Compositional Distributional Semantics with Compact Closed Categories and Frobenius Algebras

Dimitrios Kartsaklis
Wolfson College
University of Oxford

A thesis submitted for the degree of
Doctor of Philosophy
Michaelmas 2014

To my mother, Chryssa, who fought all her life for her children
to work “some place with a roof over their heads”

To my father, Euripides, who taught me to love technology
and passed me down the genes to work efficiently with it

Acknowledgements

This thesis would not be possible without the guidance and support of my immediate supervisor Mehrnoosh Sadrzadeh. I am grateful to her for giving me the opportunity to work in one of the top universities in the world; for introducing me to this new perspective of NLP, full of objects and arrows; for her patience and guidance; for treating me not as a student, but as a collaborator; above all, though, I thank her for being to me a true friend.

Just being around to people like Bob Coecke and Stephen Pulman (my other two supervisors) is sufficient reason for anyone to quit his job and move 3,500 kilometers from home to a foreign country. You have so many things to learn from them, but what struck me the most was this simple fact of life: the *really* important people don't have to try hard to prove who they are. I am grateful to both of them for their invaluable help during the course of this project.

I owe a big and heartfelt "thank you" to Samson Abramsky, Phil Blunsom, Steve Clark, Mirella Lapata, Anne Preller and Matthew Purver for all the advice, suggestions and comments they kindly offered to me. I would also like to thank my colleagues in Oxford Nal Kalchbrenner, Pengyu Wang and Ed Grefenstette. It would be an obvious omission not to mention my good friend Andreas Chatzistergiou who, although in Edinburgh, kept me company all this time via Google chat; I'm really grateful for his friendship and support.

Last, I would like to thank my family for their unconditional support to my decision to become a researcher. If there is any reason at all to make me regret the remarkable experience I lived in Oxford, this would be that my place during these difficult times was there with them, offering my help by any means I could use.

This research was supported by EPSRC grant EP/F042728/1.

Abstract

The provision of compositionality in distributional models of meaning, where a word is represented as a vector of co-occurrence counts with every other word in the vocabulary, offers a solution to the fact that no text corpus, regardless of its size, is capable of providing reliable co-occurrence statistics for anything but very short text constituents. The purpose of a *compositional distributional model* is to provide a function that composes the vectors for the words within a sentence, in order to create a vectorial representation that reflects its meaning. Using the abstract mathematical framework of category theory, Coecke, Sadrzadeh and Clark showed that this function can directly depend on the grammatical structure of the sentence, providing an elegant mathematical counterpart of the formal semantics view. The framework is general and compositional but stays abstract to a large extent.

This thesis contributes to ongoing research related to the above categorical model in three ways: Firstly, I propose a concrete instantiation of the abstract framework based on Frobenius algebras (joint work with Sadrzadeh). The theory improves shortcomings of previous proposals, extends the coverage of the language, and is supported by experimental work that improves existing results. The proposed framework describes a new class of compositional models that find intuitive interpretations for a number of linguistic phenomena.

Secondly, I propose and evaluate in practice a new compositional methodology which explicitly deals with the different levels of lexical ambiguity (joint work with Pulman). A concrete algorithm is presented, based on the separation of vector disambiguation from composition in an explicit prior step. Extensive experimental work shows that the proposed methodology indeed results in more accurate composite representations for the framework of Coecke et al. in particular and every other class of compositional models in general.

As a last contribution, I formalize the explicit treatment of lexical ambiguity in the context of the categorical framework by resorting to categorical quantum mechanics (joint work with Coecke). In the proposed extension, the concept of a distributional vector is replaced with that of a density matrix, which compactly represents a probability distribution over the potential different meanings of the specific word. Composition takes the form of quantum measurements, leading to interesting analogies between quantum physics and linguistics.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

> "When *I* use a word," Humpty Dumpty said, in rather a scornful tone, "it means just what I choose it to mean—neither more nor less."
>
> Lewis Carroll, *Through the Looking-Glass* (1871)

Language serves to convey *meaning*. Above anything else, therefore, the ultimate and long-standing goal of every computational linguist is to devise an efficient way to represent this meaning as faithfully as possible within a computer's memory. Every manifestation of what we call natural language processing—sentiment analysis, machine translation, paraphrase detection, to name just a few—culminates in some form of this endeavour. Not a trivial task, to say the least, made harder to achieve by the fact that language is dominated by ambiguity. The means of verbal or written communication—words combined in phrases and sentences according to the rules of some grammar—are rarely monosemous; almost every word has more than one dictionary definition, while in many cases the true meaning of a specific utterance can be intimately related to subjective factors such as beliefs, education, or social environment of the sayer. In order to cope with these difficulties, computational linguists employ a large arsenal of clever techniques, but, first and foremost, they have to provide an answer for a question much more philosophical than technical: *what is meaning* after all?

At the sentence level, the traditional way of approaching this important problem is a syntax-driven compositional method that follows Montague's tradition: every word in the sentence is associated with a primitive symbol or a predicate, and these are combined into larger and larger logical forms based on the syntactical rules of the grammar. At the end of the syntactical analysis, the logical representation of the whole sentence is a complex formula that can be fed to a theorem prover for

further processing. Although such an approach seems intuitive, it has two important drawbacks: First, the resulting composite formula can only state whether the sentence in question is true or false, based on the initial values of the constituents therein; second, it captures the meaning of the atomic units (words) in an axiomatic way, namely by ad-hoc unexplained primitives that have nothing to say about the real semantic value of the specific words.

On the other hand, distributional models of meaning work by building co-occurrence *vectors* for every word in a corpus based on its context, following Firth's famous intuition that "you should know a word by the company it keeps" [34]. These models have been proved useful in many natural language tasks [29, 61, 69, 107] and can provide concrete information for the words of a sentence, but they do not naturally scale up to larger constituents of text, such as phrases or sentences. The reason behind this is that a sentence or phrase is something much more unique than a word; as a result, there is currently no text corpus available that can provide reliable co-occurrence statistics for anything larger than two- or three-word text segments.

Given the complementary nature of these two distinct approaches, it is not a surprise that compositional abilities of distributional models have been the subject of much discussion and research in recent years. Towards this purpose researchers exploit a wide variety of techniques, ranging from simple mathematical operations like vector addition and multiplication to neural networks and even category theory. No matter what means is utilized, the goal is always the same: although it is not possible to directly construct a sentence vector, we can still *compose* the vectors for the words within the sentence in question, providing a vectorial representation that approximates some idealistic behaviour from a semantic perspective. In other words, the purpose of a *compositional distributional model* (CDM) of meaning is, given a sentence $w_1 w_2 \dots w_n$, to provide a function f as follows:

$$\overrightarrow{s} = f(\overrightarrow{w_1}, \overrightarrow{w_2}, \dots, \overrightarrow{w_n}) \tag{1.1}$$

where $\overrightarrow{w_i}$ refers to the vector of the ith word and $\overrightarrow{s}$ is a vector serving as a semantic representation for the sentence.

The subject of this thesis is CDMs in general, and a specific class of models based on category theory in particular. Using the abstract setting of compact closed categories, Coecke, Sadrzadeh and Clark [25] showed that under certain conditions grammar and vector spaces are structurally homomorphic; in other words, a grammatical derivation can be translated to some algebraic manipulation between vector spaces, which provides the desired passage that Eq. 1.1 aims to achieve. The cat-

egorical framework provides a theoretical justification for a certain class of CDMs that is now known as *tensor-based* models, the main characteristic of which is that they are *multi-linear* in nature. In a tensor-based model, relational words such as verbs and adjectives are functions represented by multi-linear maps (*tensors* of various orders) that act on their arguments (usually vectors representing nouns) through a process known as tensor contraction. In this sense, the framework can be seen as an elegant mathematical counterpart of the formal semantics perspective, as this was expressed by Montague [77], Lambek [60] and other pioneers of the formal approaches to language.

