w_i in $\mathcal{G}$, randomly choosing to step to one of the vertices w_j for which there is a directed edge from w_i. Since $\mathcal{G}$ satisfies Eq. 14-15, we can interpret the weight $e_{i,j}$ of the edge from w_i to w_j as the probability that the walker chooses w_j. As the walker proceeds to take steps from one vertex to another, the walker steps through certain vertices more often than through others. The more ingoing edges a vertex has and the higher the corresponding probabilities, the more often the vertex will be visited.[37] Note that the probability of the walker stepping from w_i to w_j depends only on the probability $e_{i,j}$ of the edge (i.e. its weight in the stochastic graph), not, for instance, on where the walker came from; this is called the *Markov property* of the random walk. In other words, the random walk is modeled as a *Markov chain* [29].

Specifically, a *Markov chain* M consists of a set W of *states* and a $n \times n$ *transition probability matrix* E, where the values $E_{i,j}$ in E satisfy Eq. 14-15. E is thus a *stochastic matrix*. Given an n-dimensional row *probability vector* π, we write

$$\pi^t := \pi E^t \tag{19}$$

Thus, $(\pi^t)_j$ is the probability of reaching state j after t steps in M beginning from the probability distribution π over the states in W.

M is said to be *irreducible* if, for any $i, j \leq n$, there is a sequence $E_{i,k_1}, ..., E_{k_n,j}$, of non-zero transition probabilities that starts at state i and ends at state j. A state i is said to be *aperiodic* if there is an m such that for all $m' \geq m$, there is a sequence of m' non-zero transition probabilities that starts at state i and ends at state i. Further, M is *aperiodic* if all of its states are. For a finite irreducible Markov chain M, if one of its states is aperiodic, then so is M.

A central theorem for Markov chains is as follows.

[36] This would only be an approximation, as the values $e_{i,j}$ must be modified when the edges are reversed in order for the graph to be stochastic.

[37] The informativeness of the vertex will be closely related to the probability with which the vertex is visited by the random walker.

Theorem 1. *[29] For a finite state, irreducible and aperiodic Markov chain M, there is a unique* stationary *probability vector $\mathit{\Pi}$ that is the left eigenvector of E, such that for any probability vector π,*

$$\lim_{t\to\infty} \pi^t = \mathit{\Pi}$$

From the theorem it follows that,

$$\mathit{\Pi} E = \mathit{\Pi} \tag{20}$$

and, by indicating with $E_{*,i}$ the ith column of E, that,

$$\mathit{\Pi}_i = \mathit{\Pi} E_{*,i} = \sum_{j}^{n} E_{j,i} \mathit{\Pi}_j \tag{21}$$

Assuming now that the adjacency matrix of $\mathcal{G}_{\langle w_1, w_2 \rangle}$ and the word vertices $\mathcal{W}$ and edges $\mathcal{E}$ form an *irreducible* and *aperiodic* Markov chain, letting $\mathcal{I}_1(w_i) = \mathit{\Pi}_i$ and having that $e_{j,i} = E_{j,i}$, Eq. 21 is precisely the desired criterion C_1 formulated in Eq. 16.

We must next describe how *any* word-subgraph $\mathcal{G}_{\langle w_1, w_2 \rangle}$ may be turned into an irreducible and aperiodic Markov chain. The trick is as used in the *Personalized PageRank* algorithm for ranking World Wide Web pages [34]. Given the interpretation of the Markov chain as a random walk, the trick involves endowing the walker with the ability to *jump* from any vertex i to any other vertex j according to some prior probability $J_j \in [0, 1]$ determined at the outset and depending only on j. That is, at each step, with probability $\gamma \in [0, 1]$ the walker chooses the *jump* operation, where the walker jumps from the current vertex i to vertex j with probability J_j; on the other hand, with probability $1 - \gamma$, the random walker selects as before one of the vertices having ingoing edges from the current vertex i. We thus have here two parameters:

$$\mathcal{P}_6 :: \textit{jump probability } \gamma$$

$$\mathcal{P}_7 :: \textit{priors distribution } J$$

It turns out that the adjacency matrix of any $\mathcal{G}_{\langle w_1, w_2 \rangle}$ properly adjusted to a stochastic matrix that includes jump and prior probabilities γ and J is irreducible and aperiodic.[38]

[38] For details, see [29]

Besides making the adjacency matrix aperiodic and irreducible, an appropriate choice of γ and J may have other significant effects. For example, without γ and J, if one just turns the adjacency matrix of $\mathcal{G}_{\langle w_1, w_2 \rangle}$ into a stochastic matrix, much of the PPMI weighting may be lost. One makes an adjacency matrix stochastic by replacing an edge weight $e_{i,j}$ by the value,

$$\frac{e_{i,j}}{\sum_l^{\kappa} e_{i,l}}$$

This ensures that Eq. 14-15 are satisfied. But now consider some vertices w_k and w_l with a single outgoing edge with weight, respectively, $e_{k,k'}$ and $e_{l,l'}$. No matter what the weights $e_{k,k'}$ and $e_{l,l'}$ are, after the stochastic weighting they are both simply equal to 1. To somewhat alleviate this difficulty, one may select J so as to give higher priors to vertices whose outgoing or ingoing edges tend to have higher values relatively to those of other vertices and choose a significant probability γ to give weight to such priors. We will see an example of J in Sect. 6. After the incorporation of γ and J into the stochastically weighted adjacency matrix of $\mathcal{G}_{\langle w_1, w_2 \rangle}$, a vector of *stationary* probabilities Π_J^{γ} can easily and robustly be computed determining the informativeness of a word w_i in $\mathcal{G}_{\langle w_1, w_2 \rangle}$ according to criterion C_1, modulo small differences due to the incorporation of γ and J. This concludes the method by which $\mathcal{R}$ estimates the informativeness of a vertex in the relation region given by the subgraph $\mathcal{G}_{\langle w_1, w_2 \rangle}$.

To sum up, the vector of informativeness values for vertices in the *relation region* of $\mathcal{G}^*$ determined by w_1 and w_2 is computed in $\mathcal{R}$ as follows:

1. Given initiating vertices $\langle w_1, w_2 \rangle$ and word-graph $\mathcal{G}^*$, apply PPMI weighting to $\mathcal{G}^*$;
2. Use measures Λ_1, Λ_2 or Λ_3 to obtain the subgraph $\mathcal{G}_{\langle w_1, w_2 \rangle}$ with the κ vertices on the most informative paths;
3. Merge *all* the vertices in $\mathcal{G}_{\langle w_1, w_2 \rangle}$ that have the same word and pos-type;
4. Determine J using the PPMI weighting in $\mathcal{G}_{\langle w_1, w_2 \rangle}$, and using β and J transform the adjacency matrix of $\mathcal{G}_{\langle w_1, w_2 \rangle}$ into an irreducible and aperiodic Markov chain;
5. Compute the vector of stationary probabilities Π_J^{γ} given by the Markov chain; the stationary probability $(\Pi_J^{\gamma})_i$ of vertex w_i reflects the informativeness of the word in the relation region $\mathcal{G}_{\langle w_1, w_2 \rangle}$, according to criterion C_1.

Note step 3. Vertices of certain pos-types might still not be merged by that stage, depending on the original construction of the word-graph $\mathcal{G}$ and the merging function μ; we merge them all together before applying the random walk.

We thus finally obtain a concrete, *relation-centered* representation $\langle \boldsymbol{w_1}, \boldsymbol{w_2} \rangle^{\mathsf{r}}$ for the verbal analog $\langle \mathbf{w_1}, \mathbf{w_2} \rangle$, by setting:

$$(\langle \boldsymbol{w_1}, \boldsymbol{w_2} \rangle^{\mathsf{r}})_j := \begin{cases} \Pi_J^{\gamma} & \text{if } w_j \text{ is in } \mathcal{G}_{\langle w_1, w_2 \rangle} \\ 0 & \text{otherwise} \end{cases} \tag{22}$$

Fig. 5 illustrates the importance values resulting from applying $\mathcal{R}$ to the word-graph of Fig. 3 three times, one for each of the possible Λ measures, where the intermediate subgraphs are depicted above in Fig. 4. Let us briefly show how the algorithm is modified in order to use $\mathcal{R}$ to compute informativeness values in the meaning region determined by w_1 alone.

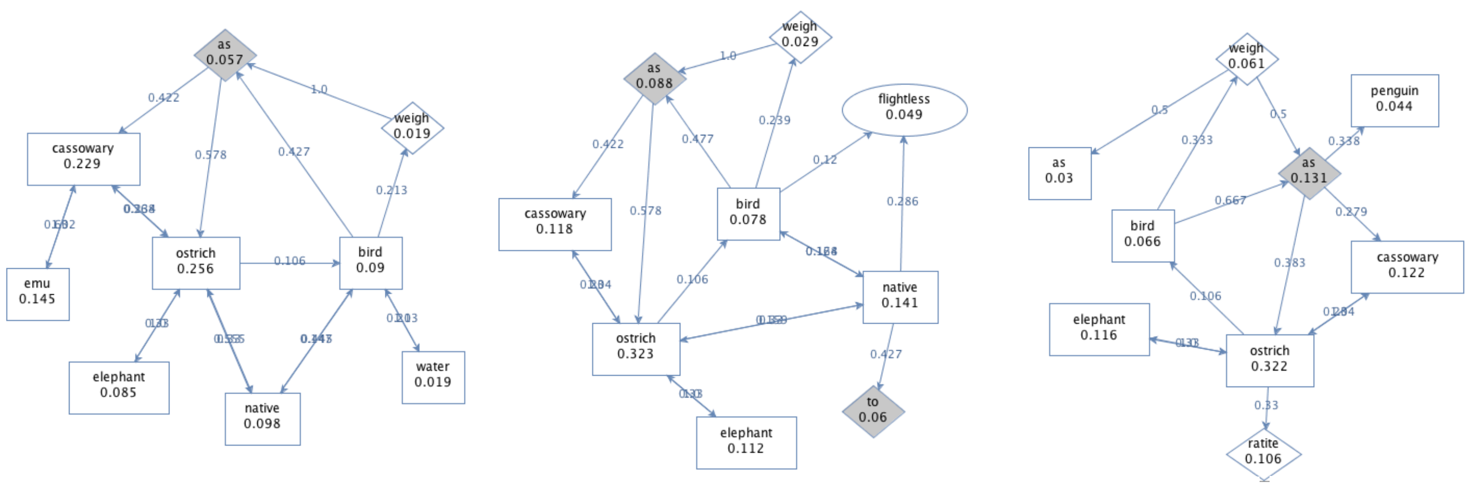


Fig. 8. Fully merged subgraphs with informativeness values resulting from applying $\mathcal{R}$ to the word-graph from Fig. 3, using respectively the Λ_1, Λ_2 and Λ_3 measures. The adopted priors vector J is as explained in Sect. 6.

Meaning regions in $\boldsymbol{\mathcal{R}}$ The variant of $\mathcal{R}$ that computes informativeness values for the vertex in the meaning region of w_1 according to criterion C_1 requires only a modification to step 2. The $\mathit{\Lambda}$ measures must be modified into $\mathit{\Lambda}^{\mathsf{m}}$ measures by dropping any λ or λ' term containing the vertex w_2. This results in the new measures $\mathit{\Lambda}_1^{\mathsf{m}}$ and $\mathit{\Lambda}_2^{\mathsf{m}}$ actually being the same. The new $\mathit{\Lambda}^{\mathsf{m}}$ are path distance measures only with respect to paths from and to w_1. The resulting subgraph $\mathcal{G}_{w_1}$ is centered around w_1. The remaining steps 3-5 are identical to those in $\mathcal{R}$ as specified above.[39] Let us now describe algorithm $\mathcal{S}$

5.v Spreading activation

Given the word-graph $\mathcal{G}^*$ and the initiating vertices w_1 and w_2, the $\mathcal{S}$ algorithm adopts a direct method of assigning informativeness values to the vertices surrounding w_1 and w_2, without needing path distance measures. The method captures the following, somewhat underspecified, criterion of informativeness that is *relative* to a vertex w_j.[40] We say that a directed path is *simple* if it does not have any repeated vertices; the *discrete length* of a directed path is defined as the number of connections it has; the *weight* of a path is defined as the sum of the weights of the connections in the path. Then, we may state the criterion as follows:

(C_3) w_i is more informative with respect to w_j, the greater the *number* of distinct simple directed paths from w_j to w_i, the smaller their *discrete lengths*, and the greater their *weights*.

The aforementioned method is the *spreading activation* algorithm. The variant given below is in essence a *depth-first search* through the graph starting from a vertex w_1 that is given an initial *activation value*. The search proceeds by decreasing the activation value by a *global decay* factor each time a vertex is visited. The activation value is summed to the previous value of the vertex (that is initially 0). If the vertex's activation value is lower than a given *firing threshold*, the search stops at that vertex. The pseudocode for the algorithm is given in Fig. 9, where `a(v)` is the activation value assigned to vertex `v` or null if not assigned,

[39] It would be possible to obtain a *meaning-centered* representation $\langle \boldsymbol{w_1}, \boldsymbol{w_2} \rangle^{\mathsf{m}}$ for the verbal analog based on these meaning regions. We do not presently consider this with $\mathcal{R}$, but only below with $\mathcal{S}$.

[40] The algorithm specified below yields precise informativeness values, but a non-specific statement of the nature of the values suffices for explanatory purposes.

```
SpreadActivation (Vertex v, Path p)
     if a(v) < firingThreshold
         return
     add v to p
     for each edge e outgoing from v
         let v' be the target of e
         let x be the weight of e
         if a(v') is null
             set a(v') = 0.0
         if v' is not in p
             let x = a(v') + a(v) * x * globalDecay
             set a(v') = min(x, maxActivation)
             SpreadActivation (v', p)
```

Fig. 9. Depth-first spreading activation algorithm

`maxActivation` is normally set to 1.0, and `firingThreshold` and `globalDecay` are free parameters:

$$\mathcal{P}_8 :: \textit{firing threshold}$$

$$\mathcal{P}_9 :: \textit{global decay}$$

Given the values `a(w)` for every vertex w obtained from spreading activation from w_1, since some of the `a(w)` might actually be *null*, we let the *outgoing activation vector* $\mathcal{S}^{out}(w_1)$ from w_1 defined on vertices w be as follows:

$$\mathcal{S}^{out}(w_1)(w) := \begin{cases} \texttt{a(w)} & \text{if } \texttt{a(w)} \text{ is not null} \\ 0 & \text{otherwise} \end{cases} \tag{23}$$

Further, let the *ingoing activation vector* $\mathcal{S}^{in}(w_1)$ from w_1 be similarly defined, but with the `a(w)` values obtained from a variant of the *spreading activation* algorithm, where `outgoing from` is replaced with `ingoing into` in the procedure `SpreadActivation`. In other words, $\mathcal{S}^{in}(w_1)$ is the activation vector obtained by spreading the activation values from w_1 over *ingoing* edges.

A technical transformation must be applied to the PPMI weighted $\mathcal{G}^*$ before spreading activation can reliably be used. To avoid unexpected behavior, all edge weights must be in the interval $[0, 1]$. This is ensured by *normalizing* the edge weights in $\mathcal{G}^*$, i.e. by replacing each edge weight $e_{i,j}$ by the value,

$$\frac{e_{i,j}}{f_{\mathcal{G}^*}}$$

where $f_{\mathcal{G}^*}$ is the *greatest* edge weight occurring in $\mathcal{G}^*$. This concludes the description of the spreading activation algorithm. Let us now see how activation vectors may be combined algebraically to capture the information of various regions in $\mathcal{G}^*$; this constitutes the next part of $\mathcal{S}$.