To this date, the categorical structures of [25] remain largely abstract. The only extensive study on the subject comes from Grefenstette [38], who provided valuable insights regarding the construction of tensors for transitive verbs, as well as a first implementation for simple text constructs. In many aspects, the present work is a direct extension of that original contribution. It essentially starts where Grefenstette's work stops, having as its main purpose to help realizing the abstract categorical structures in terms of concrete instantiations and make them applicable to mainstream natural language processing tasks.

The first contribution of this work is that it provides a robust and scalable framework for instantiating the tensors of relational and functional words based on the application of Frobenius algebras which, as shown in [24], offer a canonical way for uniformly copying or deleting the basis of a vector space. This proposal provides a way to overcome certain shortcomings of the earlier implementation, such as the fact that sentences with nested grammatical structures could not be assigned a meaning. Furthermore, we will see how the Frobenius operators can be used for modelling a number of functional words (for example prepositions, conjunctions and so on) which until now were considered as semantically vacuous and ignored by most CDMs. The introduction of Frobenius algebras in language results in a new class of CDMs, in which features from both tensor-based models and element-wise compositional models relying on vector multiplication are combined into a unified framework. As will become evident, this unique characteristic finds intuitive interpretations for a number of linguistic phenomena, such as intonation and coordination. The Frobenius framework builds on collaboration with Mehrnoosh Sadrzadeh and our published work in [51, 53].

An equally important contribution of the current thesis is that it presents the first large-scale study to date regarding the behaviour of CDMs under the presence of lexical ambiguity. On the theory side, I discuss the different levels of ambiguity in re-

lation to compositionality, and make certain connections with psycholinguistic models that attempt to describe how the human brain behaves in similar settings. One of the main arguments of this thesis is that in general a CDM cannot handle every degree of ambiguity in the same way; a certain distinction must be made between cases of homonymy (genuinely ambiguous words with two or more unrelated meanings) and polysemy (small deviations between the senses of a word), which can be realized by the introduction of an explicit disambiguation step on the word vectors of a sentence before the actual composition. The validity of the proposed methodology is verified by extensive experimental work, which provides solid evidence that the separation of disambiguation from composition in two distinct steps leads to more accurate composite representations. Furthermore, as we will see, the applicability of this method is not restricted to tensor-based models, but it has very positive effects in every class of CDMs. The prior disambiguation hypothesis originated and took shape mainly from discussions and collaboration with Stephen Pulman [52]; the experimental evidence was presented in various subsequent publications [48, 47, 19].

The fact that lexical ambiguity (read: uncertainty regarding the true meaning under which a word is used in a specific context) dominates language to a such a great extent implies that any semantic model should have some means of incorporating this notion. Interestingly, the framework of compact closed categories that comprises the basis of the categorical model has been used in the past by Abramsky and Coecke [1] for reasoning in a different field with vector space semantics, that of quantum mechanics. The relation between quantum mechanics and linguistics implied by this connection becomes concrete in Chapter 6, where the informal discussion about composition and lexical ambiguity takes shape in quantum-theoretic terms. Specifically, the categorical model of [25] is advanced such that the notion of a state tensor is replaced by that of a density matrix, an elegant compact way to represent a probability distribution. The new model is capable of handling the different levels of ambiguity imposed by homonymy and polysemy in a unified and linguistically motivated way, offering overall a richer semantic representation for language. The formalization of lexical ambiguity in categorical quantum mechanics terms was collaborative work with Bob Coecke [90].

The dissertation is structured in the following way:

Part I aims at providing the necessary background to the reader. Chapter 2 starts with a concise introduction to compositional and distributional models of meaning. Furthermore, the most important attempts at unifying the two models are briefly

presented, and a hierarchy of CDMs is devised according to their theoretical power and various other characteristics. Chapter 3 explains in some detail the categorical compositional framework of [25] which is the main subject of this work. The reader is introduced to the notions of compact closed categories, pregroup grammars, and Frobenius algebras; furthermore, I present the convenient graphical language of monoidal categories, which will serve as one of my main tools for expressing the ideas in this thesis.

Part II presents the main theoretical contributions of the current thesis. Chapter 4 introduces Frobenius algebras in language, discusses the theoretical intuition behind it, and examines the effect of the resulting model in relation to a number of linguistic phenomena. Chapter 5 extends this discussion to functional words such as prepositions, conjunctions, and infinitive particles. Chapter 6 deals with the vast topic of lexical ambiguity in relation to composition. After an informal discussion that explains the linguistic motivation, section 6.3 proceeds to incorporate ambiguity in the framework of [25] in terms of categorical quantum mechanics and density matrices.

Part III is dedicated to practical aspects of this research and experimental work. Chapter 7 provides a first evaluation of the Frobenius models in three tasks involving head verb disambiguation, verb phrase similarity, and term/definition classification with promising results. The purpose of Chapter 8 is two-fold: first, a concrete methodology for implementing the prior disambiguation step for relational tensors is detailed and evaluated; second, the prior disambiguation hypothesis is extensively tested and verified for every class of CDMs.

Finally, Chapter 9 summarizes the main contributions of this thesis, and briefly discusses the (many) opportunities for future research on the topic of tensor-based compositional models.

Related published work

The material I am going to present in this thesis builds on a large number of publications that took place in the course of three years, and inevitably some passages (especially describing background material in Part I or experimental settings in Part III) have been re-used verbatim from these original sources. The publications that are relevant to a specific chapter are cited in the abstract of that chapter. Furthermore, I provide here a list of the related published work in chronological order:

1. Kartsaklis, Sadrzadeh, and Pulman. A Unified Sentence Space for Categorical Distributional Compositional Semantics: Theory and Experiments. In *Proceed-*

ings of 24th International Conference on Computational Linguistics (COLING 2012): Posters, Mumbai, India, December 2012.

2. Kartsaklis, Sadrzadeh, Pulman, and Coecke. Reasoning about Meaning in Natural Language with Compact Closed Categories and Frobenius Algebras. In J. Chubb, A. Eskandarian, and V. Harizanov, editors, *Logic and Algebraic Structures in Quantum Computing and Information*, Association for Symbolic Logic Lecture Notes in Logic. Cambridge University Press, 2015.

3. Kartsaklis, Sadrzadeh, and Pulman. Separating Disambiguation from Composition in Distributional Semantics. In *Proceedings of 17th Conference on Computational Natural Language Learning (CoNLL-2013)*, Sofia, Bulgaria, August 2013.

4. Kartsaklis and Sadrzadeh. Prior Disambiguation of Word Tensors for Constructing Sentence Vectors. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Seattle, USA, October 2013.

5. Kartsaklis. Compositional Operators in Distributional Semantics. *Springer Science Reviews*, 2(1-2):161-177, 2014.

6. Kartsaklis, Kalchbrenner, and Sadrzadeh. Resolving Lexical Ambiguity in Tensor Regression Models of Meaning. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL): Short Papers*, Baltimore, USA, June 2014.

7. Kartsaklis and Sadrzadeh. A Study of Entanglement in a Categorical Framework of Natural Language. In B. Coecke, I.Hasuo and P. Panangaden, editors, *Proceedings of the 11th Workshop of Quantum Physics and Logic (QPL)*, EPTCS 172, Kyoto, Japan, June 2014.

8. Milajevs, Kartsaklis, Sadrzadeh and Purver. Evaluating Neural Word Representations in Tensor-Based Compositional Settings. In *Proceedings of the 2014 Conference on Empirical Methods in Natural language Processing (EMNLP)*, Doha, Qatar, October 2014.