5.vi Algebraic combinations of activations

As we saw in the previous section, $\mathcal{S}$ acts directly on the whole *normalized* word-graph $\mathcal{G}^*$ and uses the delimiting capacity of spreading activation to obtain activation vectors with respect to an initial vertex w_1. The next step combines such activation vectors by way of *algebraic operations*. Different algebraic operations, and pairs thereof, yield *high* informativeness values for different regions of $\mathcal{G}^*$; the interpretation of these regions may likewise differ. Even though some algebraic operations, and pairs thereof, clearly do not yield any meaningful informativeness values, we still specify two *schemas* for combining activation vectors and leave the algebraic operations as free parameters.

To this end, given the initiating vertices w_1, w_2, let $\mathcal{S}^{out}(w_1)$, $\mathcal{S}^{in}(w_1)$, $\mathcal{S}^{out}(w_2)$, $\mathcal{S}^{in}(w_2)$ be as defined in Sect. 5.v. The two schemas $\mathcal{S}(w_1, w_2)$ and $\mathcal{S}'(w_1, w_2)$ are given as follows,

$$\mathcal{S}(w_1, w_2) = (\mathcal{S}^{out}(w_1) \circledast \mathcal{S}^{in}(w_2)) \odot (\mathcal{S}^{out}(w_2) \circledast \mathcal{S}^{in}(w_1)) \tag{24}$$

$$\mathcal{S}'(w_1, w_2) = (\mathcal{S}^{out}(w_1) \circledast \mathcal{S}^{in}(w_1)) \odot (\mathcal{S}^{out}(w_2) \circledast \mathcal{S}^{in}(w_2)) \tag{25}$$

where the operations $\circledast$ and $\odot$ are applicable algebraic operations on vectors such as such as $-$, $+$, point-wise $*$, *max* or *min*, the tensor product $\otimes$, and possibly others. These are left as parameters:

$$\mathcal{P}_{10} :: \textit{inner operation } \circledast$$

$$\mathcal{P}_{11} :: \textit{outer operation } \odot$$

Let us thus illustrate some concrete instances of $\mathcal{S}(w_1, w_2)$ and $\mathcal{S}'(w_1, w_2)$. Consider $\mathcal{S}(w_1, w_2)$ and suppose that $\circledast = \mathit{min}$ and $\odot = \mathit{max}$. For a vertex w_j of

$\mathcal{G}^*$, the value $min(\mathcal{S}^{out}(w_1), \mathcal{S}^{in}(w_2))_j$ will be high in the range $[0, 1]$ if and only if both the activation values $\mathcal{S}^{out}(w_1)_j$ and $\mathcal{S}^{in}(w_2)_j$ are high. But the latter is the case if and only if both w_j is highly informative with respect to w_1 and w_2 is highly informative with respect to w_j. In other words, there are very significant directed paths from w_1 to w_j and from w_j to w_2. Thus, the vertices with high value $min(\mathcal{S}^{out}(w_1), \mathcal{S}^{in}(w_2))$ make up a *region* of $\mathcal{G}^*$ of *relations from* w_1 *to* w_2, in that order. By a symmetric argument, the vertices with high value $min(\mathcal{S}^{out}(w_2), \mathcal{S}^{in}(w_1))$ make up a *region* of $\mathcal{G}^*$ of *relations from* w_2 *to* w_1. By taking the *max* of the values of the vertices within these two regions, one obtains a further region of $\mathcal{G}^*$ capturing the relations between w_1 and w_2 in *both* directions. To wit, *min* and *max* have an effect of, respectively, *graded intersection* and *graded union*. A similar argument works for the selection $\circledast = *$ and $\odot = +$, as the latter in the range $[0, 1]$ have effects similar to those of *min* and *max*, respectively.

Now, consider $\mathcal{S}'(w_1, w_2)$ and suppose that $\circledast = max$ (or $\circledast = +$), and $\odot = -$. $max(\mathcal{S}^{out}(w_1), \mathcal{S}^{in}(w_1))$ takes the graded union of regions that have vertices that are highly informative with respect to w_1 and with respect to which w_1 is highly informative. That is, $max(\mathcal{S}^{out}(w_1), \mathcal{S}^{in}(w_1))$ captures the *meaning* region of w_1. If we let the *representation* of the meaning $\boldsymbol{w_1}$ simply be,

$$\boldsymbol{w_1} = max(\mathcal{S}^{out}(w_1), \mathcal{S}^{in}(w_1)) \tag{26}$$

and similarly for $\boldsymbol{w_2}$, and consider the operation $\odot = -$ on $\boldsymbol{w_1}$ and $\boldsymbol{w_2}$, we essentially reconstruct the representation $(\boldsymbol{w_1} - \boldsymbol{w_2})$ from the *MAR* model (Eq. 8).[41]

How about the pair $\circledast = *$ and $\odot = -$ applied to $\mathcal{S}(w_1, w_2)$? As before, vertices with high value in $(\mathcal{S}^{out}(w_1) * \mathcal{S}^{in}(w_2))$ make up a region of *relations from* w_1 *to* w_2. Vertices with high value in $(\mathcal{S}^{out}(w_2) * \mathcal{S}^{in}(w_1))$ make up a region of relations *from* w_2 to w_1. One may think of the former as a characterization of w_1 *as it relates to* w_2. Similarly, one may think of the latter as a characterization of w_2 as it relates to w_1. Applying now the outer operation $-$ on the two vectors returns an instance of $\mathcal{S}(w_1, w_2)$. The interpretation, geometric and theoretical, of the effect of $-$ on the resulting $\mathcal{S}(w_1, w_2)$ is similar to the one

[41] The vectors $\boldsymbol{w_1}$ and $\boldsymbol{w_2}$ are switched here. If the presumed $\boldsymbol{w_3}$ and $\boldsymbol{w_4}$ are switched just in the same way, this has no effect on the similarities as calculated by the measures in Sect. 2.v applied to Eq. 8. Also, switching back presents no difficulties.

in MAR (Sect. 3.ii); the difference is that in MAR one applies subtraction – of *meaning representations* $\boldsymbol{w_1}$ and $\boldsymbol{w_2}$, whereas here the subtraction is applied to representations of w_1 and w_2 as they *relate*, respectively, to w_2 and w_1.[42] The resulting $\mathcal{S}(w_1, w_2)$ captures both the theoretical insight of MAR and the insight on the importance of *relations* suggested in SMT and incorporated into LRA and *DM*. This representation turns out to be crucial from an experimental point of view and we return to it in Sect. 6.

Before we can deduce specific representations $\langle \boldsymbol{w_1}, \boldsymbol{w_2} \rangle$ of the desired verbal analog $\langle \mathbf{w_1}, \mathbf{w_2} \rangle$, we must briefly deal with vertices that may still not be merged in $\mathcal{G}^*$. We merge these vertices together in $\mathcal{G}^*$ by *summing* their values as obtained from one of $\mathcal{S}(w_1, w_2)$ and $\mathcal{S}'(w_1, w_2)$. This results in vectors $\nu(\mathcal{S}(w_1, w_2))$ and $\nu(\mathcal{S}'(w_1, w_2))$ with somewhat fewer dimensions than the originals.

We may finally give the above three instances of $\mathcal{S}(w_1, w_2)$ and $\mathcal{S}'(w_1, w_2)$ as concrete representations for $\langle \boldsymbol{w_1}, \boldsymbol{w_2} \rangle$. Thus, we have the *meaning-centered* one similar to that in MAR (from Eq. 25):

$$\langle \boldsymbol{w_1}, \boldsymbol{w_2} \rangle^{\mathsf{m}} = \nu((\mathcal{S}^{out}(w_1) * \mathcal{S}^{in}(w_1)) - (\mathcal{S}^{out}(w_2) * \mathcal{S}^{in}(w_2))) \tag{27}$$

We further have the *relation-centered* ones (from Eq. 24):

$$\langle \boldsymbol{w_1}, \boldsymbol{w_2} \rangle^{\mathsf{r}} = \nu(max(min(\mathcal{S}^{out}(w_1), \mathcal{S}^{in}(w_2)), min(\mathcal{S}^{out}(w_2), \mathcal{S}^{in}(w_1)))) \tag{28}$$

$$\langle \boldsymbol{w_1}, \boldsymbol{w_2} \rangle^{\mathsf{r}} = \nu((\mathcal{S}^{out}(w_1) * \mathcal{S}^{in}(w_2)) + (\mathcal{S}^{out}(w_2) * \mathcal{S}^{in}(w_1))) \tag{29}$$

$$\langle \boldsymbol{w_1}, \boldsymbol{w_2} \rangle^{\mathsf{r}} = \nu((\mathcal{S}^{out}(w_1) * \mathcal{S}^{in}(w_2)) - (\mathcal{S}^{out}(w_2) * \mathcal{S}^{in}(w_1))) \tag{30}$$

We, somewhat arguably, consider the latter also a *relation-centered* representation. Some further concrete representations $\langle \boldsymbol{w_1}, \boldsymbol{w_2} \rangle$ will be tested in Sect. 6. To sum up, $\mathcal{S}$ involves the following steps:

1. Given initiating vertices $\langle w_1, w_2 \rangle$ and word-graph $\mathcal{G}^*$, apply PPMI weighting to $\mathcal{G}^*$;
2. Apply *normalization* to $\mathcal{G}^*$;
3. Apply spreading activation to obtain activation vectors $\mathcal{S}^{out}(w_1)$, $\mathcal{S}^{in}(w_1)$, $\mathcal{S}^{out}(w_2)$, $\mathcal{S}^{in}(w_2)$;

[42] The representation of w_1 as it relates to w_2 may also be viewed as the representation of that part of the meaning of w_1 that is most relevant or central to w_2. Some of the vertices that have high value in the *meaning* representation of $\boldsymbol{w_1}$, those "closest" to w_2, will also have high value in the latter representation.

4. Use the given algebraic operations $\circledast$ and $\odot$ to compute $\mathcal{S}(w_1, w_2)$ and $\mathcal{S}'(w_1, w_2)$;
5. Apply the *merging* procedure ν, that merges all the vertices with same word and pos-type.

This concludes the description of $\mathcal{S}$. Let us finally consider $\mathcal{T}$.

Combining spreading activation with path distance measures The algorithm $\mathcal{T}$ adopts the Λ or Λ^{m} measures from Sect. 5.iii to obtain relation regions $\mathcal{G}_{\langle w_1, w_2 \rangle}$ or meaning regions $\mathcal{G}_{w_1}$. Then, instead of estimating informativeness with random walks, it estimates it with spreading activation. Eq. 24-25 are thus applied directly to the *subgraph* $\mathcal{G}_{\langle w_1, w_2 \rangle}$ or $\mathcal{G}_{w_1}$. In this case, it is possible to simplify Eq. 24-25 by dropping the $\mathcal{S}^{in}$ terms and the inner operation, as the extraction of an appropriate subgraph has already been performed through the Λ or Λ^{m} measures. Fig. 10 gives informativeness values calculated through spreading activation on subgraphs obtained from the word-graph in Fig. 5; the subgraphs are the same as those in Fig. 7. Concrete representations $\langle \boldsymbol{w_1}, \boldsymbol{w_2} \rangle^{\mathsf{r}}$ are obtained analogously to Eq. 22. This concludes the description of $\mathcal{T}$.

* * *

In Sect. 5, we have considered regions of word-graphs capturing information pertaining to concepts such as meanings and relations. We have presented two main algorithms $\mathcal{R}$ and $\mathcal{S}$ and a variation $\mathcal{T}$ for extracting such regions and the informativeness values of the words in the regions from a word-graph. We have thus obtained some concrete representations $\langle \boldsymbol{w_1}, \boldsymbol{w_2} \rangle$ of verbal analogs. In the last main section, we now investigate the experimental accuracies of such representations and what these suggest about meaning and analogy.

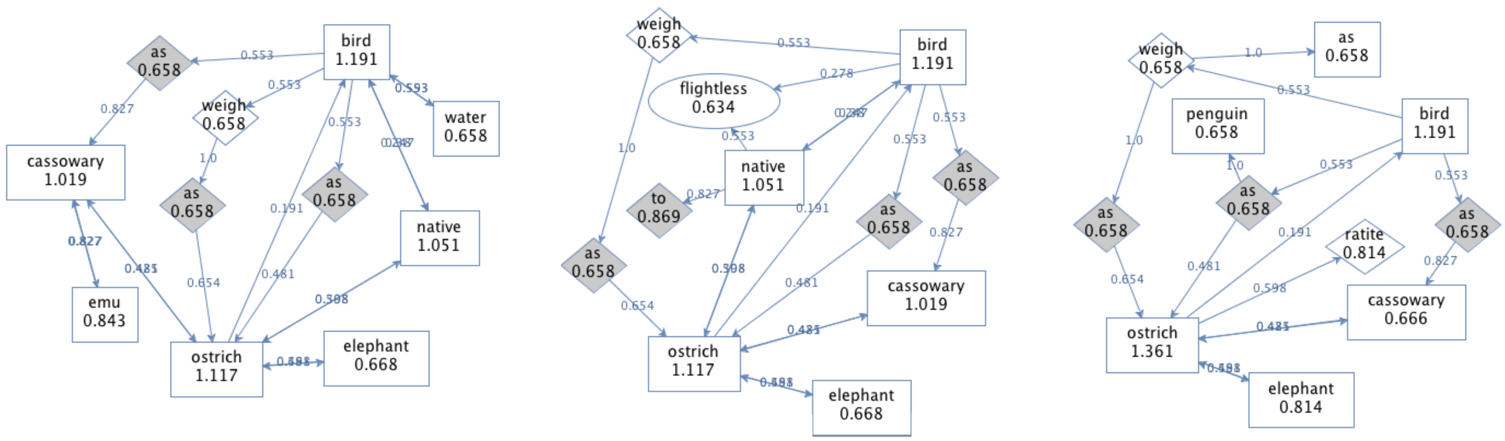


Fig. 10. Subgraphs resulting from applying $\mathcal{T}$ to the word-graph from Fig. 5, using respectively the Λ_1, Λ_2 and Λ_3 measures. Firing threshold is set to 0.1, global decay is 1.0 and the outer operation is $+$. This operation allows for the final, summed values to be greater than 1.

6 Accuracy Results and Experimental Findings

We first describe the way the parameters left free in Sect. 4-5 are assigned. The description of the parameters includes some implementation details. Then, we report on the initial batch of experiments that has been performed and we state and compare the respective accuracy values.[43] Finally, we examine the accuracy values of the algorithms from a more theoretical perspective, illustrating how the highest accuracy value for an algorithm is obtained by merging theoretical insights from MAR with those from SMT.

6.i Parameters

We here describe the way one selects the parameters that are left free in Sect. 4-5; these parameters are adopted in the experiments reported below. We also relate some of the details of the implementation of the framework.

Corpus of language $\mathcal{L}$ The collection of analogy problems consists of 374 problems, each in turn made up of a *source* analog and five *target* analogs; one must choose the target analog whose analogical strength with the source is highest (Sect. 3.i). For each problem P of the 374 problems, we generate a small corpus $\mathcal{L}_P$ containing *sentences* relevant to the 6 analogs in P. Each small corpus $\mathcal{L}_P$ is generated from a large corpus $\mathcal{L}^*$.

The large corpus $\mathcal{L}^*$ consists of *preprocessed, sentence tokenized and tokenized* versions of the UkWac corpus and of Wikipedia.[44] UkWac and Wikipedia together consist of about $3{*}10^9$ tokens.[45] Wikipedia is first preprocessed to extract only the *text* from it; this is done using the Wikiprep script [15,16] and additional Perl scripts. Both UkWac and Wikipedia are then tokenized and sentence tokenized into the Penn TreeBank format using the NLTK Python toolkit [3].