9. Cheng, Kartsaklis and Grefenstette. Investigating the Effect of Prior Disambiguation in Deep-Learning Compositional Models of Meaning. *Learning Semantics* workshop, NIPS 2014, Montreal, Canada, December 2014.

Part I

Background

Chapter 2

Compositionality in Distributional Models of Meaning

Chapter Abstract

In this chapter the reader is introduced to the two prominent semantic paradigms of natural language: compositional semantics and distributional models of meaning. Then I proceed to the central topic of this thesis, CDMs, presenting a survey of the most important attempts to date for providing compositionality in distributional models. I conclude by putting together a taxonomy of CDMs based on composition function and various other characteristics. Material is based on my review article in [46].

2.1 Compositionality in language

Compositionality in semantics offers an elegant way to address the inherent property of natural language to produce an infinite number of structures (phrases and sentences) from finite resources (words). The *principle of compositionality* states that the meaning of a complex expression can be determined by the meanings of its constituents and the rules used for combining them. This idea is quite old, and glimpses of it can be spotted even in works of Plato. In his dialogue *Sophist*, Plato argues that a sentence consists of a noun and a verb, and that the sentence is true if the verb denotes the action that the noun is currently performing. In other words, Plato argues that (a) a sentence has a structure; (b) the parts of the sentence have different functions; (c) the meaning of the sentence is determined by the function of its parts. Nowadays, this intuitive idea is often attributed to Gottlob Frege, although it was

never explicitly stated by him in some of his published works. In an undated letter to Philip Jourdain, though, included in "Philosophical and Mathematical Correspondence" [36], Frege justifies the reasoning behind compositionality by saying:

> The possibility of our understanding propositions which we have never heard before rests evidently on this, that we can construct the sense of a proposition out of parts that correspond to words.

This forms the basis of the *productivity* argument, often used as a proof for the validity of the principle: humans only know the meaning of words, and the rules to combine them in larger constructs; yet, being equipped with this knowledge, we are able to produce new sentences that we have never uttered or heard before. Indeed, this task seems natural even for a 3-year old child—however, its formalization in a way reproducible by a computer has been proven anything but trivial. The modern compositional models owe a lot to the seminal work of Richard Montague (1930-1971), who managed to present a systematic way of processing fragments of the English language in order to get semantic representations capturing their "meaning" [77, 78, 79].

In "Universal Grammar" [78], Montague starts detailing a systematization of the natural language, an approach that became known as Montague grammar. To use Montague's method, one would need two things: first, a resource which will provide the logical forms of each specific word (a lexicon); and second, a way to determine the correct order in which the elements in the sentence should be combined in order to end up with a valid semantic representation. A natural way to address the latter is to use the syntactic structure as a means of driving the semantic derivation (an approach called *syntax-driven semantic analysis*). In other words, we assume that there exists a mapping from syntactic to semantic types, and that the composition at the syntax level implies a similar composition at the semantic level. This is known as the *rule-to-rule hypothesis* [3].

In order to provide an example, I will use the sentence 'Every man walks'. I begin from the lexicon, the job of which is to assign a grammar type and a logical form to every word in the sentence:

(1) a. every $\vdash Dt : \lambda P.\lambda Q.\forall x[P(x) \rightarrow Q(x)]$
    b. man $\vdash N : \lambda y.man(y)$
    c. walks $\vdash V_I : \lambda z.walk(z)$

The above use of formal logic (especially higher-order) in conjunction with λ-calculus was first introduced by Montague, and from then on it constitutes the stan-

dard way of providing logical forms to compositional models. In the above lexicon, predicates of the form $man(y)$ and $walk(z)$ are true if the individuals denoted by y and z carry the property (or, respectively, perform the action) indicated by the predicate. From an extensional perspective, the semantic value of a predicate can be seen as the set of all individuals that carry a specific property. Hence, given a universe of elements $\mathcal{U}$ we define the semantic value of *walk* as follows:

$$[\![walk]\!] = \{x | x \in \mathcal{U} \wedge x \text{ walks}\} \tag{2.1}$$

The expression $walk(john)$ will be true if (and only if) the individual denoted by $john$ belongs to the set of all individuals who perform the action of walking; that is, iff $[\![john]\!] \in [\![walk]\!]$. Furthermore, λ-terms like λx or λQ have the role of placeholders that remain to be filled. The logical form $\lambda y.man(y)$, for example, reflects the fact that the entity which is going to be tested for the property of manhood is still unknown and it will be later specified based on the syntactic combinatorics. Finally, the form in (1a) reflects the traditional way for representing a universal quantifier in natural language, where the still unknown parts are actually the predicates acting over a range of entities.

In λ-calculus, function application is achieved via the process of β-reduction: given two logical forms $\lambda x.t$ and s, the application of the former to the latter will produce a version of t where all the free occurrences of x in t have been replaced by s. More formally:

$$(\lambda x.t)s \rightarrow_\beta t[x := s] \tag{2.2}$$

Let us see how we can use the principle of compositionality to get a logical form for the above example sentence, by applying β-reduction between the semantic forms of text constituents following the grammar rules. Fig. 2.1 provides us a syntactic analysis and the corresponding grammar rules augmented with semantic applications.

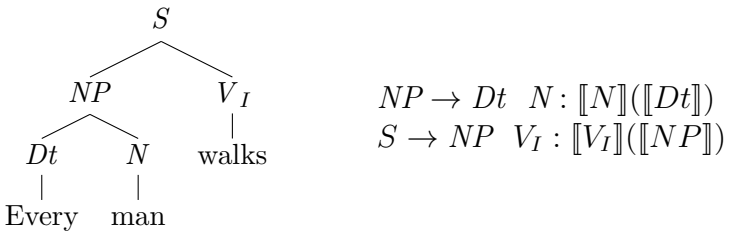


Figure 2.1: Syntax and semantics correspondence.

The rules on the right part of Fig. 2.1 will drive the semantic derivation, the full form of which can be found in Fig. 2.2. At the top of the tree you can see the logical form that is produced by the β-reduction process and serves as a semantic representation for the whole sentence.

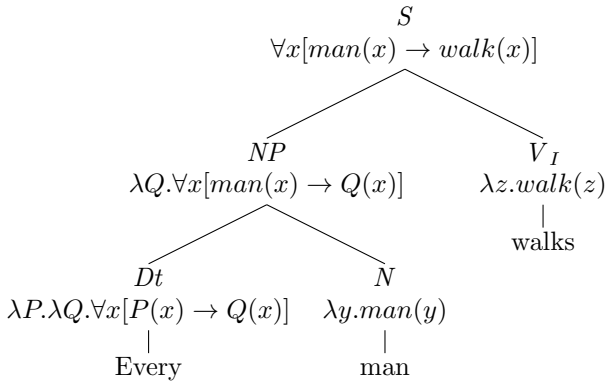


Figure 2.2: A semantic derivation for a simple sentence.

A logical form such as $\forall x[man(x) \rightarrow walk(x)]$ simply states the truth (or falsity) of the expression given the sub-expressions and the rules for composing them. It does not provide any quantitative interpretation of the result (e.g. grades of truth) and, more importantly, leaves the meaning of words as unexplained primitives (*man*, *walk* etc). In the next section I will show how distributional semantics can fill this gap.

2.2 Distributional models of meaning

2.2.1 Meaning is context

The distributional paradigm is based on the *distributional hypothesis* (Harris [42]), stating that words that occur in the same context have similar meanings. Various forms of this popular idea keep recurring in the literature: Firth calls it collocation [34], while Frege himself states that "never ask for the meaning of a word in isolation, but only in the context of a proposition"[1] [35]. The attraction of this principle in the context of Computational Linguistics is that it provides a way of

[1]Interestingly, a literal interpretation of this statement leads to the conclusion that the meaning of an isolated word cannot be defined. This idea, a more generic form of which is referred to as *principle of contextuality*, contradicts compositionality, which requires the assignment of meaning to each part of a whole. The question regarding which of the two principles Frege actually embraced and to what extent has not yet found a satisfactory answer, as the study of [87] shows.

concretely representing the meaning of a word via mathematics: each word is a vector whose elements show how many times this word occurred in some corpus in the same context with every other word in the vocabulary. If, for example, our basis is $\{cute, sleep, finance, milk\}$, the vector for word 'cat' could have the form $(15, 7, 0, 22)$ meaning that 'cat' appeared 15 times together with 'cute', 7 times with 'sleep' and so on. More formally, given an orthonormal basis $\{\overrightarrow{n_i}\}_i$ for our vector space, a word is represented as:

$$\overrightarrow{word} = \sum_i c_i \overrightarrow{n_i} \tag{2.3}$$

where c_i is the coefficient for the ith basis vector. As mentioned above, in their simplest form these coefficients can be just co-occurrence counts, although in practice a function on raw counts is often used in order to remove some of the unavoidable frequency bias. A well-known measure is the information-theoretic *point-wise mutual information* (PMI), which can reflect the relationship between a context word c and a target word t as follows:

$$\text{PMI}(c, t) = \log \frac{p(c|t)}{p(c)} \tag{2.4}$$

In contrast to compositional semantics which leaves the meaning of lexical items unexplained, a vector like the one used above for 'cat' provides some concrete information about the meaning of the specific word: cats are cute, sleep a lot, they really like milk, and they have nothing to do with finance. Additionally, this quantitative representation allows us to compare the meanings of two words, e.g. by computing the cosine distance of their vectors, and evaluate their semantic similarity. The cosine distance is a popular choice for this task (but not the only one) and is given by the following formula:

$$\text{sim}(\overrightarrow{v}, \overrightarrow{u}) = \cos(\overrightarrow{v}, \overrightarrow{u}) = \frac{\langle \overrightarrow{v} | \overrightarrow{u} \rangle}{\| \overrightarrow{v} \| \| \overrightarrow{u} \|} \tag{2.5}$$

where $\langle \overrightarrow{v} | \overrightarrow{u} \rangle$ denotes the dot product between $\overrightarrow{v}$ and $\overrightarrow{u}$, and $\| \overrightarrow{v} \|$ the magnitude of $\overrightarrow{v}$. As an example, in the 2-dimensional vector space of Fig. 2.3 we see that 'cat' and 'puppy' are close together (and both of them closer to the basis 'cute'), while 'bank' is closer to basis vector 'finance'.

A more realistic example is shown in Fig. 2.4. The points in this space represent real distributional word vectors created from the British National Corpus

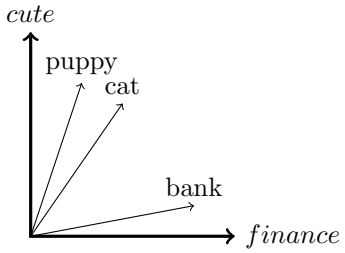


Figure 2.3: A toy "vector space" for demonstrating semantic similarity between words.

(BNC)[2], originally 2,000-dimensional and projected onto two dimensions for visualization. Note how words form distinct groups of points according to their semantic correlation. Furthermore, it is interesting to see how ambiguous words behave in these models: the ambiguous word 'mouse' (with the two meanings to be that of a rodent and of a computer pointing device), for example, is placed almost equidistantly from the group related to IT concepts (lower left part of the diagram) and the animal group (top left part of the diagram), having a meaning that can be indeed seen as the average of both senses.

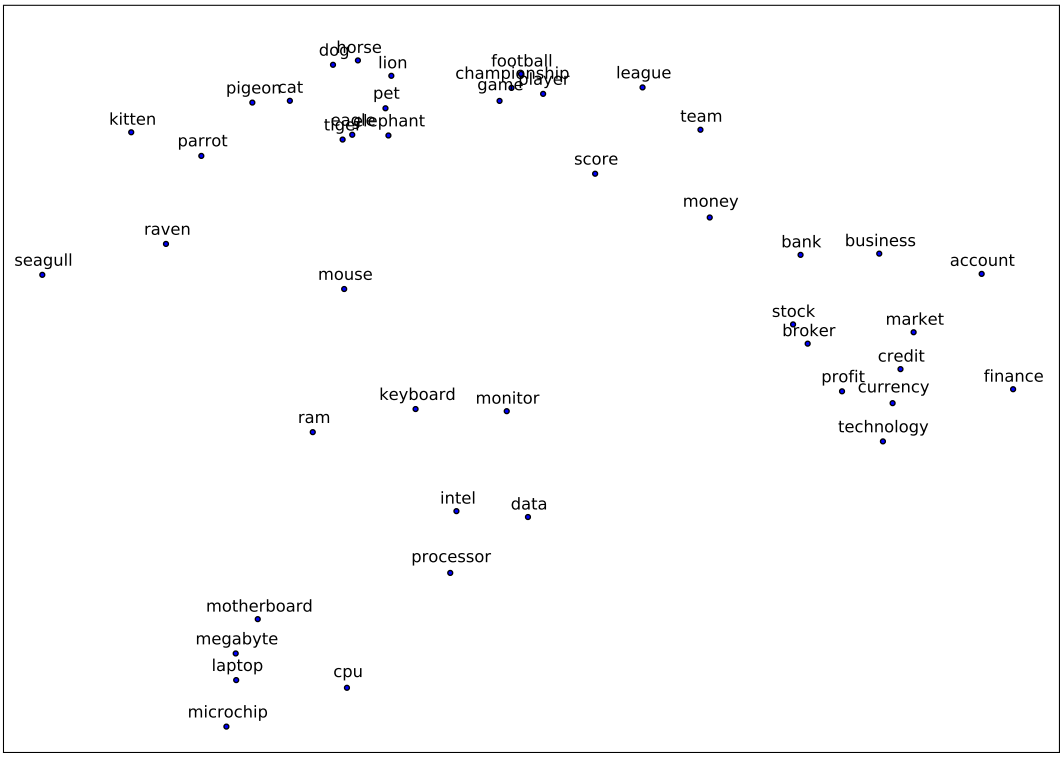


Figure 2.4: Visualization of a word space in two dimensions (original vectors are 2000-dimensional vectors created from BNC).

[2]The BNC is a 100 million-word text corpus consisting of samples of written and spoken English. It can be found online at `http://www.natcorp.ox.ac.uk/`.

2.2.2 Forms of word spaces

In the simplest form of a word space (a vector space for words), the context of a word is a set containing all the words that occur within a certain distance from the target word, for example within a 5-word window. Models like these have been extensively studied and implemented the past years, see for example [65, 66]. However, although such an approach is simple, intuitive and computationally efficient, it is not optimal, since it assumes that every word within this window will be semantically relevant to the target word, while treating every word outside of the window as irrelevant. Unfortunately, this is not always the case. Consider for example the following phrase:

(2) The movie I saw and John said he really likes

Here, a word-based model cannot efficiently capture the long-range dependency between 'movie' and 'likes'; on the other hand, since 'said' is closer to 'movie' it is more likely to be considered as part of its context, despite the fact that the semantic relationship between the two words is actually weak. Cases like the above suggest that a better approach for the construction of the model would be to take into account not just the surface form of context words, but also the specific grammatical relations that hold between them and the target word. A model like this, for example, will be aware that 'movie' is the object of 'likes', so it could safely consider the latter as part of the context for the former, and vice versa. This kind of observation motivated many researchers to experiment with vector spaces based not solely on the surface forms of words but also on various syntactical properties of the text.