[43] We do not presently report any experimental results on the $\mathcal{T}$ algorithm.

[44] We use an early-2012 XML dump of Wikipedia.

[45] Much of this corpus is identical with the corpus used in DM and in the restricted version of LRA [2]; the corpus in DM and LRA includes the British National Corpus [6], whereas $\mathcal{L}^*$ includes a somewhat larger version of Wikipedia.

Next, for each word pair $\langle w_1, w_2\rangle$ in P, *synonyms* of w_1 and w_2 are determined using WordNet [13,31] and the Adapted Lesk algorithm for *word disambiguation* [1]. Of all the *synonym sets* (*synsets*) of w_1 in WordNet, the one synset s is chosen that has the *highest* Lesk score with any of the synsets of w_2. The words in s are chosen as synonyms for w_1. Thus, w_1 is disambiguated using w_2 as "context" and the words from the corresponding synset are taken as synonyms for w_1. The same is performed for w_2 with respect to w_1. Let us indicate by $s_1, ..., s_n$ and $t_1, ..., t_m$ the determined synonyms, respectively, of w_1 and w_2.

Now $\mathcal{L}_P$ is extracted from $\mathcal{L}^*$ as follows. For each word pair $\langle w_1, w_2\rangle$ in P, a *Boolean query* of the following form is constructed:

$$(w_1 \wedge w_2) \vee \bigvee_i^n (w_1 \wedge s_i) \vee \bigvee_i^m (t_i \wedge w_2) \tag{31}$$

The Apache Lucene search engine is then used to search $\mathcal{L}^*$ using the query from Eq. 31; up to 10,000 of the most relevant sentences found by this query make up the corpus $\mathcal{L}_P$ for the problem P. Each sentence in turn contains either w_1 and w_2, or w_1 and one of the synonyms t_i of w_2, or w_2 and one of the synonyms s_i of w_1. This is how $\mathcal{L}_P$ is generated. Parameter $\mathcal{P}_1$ is thus assigned the corpus $\mathcal{L}_P$. A word-graph for each problem P is then built according to the procedure described in Sect. 4.

Notice here the various aspects that attempt to deal with the *ambiguity* of the words w_1 and w_2. First, synonyms of each word are found by disambiguating with respect to the other of the two words. Secondly, the sentences in $\mathcal{L}_P$ tend to be about the intended meanings of w_1 and w_2, since both words (or their synonyms) must occur in the sentence. Hence, if 'cub' and 'bear' occur in a sentence, 'bear' tends to stand for the corresponding *animal* and not for the *act of support* or any other of its meanings.

Merging function $\boldsymbol{\mu}$ The merging function used in all of the present experiments is the same as the one mentioned in Sect. 4-5. It merges vertices with pos-type $\pi \in \{\mathsf{n}, \mathsf{v}, \mathsf{a}, \mathsf{d}\}$, whereas it does not merge *prepositions* vertices of pos-type i.

Maximum size of a word-graph Let $\mathcal{G}_P$ be the word-graph generated from $\mathcal{L}_P$. We experiment with maximum size parameters $\mathcal{P}_3$ of 10,000, 50,000 and *unlimited* (∞). A *finite limit* on the maximum size can be used to increase the efficiency of the optimization process, though significant linguistic information is generally lost in a *reduced size* word-graph (depending naturally on the extent of the reduction).

Initiating vertices selection $\boldsymbol{\alpha}$ Given a word-graph $\mathcal{G}_P$ and a pair $\langle w_1, w_2 \rangle$ together with their synonyms, the selection procedure α (parameter $\mathcal{P}_4$) that we presently adopt merges the vertex corresponding to w_1 in $\mathcal{G}_P$ with all the vertices corresponding to the synonyms $s_1, ..., s_n$ of w_1. It does the same with w_2 and its synonyms $t_1, ..., t_m$. Let the merged vertices be, respectively, w_1^* and w_2^*. α thus returns the new graph $\mathcal{G}_P^*$ and the *initiating vertices* w_1^* and w_2^*. The aim of merging the vertices is to accumulate into a single vertex the information in $\mathcal{G}_P$ concerning both w_1 (respectively w_2) and all of its synonyms.

Maximum size of subgraph For the maximum size κ (parameter $\mathcal{P}_5$) of a subgraph for a word pair $\langle w_1, w_2 \rangle$ generated during $\mathcal{R}$, we experiment with

$$\kappa \in \{200, 400, 800, 1000, 2000, 3000, ..., 30000\} \tag{32}$$

κ is thus serves as an *optimization* parameter. As we see below, it turns out that due to the robustness of random walks the size of κ within a large range does not affect accuracy significantly.

Jump probability $\boldsymbol{\gamma}$ and priors distribution $\boldsymbol{J}$ Given the word-graph $\mathcal{G}_P^*$ and a vertex w_i, we let its prior J_i be defined by:

$$J_i = \frac{\sum_j e_{i,j}}{\sum_k \sum_j e_{k,j}} \tag{33}$$

where $e_{i,j}$ is, as above, the weight of the edge from vertex w_i to w_j. Hence, if sum of the PPMI weights of the *outgoing* edges of w_i is relatively high, we assign to w_i a larger prior. This aims at alleviating the canceling effects of stochastic weighing in $\mathcal{R}$ on the PPMI weights. We briefly explore some possible variants of J in Sect. 7.

To give significance to the priors, we experiment with jump probability values $\gamma = 0.3$ and $\gamma = 0.4$. These are somewhat higher than the usually suggested 0.1 or 0.15 [29,34].

Firing threshold and global decay We optimize the threshold t (parameter $\mathcal{P}_8$) and decay d (parameter $\mathcal{P}_9$) values over the following ranges:[46]

$$t \in \{10^{-1}, ..., 10^{-9}\} \tag{34}$$

$$d \in \{1.0, 0.9, ..., 0.1\} \tag{35}$$

The firing threshold values t fall exponentially. This is to capture the exponential decrease in the activation values, due among others to d, as the activation spreads further away from the initial vertex.

Inner and outer operations There is a large number of possible algebraic operations to adopt for $\circledast$ and $\odot$.[47] We experiment with a few combinations involving $*$, $+$, $-$, max and min.

This completes the survey of the parameters. Let us now consider their accuracy when incorporated into the algorithms $\mathcal{R}$ and $\mathcal{S}$.

6.ii Experiments and Accuracy

Algorithm $\mathcal{R}$ The $\mathcal{R}$ algorithm coupled with the Λ_3 measure was tested with $\mathcal{P}_3$ set to *unlimited*; when $\mathcal{R}$ was coupled with the Λ_1 or Λ_2 measures $\mathcal{R}$ was tested with $\mathcal{P}_3$ set to 10,000 for efficiency reasons.[48] The results are reported in Table 1. $\mathcal{R}$ with the Λ_3 measure and unlimited $\mathcal{P}_3$ (top row) performs significantly better than $\mathcal{R}$ with the Λ_2 and reduced $\mathcal{P}_3$ (bottom row) according to the Fischer test ($p = 0.036$).[49] The accuracies for the Λ_1 measures (mid-rows) are not significantly different from the Λ_3 accuracies or the Λ_2 accuracies. It is interesting to note that, even on a *reduced* word-graph, $\mathcal{R}$ coupled with the Λ_1 measure performs comparatively to $\mathcal{R}$ coupled with the Λ_3 measure on a *unreduced* word-graph. A related observation stems from Table 2. It shows the accuracies of $\mathcal{R}$ coupled with Λ_3 by varying the parameter $\mathcal{P}_5$ corresponding to the maximum size of the subgraph. The accuracies are not significantly different from each other over a large part $(2,000 - 30,000)$ of the tested interval. This is likely to be a result of the general stability of random walks with jumps under perturbations of the *unimportant* nodes underlying the link structure [33].