One of the earliest attempts to add syntactical information in a word space was that of Grefenstette [41], who used a structured vector space with a basis constructed by grammatical properties such as 'subject-of-buy' or 'argument-of-useful', denoting that the target word has occurred in the corpus as subject of the verb 'buy' or as argument of the adjective 'useful'. The weights of a word vector were binary, either 1 for at least one occurrence or 0 otherwise. Lin [64] moves one step further, replacing the binary weights with frequency counts. Following a different path, Erk and Padó [32] argue that a single vector is not enough to catch the meaning of a word; instead, the vector of a word is accompanied by a set of vectors R representing the lexical preferences of the word for its arguments positions, and a set of vectors R^{-1}, denoting the inverse relationship, that is, the usage of the word as argument in the lexical preferences of other words. Subsequent works by Thater et al. [116, 117] present a version of this idea extended with the inclusion of grammatical dependency contexts. Finally, in an attempt to provide a generic distributional framework, Padó

and Lapata [84] presented a model based on dependency relations between words. The interesting part of this work is that, given the proper parametrization, it is indeed able to essentially subsume a large amount of other works on the same subject.

2.2.3 Neural word embeddings

Recently, a new class of distributional models in which the training of word representations is performed by some form of neural network architecture has received a lot of attention, showing state-of-the-art performances in many natural language processing tasks. *Neural word embeddings* [9, 28, 71] interpret the distributional hypothesis differently: instead of relying on observed co-occurrence frequencies, a neural model is trained to maximise some probabilistic objective function; in the skip-gram model of Mikolov et al. [72], for example, this function is related to the probability of observing the surrounding words in some context, and its objective is to maximize the following quantity:

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0}\log p(w_{t+j}|w_t) \tag{2.6}$$

Specifically, the function maximises the conditional probability of observing words in a context around the target word w_t, where c is the size of the training context, and $w_1 w_2 \cdots w_T$ is a sequence of training words. A model like this therefore captures the distributional intuition and can express degrees of lexical similarity. However, it works differently from a typical co-occurrence distributional model in the sense that, instead of simply counting the context words, it actually *predicts* them. Compared to the co-occurrence method, this has the obvious advantage that the model in principle can be much more robust to data sparsity problems, which is always an important issue for traditional word spaces. Indeed, a number of studies [8, 63, 73] suggest that neural vectors reflect better than their "counting" counterparts the semantic relationships between the various words.

2.2.4 A philosophical digression

To what extent is the distributional hypothesis correct? Even if we accept that a co-occurrence vector can indeed capture the "meaning" of a word, it would be far too simplistic to assume that this holds for *every* kind of word. The meaning of some words can be determined by their denotations; one could claim, for example, under

a Platonic view of semantics, that the meaning of the word 'tree' is the set of all trees, and we are even able to answer the question "what is a tree?" by pointing to a member of this set, a technique known as *ostensive* definition. But this is not true for all words. In "Philosophy" (published in "Philosophical Occasions: 1912-1951", [122]), Ludwig Wittgenstein notes that there exist certain words, like 'time', the meaning of which is quite clear to us until the moment we have to explain it to someone else; then we realize that suddenly we are not able any more to express in words what we certainly know—it is like we have forgotten what that specific word really means. Wittgenstein claims that "if we have this experience, then we have arrived at the limits of language". This observation is related to one of the central ideas of his work: that the meaning of a word does not need to rely on some kind of definition; what really matters is the way we use this word in our everyday communications. In "Philosophical Investigations" ([123]), Wittgenstein presents a thought experiment:

> Now think of the following use of language: I send someone shopping. I give him a slip marked 'five red apples'. He takes the slip to the shopkeeper, who opens the drawer marked 'apples'; then he looks up the word 'red' in a table and finds a colour sample opposite it; then he says the series of cardinal numbers—I assume that he knows them by heart—up to the word 'five' and for each number he takes an apple of the same colour as the sample out of the drawer. It is in this and similar ways that one operates with words.

For the shopkeeper, the meaning of words 'red' and 'apples' was given by ostensive definitions (provided by the colour table and the drawer label). But what was the meaning of word 'five'? Wittgenstein is very direct on this:

> No such thing was in question here, only how the word 'five' is used.

Note that this operational view of the "meaning is use" idea is quite different from the distributional perspective we discussed in the previous sections. It simply implies that language is not expressive enough to describe certain fundamental concepts of our world which, in turn, means that the application of distributional models is by definition limited to a subset of the vocabulary for any language. While this limitation is disappointing, it certainly does not invalidate the established status of this technology as a very useful tool in natural language processing; however, knowing the fundamental weaknesses of our models is imperative since it helps us to set realistic expectations and opens opportunities for further research.

2.3 Unifying the two semantic paradigms

Distributional models of meaning have been widely studied and successfully applied to a variety of language tasks, especially during the last decade with the availability of large-scale corpora, like Gigaword [37] and ukWaC [33], which provide a reliable resource for training the vector spaces. For example, Landauer and Dumais [61] use vector space models in order to reason about human learning rates in language; Schütze [107] performs word sense induction and disambiguation; Curran [29] shows how distributional models can be applied to automatic thesaurus extraction; Manning et al. [69] discuss possible applications in the context of information retrieval.

Despite the size of the various text corpora, though, the creation of vectors representing the meaning of multi-word sentences by statistical means is still impossible. Therefore, the provision of distributional models with compositional abilities similar to what was described in §2.1 seems a very appealing solution that could offer the best of both worlds in a unified manner. The goal of such a system would be to combine the distributional vectors of words into vectors of larger and larger text constituents, up to the level of a sentence. A sentence vector, then, could be compared with other sentence vectors, providing a way for assessing the semantic similarity between sentences as if they were words. The benefits of such a feature are obvious for many natural language processing tasks, such as paraphrase detection, machine translation, information retrieval, and so on, and in the following sections I will review all the important approaches and current research towards this challenging goal.

2.3.1 Vector mixture models

The transition from word vectors to sentence vectors implies the existence of a composition operation that can be applied between text constituents: the composition of 'red' and 'car' into the adjective-noun compound 'red car', for example, should produce a new vector derived from the composition of the distributional vectors for 'red' and 'car'. Since we work with vector spaces, the candidates that first come to mind are vector addition and vector (point-wise) multiplication. Indeed, Mitchell and Lapata [75] present and test various models, where the composition of vectors is based on these two simple operations. Given two word vectors $\overrightarrow{w_1}$ and $\overrightarrow{w_2}$ and assuming an orthonormal basis $\{\overrightarrow{n_i}\}_i$, the multiplicative model computes the meaning vector of the new compound as follows:

$$\overrightarrow{w_1 w_2} = \overrightarrow{w_1} \odot \overrightarrow{w_2} = \sum_i c_i^{w_1} c_i^{w_2} \overrightarrow{n_i} \tag{2.7}$$

Similarly, the meaning according to the additive model is:

$$\overrightarrow{w_1 w_2} = \alpha \overrightarrow{w_1} + \beta \overrightarrow{w_2} = \sum_i (\alpha c_i^{w_1} + \beta c_i^{w_2}) \overrightarrow{n_i} \tag{2.8}$$

where α and β are optional weights denoting the relative importance of each word. The main characteristic of these models is that all word vectors live in the same space, which means that in general there is no way to distinguish between the type-logical identities of the different words. This fact, in conjunction with the element-wise nature of the operators, makes the output vector a kind of *mixture* of the input vectors, all of which contribute (in principle) *equally* to the final result; thus, each element of the output vector can be seen as an "average" of the two corresponding elements in the input vectors. In the additive case, the components of the result are simply the cumulative scores of the input components. So in a sense the output element embraces both input elements, resembling a union of the input features. On the other hand, the multiplicative version is closer to intersection: a zero element in one of the input vectors will eliminate the corresponding feature in the output, no matter how high the other component was.

Vector mixture models constitute the simplest compositional method in distributional semantics. Despite their simplicity, though, (or because of it) these approaches have been proved very popular and useful in many NLP tasks, and they are considered hard-to-beat baselines for many of the more sophisticated models we are going to discuss next. In fact, the comparative study of Blacoe and Lapata [11] suggests something really surprising: that, for certain tasks, additive and multiplicative models can be almost as effective as state-of-the-art deep learning models, which will be the subject of §2.3.4.

2.3.2 Tensor product and circular convolution

The low complexity of vector mixtures comes with a price, since the produced composite representations disregard grammar in many different ways. For example, an obvious problem with these approaches is the commutativity of the operators: the models treat a sentence as a "bag of words" where the word order does not matter, equating for example the meaning of the sentence 'dog bites man' with that of 'man bites dog'. This fact motivated researchers to seek solutions on non-commutative op-

erations, such as the tensor product between vector spaces. Following this suggestion, which was originated by Smolensky [110], the composition of two words is achieved by a structural mixing of the basis vectors that results in an increase in dimensionality:

$$\overrightarrow{w_1} \otimes \overrightarrow{w_2} = \sum_{i,j} c_i^{w_1} c_j^{w_2} (\overrightarrow{n_i} \otimes \overrightarrow{n_j}) \tag{2.9}$$

Clark and Pulman [21] take this original idea further and propose a concrete model in which the meaning of a word is represented as the tensor product of the word's distributional vector with another vector that denotes the grammatical role of the word and comes from a different abstract vector space of grammatical relationships. As an example, the meaning of the sentence 'dog bites man' is given as:

$$\overrightarrow{dog\ bites\ man} = (\overrightarrow{dog} \otimes \overrightarrow{subj}) \otimes \overrightarrow{bites} \otimes (\overrightarrow{man} \otimes \overrightarrow{obj}) \tag{2.10}$$

Although tensor product models solve the bag-of-words problem, unfortunately they introduce a new very important issue: given that the cardinality of the vector space is d, the space complexity grows exponentially as more constituents are composed together. With $d = 300$, and assuming a typical floating-point machine representation (8 bytes per number), the vector of Equation 2.10 would require $300^5 \times 8 = 1.944 \times 10^{13}$ bytes (≈ 19.5 terabytes). Even more importantly, the use of tensor product as above only allows the comparison of sentences that share the same structure, i.e. there is no way for example to compare a transitive sentence with an intransitive one, a fact that severely limits the applicability of such models.

Using a concept from signal processing, Plate [91] suggests the replacement of tensor product by circular convolution. This operation carries the appealing property that its application on two vectors results in a vector of the same dimensions as the operands. Let $\overrightarrow{v}$ and $\overrightarrow{u}$ be vectors of d elements, the circular convolution of them will result in a vector $\overrightarrow{c}$ of the following form:

$$\overrightarrow{c} = \overrightarrow{v} \circledast \overrightarrow{u} = \sum_{i=0}^{d-1} \left(\sum_{j=0}^{d-1} v_j u_{i-j} \right) \overrightarrow{n}_{i+1} \tag{2.11}$$

where $\overrightarrow{n_i}$ represents a basis vector, and the subscripts are modulo-d, giving to the operation its circular nature. This can be seen as a compressed outer product of the two vectors. However, a successful application of this technique poses some restrictions. For example, it requires a different interpretation of the underlying vector space, in which "micro-features", as PMI weights for each context word, are replaced

by what Plate calls "macro-features", i.e. features that are expressed as whole vectors drawn from a normal distribution. Furthermore, circular convolution is commutative, re-introducing the bag-of-words problem.[3]

2.3.3 Tensor-based models

A weakness of vector mixture models and the generic tensor product approach is the symmetric way in which they treat all words, ignoring their special roles in the sentence. An adjective, for example, lives in the same space with the noun it modifies, and both will contribute equally to the output vector representing the adjective-noun compound. However, a treatment like this seems unintuitive; we tend to see relational words, such as verbs or adjectives, as functions acting on a number of arguments rather than entities of the same order as them. Following this idea, a recent line of research represents words with special meaning as multi-linear maps (tensors[4] of higher order) that apply on one or more arguments (vectors or tensors of lower order). An adjective, for example, is not any more a simple vector but a matrix (a tensor of order 2) that, when matrix-multiplied with the vector of a noun, will return a modified version of it. That is, for an adjective $\overline{adj} = \sum_{ij} c_{ij}^{adj} (\overrightarrow{n_i} \otimes \overrightarrow{n_j})$ and a noun $\overrightarrow{noun} = \sum_j c_j^{noun} \overrightarrow{n_j}$ we have:

$$\overrightarrow{adj\ noun} = \sum_{ij} c_{ij}^{adj} c_j^{noun} \overrightarrow{n_j} \tag{2.12}$$

This approach is based on the principle of *map-state duality*: every linear map from V to W (where V and W are finite-dimensional Hilbert spaces) stands in bijective correspondence to a tensor living in the tensor product space $V \otimes W$. For the case of a multi-linear map (a function with more than one argument), this can be generalized to the following:

$$f : V_1 \to \cdots \to V_j \to V_k \cong V_1 \otimes \cdots \otimes V_j \otimes V_k \tag{2.13}$$

In general, the order of the tensor is equal to the number of arguments plus one order for carrying the result; so a unary function (such as an adjective) is a tensor of order 2, while a binary function (e.g. a transitive verb) is a tensor of order 3.

[3] Actually, Plate proposes a workaround for the commutativity problem; however this is not specific to his model, since it can be used with any other commutative operation such as vector addition or point-wise multiplication.

[4] Here, the word *tensor* refers to a geometric object that can be seen as a generalization of a vector in higher dimensions. A matrix, for example, is an order-2 tensor.

The composition operation is based on the inner product and is nothing more than a generalization of matrix multiplication in higher dimensions, a process known as *tensor contraction*. Given two tensors of orders m and n, the tensor contraction operation always produces a new tensor of order $n + m - 2$. Under this setting, the meaning of a simple transitive sentence can be calculated as follows:

$$\overline{subj\ verb\ obj} = \overrightarrow{subj}^{\mathsf{T}} \times \overline{verb} \times \overrightarrow{obj} \tag{2.14}$$

where the symbol $\times$ denotes tensor contraction. Given that $\overrightarrow{subj}$ and $\overrightarrow{obj}$ live in N and $\overline{verb}$ lives in $N \otimes S \otimes N$, the above operation will result in a tensor in S, which represents the sentence space of our choice.

Tensor-based models provide an elegant solution to the problems of vector mixtures: they are not bag-of-words approaches and they respect the type-logical identities of special words, following an approach very much aligned with the formal semantics perspective. Furthermore, they do not suffer from the space complexity problems of models based on raw tensor product operations (§2.3.2), since the tensor contraction process guarantees that every sentence will eventually live in our sentence space S. On the other hand, the highly linguistic perspective they adopt has also a downside: in order for a tensor-based model to be fully effective, an appropriate linear map to vector spaces must be concretely devised for every functional word, such as prepositions, relative pronouns or logical connectives. As we will see in Chapter 5, this problem is far from trivial; actually it constitutes one of the most important open issues, and at the moment restricts the application of these models on well-defined text structures (for example, simple transitive sentences of the form 'subject-verb-object' or adjective-noun compounds).

The notion of a framework where relational words act as multi-linear maps on noun vectors has been formalized by Coecke et al. [25] in the abstract setting of category theory and compact closed categories, a topic we are going to discuss in more detail in Chapter 3. Baroni and Zamparelli's composition method for adjectives and nouns also follows the very same principle [6].