[46] Similar ranges are used in [21].
[47] For a systematic study of some in a different framework, see [32].
[48] The average size of a word-graph across the 374 problems was about 74,000 vertices.
[49] The accuracies are significantly different according to the Fischer test if $p < 0.05$.

A further observations concerns similarity measures. As Table 2 suggest, using the L_1-norm similarity measure for representations obtained from the random walks in $\mathcal{R}$ consistently yields somewhat higher accuracies than those obtained using the *cosine* measure. This reflects the fact that such representations are in fact probability distributions and the length normalization implicit in the *cosine* measure does not seem to be meaningful in this case.

Λ meas.	$\mathcal{P}_3$	$\mathcal{P}_5$	$\mathcal{P}_6$	*sim. meas.*	*# non-skip*	*# correct*	*% correct non-skip*	*% correct*
Λ_3	∞	17,000	0.3	L_1-norm	366	122	33.3	32.6
Λ_1	10,000	2,250	0.3	cosine	359	113	31.5	30.2
Λ_1	10,000	2,000	0.4	cosine	355	109	30.7	29.1
Λ_2	10,000	4,000	0.4	L_1-norm	344	95	27.6	25.4

Table 1. Accuracy results from experiments with $\mathcal{R}$.

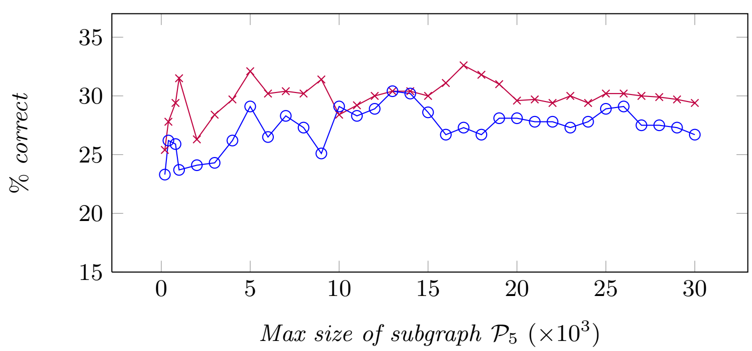


Table 2. Accuracy of $\mathcal{R}$ coupled with Λ_3. The upper line with cross points indicates accuracies measured with the L_1-norm, whereas the lower line with circled points indicates accuracies measured with the *cosine* measure.

Algorithm $\mathcal{S}$ We experimented with the $\mathcal{S}$ algorithm on *reduced* word-graphs with a limit on their maximum size $\mathcal{P}_3$ of $50,000$ vertices. Table 3 summarizes the results. We notice that the $\mathcal{S}(\cdot,\cdot)$ schema from Eq. 24 with inner operation $\circledast = *$ and outer operation $\odot = -$ (top row) has by some margin the highest accuracy. The difference is significant with respect to the lowest $\mathcal{S}(\cdot,\cdot)$ score reported (28.7%) according to the Fischer test ($p = 0.019$). The difference is significant also with respect to the two lowest $\mathcal{S}'(\cdot,\cdot)$ scores ($p = 0.043$ and $p = 0.009$, respectively). We consider the significance of these results in Sect. 6.iii.

It is interesting to observe the way accuracy changes as the threshold and decay parameters are varied. Table 4 shows the variations in accuracy of the $\mathcal{S}(\cdot,\cdot)$ schema with inner operation $\circledast = *$ and outer operation $\odot = -$. A decay value too close to 1.0 decreases somewhat the resulting accuracy. This suggests the expected fact that activating too large of a region in the word-graph has negative effects on accuracy. Variations in the firing threshold do not alter accuracy significantly. Let us thus consider the overall accuracies of the $\mathcal{R}$ and $\mathcal{S}$ algorithms as they compare to the *LRA* and *DM* models.

$\mathcal{P}_3$	*schema*	*inner* $\circledast$	*outer* $\odot$	*threshold* $\mathcal{P}_8$	*decay* $\mathcal{P}_9$	*sim. meas.*	*# non-skip*	*# correct*	*% correct non-skip*	*% correct*
50,000	$\mathcal{S}(\cdot,\cdot)$	$*$	$-$	10^{-4}	0.3	*cosine*	369	138	37.4	36.9
50,000	$\mathcal{S}(\cdot,\cdot)$	$*$	$+$	10^{-1}	0.6	*cosine*	369	115	31.2	30.7
50,000	$\mathcal{S}(\cdot,\cdot)$	*min*	*max*	10^{-1}	0.8	*cosine*	369	115	31.2	30.7
50,000	$\mathcal{S}(\cdot,\cdot)$	$*$	$+$	10^{-4}	0.3	*cosine*	369	107	29.0	28.7
50,000	$\mathcal{S}'(\cdot,\cdot)$	$+$	$+$	10^{-2}	0.9	*cosine*	369	119	32.2	31.8
50,000	$\mathcal{S}'(\cdot,\cdot)$	$+$	$*$	10^{-1}	0.8	*cosine*	369	111	30.1	29.7
50,000	$\mathcal{S}'(\cdot,\cdot)$	$+$	$-$	10^{-4}	0.7	*cosine*	369	104	28.2	27.8

Table 3. Accuracy results from experiments with $\mathcal{S}$.

Overall accuracies Table 5 reports the accuracy of the optimized algorithms $\mathcal{S}$ and $\mathcal{R}$, of *LRA* and of the three *DM* models [2]. The accuracy values are comparable as they are based on similar corpora (Sect. 6.i). The significance of the values is as follows. According to Fischer tests, $\mathcal{R}$'s accuracy is not significantly different from that of *LexDM* ($p = 0.342$), *DepDM* ($p = 0.753$), $\mathcal{S}$ ($p = 0.249$) and *LRA* ($p = 0.168$); the accuracy of $\mathcal{R}$ is significantly lower than that of *TypeDM* ($p = 0.008$). The $\mathcal{S}$ algorithm has an accuracy that is significantly higher than that of *LexDM* ($p = 0.0294$); $\mathcal{S}$'s accuracy is not sig-

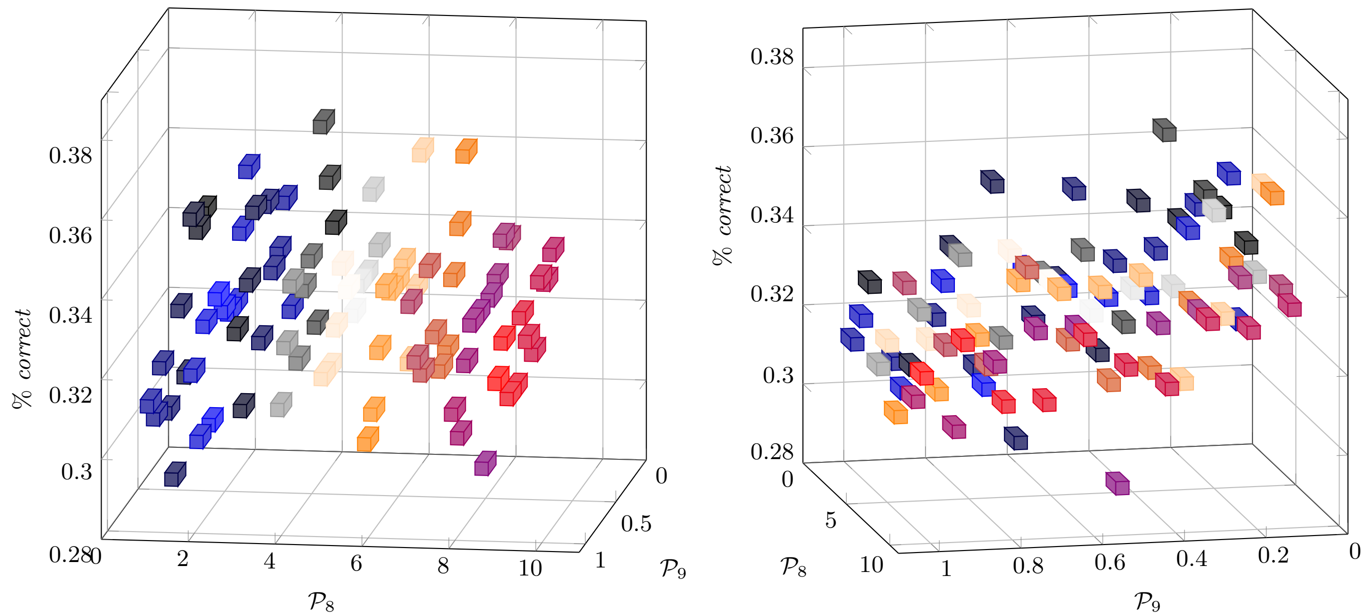


Table 4. Accuracies of algorithm $\mathcal{S}$ with schema $\mathcal{S}'(\cdot,\cdot)$, inner operation $\circledast = *$ and outer operation $\odot = -$. $\mathcal{P}_8$ is the threshold parameter (on a negated logarithmic scale, $x = -log(y)$, where y is the actual threshold value) and $\mathcal{P}_9$ is the decay parameter.

nificantly different from that of *DepDM* ($p = 0.122$), *LRA* ($p = 0.879$) and *TypeDM* ($p = 0.155$).

In the current specification of the framework, the weights on the edges are based on frequency in a way not dissimilar to the one adopted in *DepDM* and *LexDM*. Hence, what makes $\mathcal{S}$ significantly better than *LexDM* and nearly better than *DepDM* seems to be a combination of the graph-structure, the additional semantic analysis and the algorithm $\mathcal{S}$ with the specific inner and outer operations $*$ and $-$. From the results above, the graph-structure and semantic analysis do not appear to be sufficient by themselves; the choice of algebraic operations matters significantly.

A minor observation concerns similarity measures. We have seen above how the L_1-norm is a somewhat more accurate measure of similarity for probability distributions obtained from random walks in the $\mathcal{R}$ algorithm. It turns out that the *cosine* measure was more accurate in the case of representations obtained from the $\mathcal{S}$ algorithm. It thus seems that the additional normalization implicit in the cosine measure is effective for vectors of activation values.

Algorithm	*% correct*	*95% CI*
TypeDM	42.4	37.4−47.7
LRA	37.8	32.8−42.8
$\mathcal{S}$	36.9	32.1−41.8
$\mathcal{R}$	32.6	27.9−37.6
DepDM	31.4	26.6−36.2
LexDM	29.3	24.8−34.3

Table 5. Highest accuracy results for $\mathcal{S}$ and $\mathcal{R}$ and the LRA and DM models.

6.iii On theoretical insights into analogy

Theoretical insights about analogy may give rise to a particular kind of representations for verbal analogs. As we have seen in Sect. 3.ii, MAR postulates a meaning-centered view, according to which a representation for the verbal analog is the result of a subtraction of the representations of the meaning of the two words. Restricting our focus to the $\mathcal{S}$ algorithm, a most natural counterpart of the MAR representation is the $\mathcal{S}'(\cdot,\cdot)$ from Eq. 25 applied with $\circledast = +$ and $\odot = -$ (Sect. 5.iv). This achieves an accuracy of 27.8% (bottom row in Table. 2). Similarly, the most natural counterpart to the *relation-centered* view of relations within a verbal analog postulated by SMT and adopted in LRA and DM is the $\mathcal{S}(\cdot,\cdot)$ schema either with $\circledast = *$ and $\odot = +$, or with $\circledast = min$ and $\odot = max$. The latter two seem to have similar effects on activation values and indeed they both achieve the same accuracy of 30.7%. Even though the latter accuracy of the $\mathcal{S}(\cdot,\cdot)$ schema is somewhat higher than that of the $\mathcal{S}'(\cdot,\cdot)$ schema, the difference is not statistically significant. Thus, the evidence here does not yield a preference for either type of representations and respective insights.

By contrast, the accuracy of 36.9% of the $\mathcal{S}(\cdot,\cdot)$ schema with $\circledast = *$ and $\odot = -$ is significantly better than that of the schema $\mathcal{S}'(\cdot,\cdot)$ suggested by MAR. But what is the interpretation of the insight behind this higher performing schema $\mathcal{S}(\cdot,\cdot)$ for a verbal analog $\langle \mathbf{w_1}, \mathbf{w_2} \rangle$? The algebraic operation $(-)$ is applied to the following two vectorial representations. The first representation $\boldsymbol{r}_1$ is the one obtained by considering the *relations*, i.e. sequences of connections, *from* w_1 to w_2; the second representation $\boldsymbol{r_2}$ is the one obtained by considering the *relations from* w_2 to w_1. By interpreting the operation $(-)$ between two vectors geometrically as the *vectorial distance* between them, if, on the one hand, in MAR one takes the representation of the verbal analog $\langle \mathbf{w_1}, \mathbf{w_2} \rangle$ to be the *vectorial distance* of the *meaning* representations $\boldsymbol{w_1}$ and $\boldsymbol{w_2}$, on the present interpretation, one takes the representation of the verbal analog $\langle \mathbf{w_1}, \mathbf{w_2} \rangle$ to be the *vectorial distance* between the *one-sided* relational representations $\boldsymbol{r_1}$ and $\boldsymbol{r_2}$.[50] Thus, both the vectorial distance suggested by MAR and the centrality of the relations suggested by SMT play a crucial role in yielding in the present experiments the most highly performing representation of the verbal analog $\langle \mathbf{w_1}, \mathbf{w_2} \rangle$.

We conclude here our consideration on the experiments. Let us now end by considering in the last section possible additions and extensions of the current specification of the framework.

[50] It does not matter which of the vectors is subtracted from the other as long as this is done consistently.

7 Concluding Remarks

In this essay we have seen how one constructs a word-graph from a corpus of language and how one extracts by way of the $\mathcal{R}$ and $\mathcal{S}$ algorithms vectorial representations of the information in selected *regions* of the word-graph. Led by various theoretical insights and by viewing different such regions as incorporating information pertaining to verbal analogs, we have obtained various types of representations for verbal analogs of pairs. We have experimented with the resulting representations and those obtained from a variant of the $\mathcal{S}$ algorithm, notwithstanding a limiting parameter ($\mathcal{P}_3$), achieve an accuracy that is not significantly different from that of the state-of-the-art performing model $TypeDM$. We have also evaluated the representations induced by the theoretical insights behind MAR and SMT, not finding a significant difference between the respectively induced representations; we have pointed to a novel insight into analogy suggested by the significantly better accuracy of the $\mathcal{S}(\cdot, \cdot)$ schema representations couple with the operations $\circledast = *$ and $\odot = -$.

The possibilities of the framework have not been exhausted by the experiments reported in Sect. 6. Further, the framework may be altered in significant ways while maintaining its core ideas. A list of possible pathways to explore is as follows:

- Merging functions μ have a very significant effect on the way the information extracted from the corpus is represented in the word-graph. It would be possible to adopt different merging functions that merge different types of nodes. It would also be possible to deal with *ambiguity* by including more vertices for highly polysemous words; one such method is described in [23]. It would be possible to introduce *probabilistic* merging where each vertex is merged with a certain probability to achieve a specific factor of connectivity in the word-graph.
- The stability and robustness of random walks under various subgraph sizes is a highly desirable property. Random walks ought to be better exploited to achieve more accurate, but still highly robust representations. Random walks over reversed edges that approximately capture criterion C_2 may also yield improvements in accuracy (Sect. 6).
- It would be possible to generate word-graphs from precisely the links and weights used in the $TypeDM$ model together with appropriate merging func-

tions. This would in principle generalize the $TypeDM$ model, as the original representations would still be retrievable.