2.3.4 Deep learning models

A recent trend in compositionality of distributional models is based on deep learning techniques, a class of machine learning algorithms (usually neural networks) that approach models as multiple layers of representations, where the higher-level concepts are induced from the lower-level ones. For example, following ideas from the seminal

work of Pollack [94] conducted 25 years ago, Socher and colleagues [111, 112, 113] use recursive neural networks in order to produce compositional vector representations for sentences, with promising results in a number of tasks. In its most general form, a neural network like this takes as input a pair of word vectors $\overrightarrow{w_1}, \overrightarrow{w_2}$ and returns a new composite vector $\overrightarrow{y}$ according to the following equation:

$$\overrightarrow{y} = g(\mathbf{W}[\overrightarrow{w}_1; \overrightarrow{w}_2] + \overrightarrow{b}) \tag{2.15}$$

where $[\overrightarrow{w}_1; \overrightarrow{w}_2]$ denotes the concatenation of the two child vectors, $\mathbf{W}$ and $\overrightarrow{b}$ are the parameters of the model, and g is a element-wise non-linear function such as tanh. This output, then, will be used in a recursive fashion again as input to the network for computing the vector representations of larger constituents. This class of models is known as *recursive neural networks* (RecNNs), and adheres to the general architecture of Fig. 2.5.

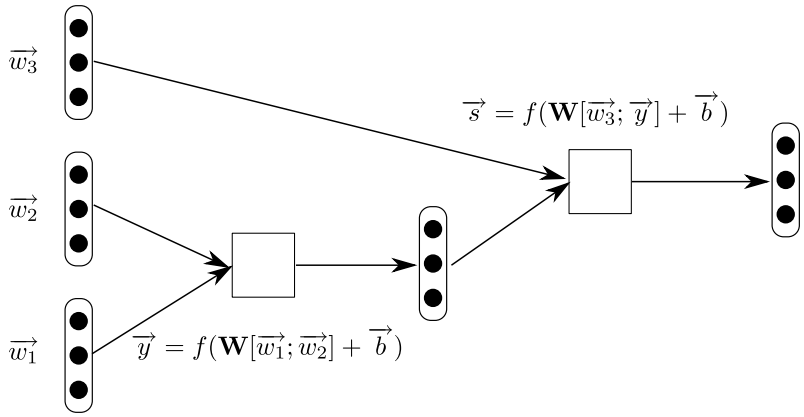


Figure 2.5: A recursive neural network with a single layer for providing compositionality in distributional models.

In a variation of the above structure, each intermediate composition is performed via an auto-encoder instead of a feed-forward network. A *recursive auto-encoder* (RAE) acts as an identity function, learning to reconstruct the input, encoded via a hidden layer, as faithfully as possible. The state of the hidden layer of the network is then used as a compressed representation of the two original inputs. Since the optimization is solely based on the reconstruction error, a RAE is considered as a form of unsupervised learning [111]. Finally, neural networks can vary in design and topology. Kalchbrenner et al. [45], for example, model sentential compositionality using a *convolutional* neural network, the main characteristic of which is that it acts on small overlapping parts of the input vectors.

In contrast to the previously discussed compositional approaches, deep learning methods are based on a large amount of training: the parameters $\mathbf{W}$ and $\overrightarrow{b}$ in the network of Fig. 2.5 must be learned through an iterative algorithm known as *backpropagation*, a process that can be very time-consuming and in general cannot guarantee optimality. However, the non-linearity in combination with the layered approach in which neural networks are based provide these models with great power, allowing them to simulate the behaviour of a range of functions much wider than the multi-linear maps of tensor-based approaches. Indeed, deep learning compositional models have been applied in various paraphrase detection and sentiment analysis tasks [111, 112, 113] delivering results that fall into state-of-the-art range.

2.3.5 Intuition behind various compositional approaches

At this point, it is instructive to pause for a moment and consider what exactly the term *composition* would mean for each one of the approaches reviewed in this section so far. Remember that in the case of vector mixtures every vector lives in the same base space, which means that a sentence vector has to share the same features and length with the word vectors. What a distributional vector for a word actually shows us is to what extent all other words in the vocabulary are related to this specific target word. If our target word is a verb, then the components of its vector can be seen as related to the *action* described by the verb: a vector for the verb 'run' reflects the degree to which a 'dog' can run, a 'car' can run, a 'table' can run and so on. The element-wise mixing of vectors $\overrightarrow{dog}$ and $\overrightarrow{run}$ then in order to produce a compositional representation for the meaning of the simple intransitive sentence 'dogs run', finds an intuitive interpretation: the output vector will reflect the extent to which things that are related to dogs can also run (and vice versa); in other words, it shows how *compatible* the verb is with the specific subject.[5]

A tensor-based model, on the other hand, goes beyond a simple compatibility check between the relational word and its arguments; its purpose is to *transform* the noun into a sentence. Furthermore, the size and the form of the sentence space become tunable parameters of the model, which can depend on the specific task in hand. Let us assume that in our model we select sentence and noun spaces such that $S \in \mathbb{R}^s$ and $N \in \mathbb{R}^n$, respectively; here, s refers to the number of distinct features that we consider appropriate for representing the meaning of a sentence in our model, while n is the corresponding number for nouns. An intransitive verb, then, like 'play'

[5]Admittedly, this rather idealistic behaviour does not apply to the same extent on every pair of composed words; however the provided basic intuition is in principle valid.

in 'kids play', will live in $N \otimes S \in \mathbb{R}^{n \times s}$, and will be a map $f : N \to S$ built (somehow) in a way to take as input a noun and produce a sentence; similarly, a transitive verb will live in $N \otimes S \otimes N \in \mathbb{R}^{n \times s \times n}$ and will correspond to a map $f_{tr} : N \otimes N \to S$.

Let me demonstrate this idea using a concrete example, where the goal is to simulate the truth-theoretic nature of formal semantics view in the context of a tensor-based model (perhaps for the purposes of a textual entailment task). In that case, our sentence space will be nothing more than the following:

$$S = \left\{ \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ 0 \end{pmatrix} \right\} \tag{2.16}$$

with the two vectors representing $\top$ and $\bot$, respectively. Each individual in our universe will correspond to a basis vector of our noun space; with just three individuals (Bob, John, Mary), we get the following mapping:

$$\overrightarrow{bob} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \quad \overrightarrow{john} = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \quad \overrightarrow{mary} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \tag{2.17}$$

In this setting, an intransitive verb will be a matrix formed as a series of truth values, each one of which is associated with some individual in our universe. Assuming for example that only John performs the action of sleeping, then the meaning of the sentence 'John sleeps' is given by the following computation:[6]

$$\overrightarrow{john}^{\mathsf{T}} \times \overline{sleep} = \begin{pmatrix} 0 & 1 & 0 \end{pmatrix} \times \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \top \tag{2.18}$$

For the case of tensor-based models, the nouns-to-sentence transformation has to be linear. The premise of deep-learning compositional models is to make this process much more effective by applying consecutive non-linear layers of transformation. Indeed, a neural network is not only able to project an input vector into a sentence space, but it can also *transform the geometry* of the space itself in order to make it reflect better the relationships between the points (sentences) in it. This quality is better demonstrated with a simple example: although there is no linear map which sends an input $x \in \{0, 1\}$ to the correct XOR value (as it is evident from part (a) of Fig. 2.6), the function can be successfully approximated by a simple feed-forward neural network with one hidden layer, similar to that in part (c) of the figure. Part (b) shows a generalization to an arbitrary number of points, which for our purposes

[6] Clark [20] details an extension of this simple model from truth values to probabilities that show how plausible is the statement expressed by the composite vector.

can be thought as representing two semantically distinct groups of sentences.