- Hyperlinks from Wikipedia and other online resources could easily be incorporated into word-graphs together with the connections stemming from the language corpus. Also, more semantic annotation coming from the Boxer analyzer as well as from other sources could also be incorporated.
- A single, larger word-graph would be desirable in order to be able to tackle many semantic tasks with a unique structure.
- The framework can be extended to capture representations of n-tuples for $n \geq 3$, either by selecting different appropriate regions in the word-graph, or by combining algebraically representations of verbal analogs of the pairs in the n-tuple.

The exploration of these pathways is left for future computational journeys into the workings of language.

References

1. Banerjee S., Pedersen T.: An adapted Lesk algorithm for word sense disambiguation using WordNet, Computational Linguistics and Intelligent Text Processing, LNCS Vol. 2276, (2002)
2. Baroni M., Lenci A.: Distributional Memory: A general framework for corpus-based semantics, Association for Computational Linguistics, 36 (4), (2010)
3. Bird, S., Loper E., Klein E : Natural Language Processing with Python. O'Reilly Media Inc. (2009)
4. Bos J., Clark S., Steedman M., Curran J. R., Hockenmaier J.: Wide-coverage semantic representations from a CCG parser, Proceedings of the 20th International Conference on Computational Linguistics (COLING-04), 1240–1246 (2004)
5. Bos J.: Wide-coverage semantic analysis with Boxer, STEP '08 Proceedings of the 2008 Conference on Semantics in Text Processing 277–286, (2008)
6. The British National Corpus, version 3 (BNC XML Edition). 2007. Distributed by Oxford University Computing Services on behalf of the BNC Consortium. URL: http://www.natcorp.ox.ac.uk/
7. Chalmers D. J., French R. M., Hofstadter D. R.: High-level perception, representation and analogy: a critique of artificial intelligence methodology, Journal of experimental and theoretical artificial intelligence, 4(3), (1992)
8. Chiang D.: A Hierarchical Phrase-Based Model for Statistical Machine Translation, ACL (2005)
9. Crestani F.: Application of spreading activation techniques in information retrieval, Artificial Intelligence Review 11, 453 – 482 (1997)
10. Curran J.R., Clark S., and Bos J.: Linguistically Motivated Large-Scale NLP with C&C and Boxer. Proceedings of the ACL 2007 Demonstrations Session (ACL-07 demo), 33-36 (2007)
11. Erk, K., Padó S.: A structured vector space model for word meaning in context. In Proceedings of EMNLP, pages 897906, (2008)
12. Falkenhainer B., Forbus K. D., Gentner D.: The structure-mapping engine: Algorithm and examples, Artificial Intelligence, 41, 1–63 (1989)
13. Fellbaum C.: WordNet: An Electronic Lexical Database. Cambridge, MA: MIT Press (1998)
14. Ferraresi A., Zanchetta E., Baroni M., Bernardini S.: Introducing and evaluating ukWaC, a very large web-derived corpus of English, Proceedings of LREC 2008, (2008)
15. Gabrilovich E., Markovitch S.: Computing Semantic Relatedness using Wikipedia-based Explicit Semantic Analysis, Proceedings of The 20th International Joint Conference on Artificial Intelligence (IJCAI), (2007)

16. Gabrilovich E., Markovitch S.: Overcoming the Brittleness Bottleneck using Wikipedia: Enhancing Text Categorization with Encyclopedic Knowledge Proceedings of The 21st National Conference on Artificial Intelligence (AAAI), (2006)
17. Gentner D., Wolff P.: Metaphor and Knowledge Change, in Dietrich E. and Markman A. (Eds.), Cognitive dynamics: Conceptual change in humans and machines, Mahwah, NJ: Lawrence Erlbaum Associates, 295–342, (2000)
18. Gentner D.: Structure-mapping: A theoretical framework for analogy. Cognitive Science, 7 (2), (1983),155–170.
19. Gick, M. L., Holyoak K. J.: Schema induction and analogical transfer, Cognitive Psychology, 15, 1–38 (1983)
20. Geurts, Bart and Beaver, David I., "Discourse Representation Theory", The Stanford Encyclopedia of Philosophy (Fall 2011 Edition), Edward N. Zalta (ed.)
21. Gouws S., van Rooyen G-J, Engelbrecht H.A., Measuring conceptual similarity by spreading activation over Wikipedia's hyperlink structure, in Proceedings of the 2nd Workshop on "Collaboratively constructed semantic resources", Coling 2010, 46–54 (2010)
22. Harrington B.: A semantic network approach to measuring relatedness, Coling 2010: Poster Volume, 356–364 (2010)
23. Harrington B., Clark S.: AskNet: Automated semantic knowledge network, Association for the Advancement of Artificial Intelligence, (2007)
24. Hofstadter D. R.: Epilogue: Analogy as the core of cognition, The Analogical Mind: perspectives from cognitive science, Gentner D., Holyoak K. J., Kokinov, B. N. (eds), 15, MIT Press (2001)
25. Holyoak K. J.: Analogy, The Cambridge Handbook of Thinking and Reasoning, Holyoak, K. J., Morrison, G. (eds), 6, Cambridge University Press (2005)
26. Kamp, H.: A theory of truth and semantic representation, in Groenendijk J.A.G., Janssen T.M.V., and Stokhof M.B.J. (eds.), Formal Methods in the Study of Language. Mathematical Centre Tracts 135, Amsterdam. 277–322 (1981)
27. Lenci A.: Distributional semantics in linguistic and cognitive research. A foreword. Rivista di linguistica 20.1, 1–30 (2008)
28. Landauer T. K., Dumais S.: A Solution to Plato's Problem: The Latent Semantic Analysis Theory of Acquisition, Induction, and Representation of Knowledge, Psychological Review (1997)
29. Manning C., Raghavan P., and Schtze H.: Introduction to Information Retrieval, Cambridge University Press (2008)
30. Mihalcea, R., Radev, D.: Graph-based Natural Language Processing and Information Retrieval, Cambridge University Press (2011)
31. Miller, G. A. WordNet: A Lexical Database for English. Communications of the ACM Vol. 38, No. 11: 39-41 (1995)

32. Mitchell J., Lapata M.: Composition in distributional models of semantics, Cognitive Science 34, 1388–1429 (2010)
33. Ng A., Zheng A. Z., Jordan M.I.: Link Analysis, Eigenvectors and Stability, Seventeenth International Joint Conference on Artificial Intelligence, (2001)
34. Page L., Brin S.: The anatomy of a large-scale hypertextual Web search engine, Proceedings of the seventh international conference on World Wide Web (1998)
35. Rumelhart D. E., Abrahamson A.: A model for analogical reasoning, Cognitive Psychology 5, 1–28 (1973)
36. Turney D. P.: Similarity of Semantic Relations, Computational Linguistics, 32 (3), 379–416, (2006)
37. Turney, P.D.: The latent relation mapping engine: Algorithm and experiments, Journal of Artificial Intelligence Research (JAIR), 33, 615-655, (2008)
38. Turney P.D., and Littman M.L., Corpus-based learning of analogies and semantic relations, Machine Learning, 60 (1-3), 251-278 (2005)
39. Turney P.D., Pantel P.: From Frequency to Meaning: Vector Space Models of Semantics, Journal of Artificial Intelligence Research 37, 141-181 (2010)
40. Wilson M.: Wandering Significance, Oxford University Press, (2006)
41. Wisniewski E. J.: Construal and similarity in conceptual combination, Journal of Memory and Language 35, 434-453 (1996)
42. Wojtinnek P., Harrington B., Rudolph S., Pulman S.: Conceptual knowledge acquisition using automatically generated large-scale semantic networks, CS-RR-10-04, Oxford University Computing Laboratory, (2010)
43. Yeh E., Ramage D., Manning C. D., Agirre E., Soroa A.: WikiWalk: Random walks on Wikipedia for Semantic Relatedness, in Proceedings of the 2009 Workshop on Graph-based Methods for Natural Language Processing (TextGraphs-4), (2009)