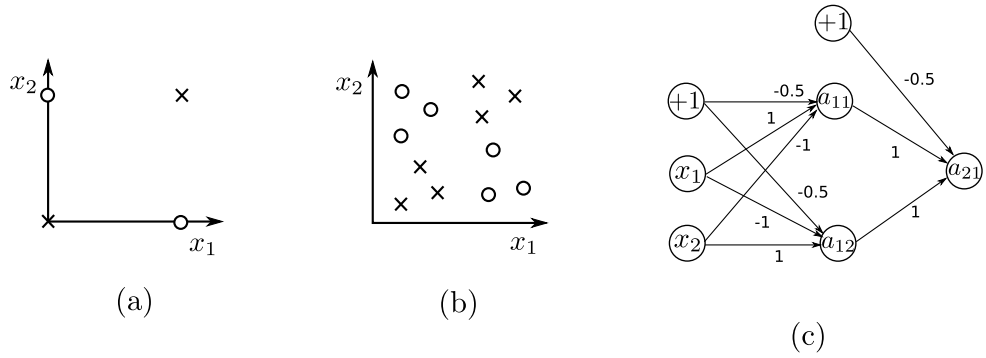


Figure 2.6: Intuition behind neural compositional models.

2.4 A taxonomy of CDMs

I would like to conclude this chapter by summarizing the presentation of the various CDM classes in the form of a concise diagram (Fig. 2.7). As we will see in Part II, this hierarchy is not complete; specifically, the work I present in this thesis introduces a new class of CDMs, which is flexible enough to include features from both tensor-based and vector mixture models. This is a direct consequence of applying Frobenius algebras on the creation of relational tensors, a topic that we will study in detail in Chapter 4.

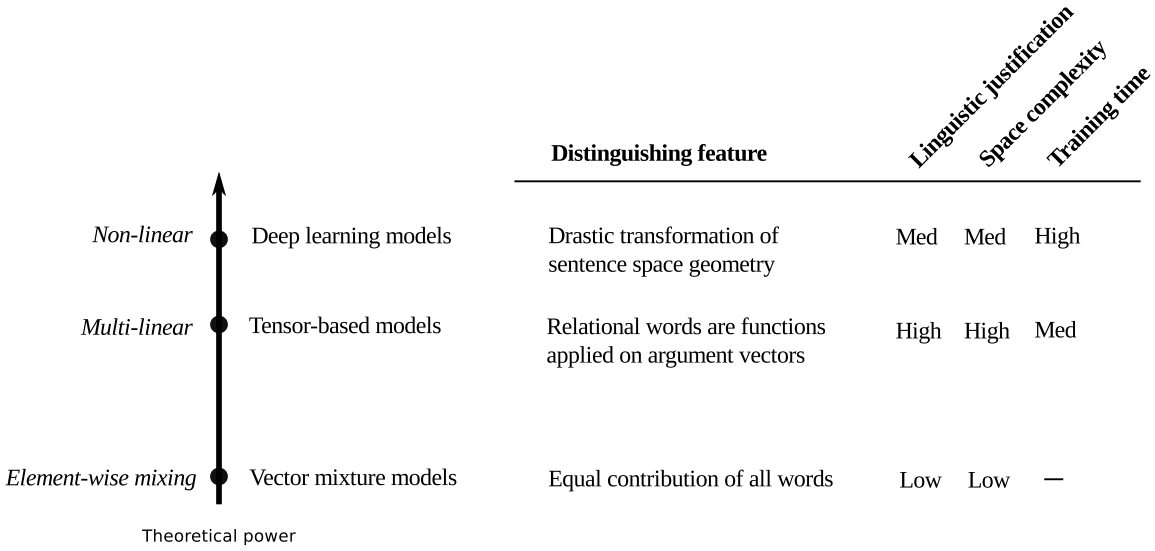


Figure 2.7: A taxonomy of CDMs.

Chapter 3

A Categorical Framework for Natural Language

Chapter Abstract

We now proceed to our main subject of study, the categorical framework of Coecke et al. [25]. The current chapter provides an introduction to the important notions of compact closed categories and pregroup grammars, and describes in detail how the passage from grammar to vector spaces is achieved. Furthermore, the reader is introduced to the concept of Frobenius algebras (another main topic of this work), as well as to a convenient diagrammatic notation that will be extensively used in the rest of this thesis. Material is based on [53, 46].

Tensor-based models stand in between the two extremes of vector mixtures and deep learning methods, offering an appealing alternative that can be powerful enough and at the same time fully aligned with the formal semantics view of natural language. Actually, it has been shown that the linear-algebraic formulas for the composite meanings produced by a tensor-based model emerge as the natural consequence of a structural similarity between a grammar and finite-dimensional vector spaces. Exploiting the abstract mathematical framework of category theory, Coecke, Sadrzadeh and Clark [25] managed to equip the distributional models of meaning with compositionality in a way that every grammatical reduction corresponds to a linear map defining mathematical manipulations between vector spaces. This result builds upon the fact that the base type-logic of the framework, a pregroup grammar [59], shares the same abstract structure with finite-dimensional vector spaces, that of a compact closed category. Mathematically, the transition from grammar types to vector spaces has the

form of a strongly monoidal functor, that is, of a map that preserves the basic structure of compact closed categories. The following section provides a short introduction to the category theoretic notions above.

3.1 Introduction to categorical concepts

Category theory is an abstract area of mathematics, the aim of which is to study and identify universal properties of mathematical concepts that often remain hidden by traditional approaches. A *category* is a collection of objects and morphisms that hold between these objects, with composition of morphisms as the main operation. That is, for two morphisms $A \xrightarrow{f} B \xrightarrow{g} C$, we have $g \circ f : A \to C$. Morphism composition is associative, so that $(h \circ g) \circ f = h \circ (g \circ f)$. Furthermore, every object A has an identity morphism $1_A : A \to A$; for $f : A \to B$ we moreover require that:

$$f \circ 1_A = f \quad \text{and} \quad 1_B \circ f = f \tag{3.1}$$

In category theory, the relationships between morphisms are often depicted through the use of *commutative diagrams*. We say that a diagram *commutes* if all paths starting and ending at the same points lead to the same result by the application of composition. The diagram below, for example, shows the associativity of composition:

$$\begin{array}{ccccc} A & & \xrightarrow{\quad f \quad} & & B \\ & {\scriptstyle g \circ f} \searrow & & \swarrow {\scriptstyle g} \; \searrow & \\ & & C & & \xrightarrow[h]{} D \end{array} \qquad {\scriptstyle h \circ g} \tag{3.2}$$

We now proceed to the definition of a *monoidal category*, which is a special type of category equipped with another associative operation, a monoidal tensor $\otimes : \mathcal{C} \times \mathcal{C} \to \mathcal{C}$ (where $\mathcal{C}$ is our basic category). Specifically, for each pair of objects (A, B) there exists a composite object $A \otimes B$, and for every pair of morphisms ($f : A \to C$, $g : B \to D$) a parallel composite $f \otimes g : A \otimes B \to C \otimes D$. For a *symmetric monoidal category*, it is also the case that $A \otimes B \cong B \otimes A$. Furthermore, there is a unit object I which satisfies the following natural isomorphisms:

$$A \otimes I \cong A \cong I \otimes A \tag{3.3}$$

The natural isomorphisms in Eq. 3.3 above, together with the natural isomor-

phism $A \otimes (B \otimes C) \xrightarrow{\alpha_{A,B,C}} (A \otimes B) \otimes C$ need to satisfy certain coherence conditions [67], which basically ensure the commutativity of all relevant diagrams.

Finally, a monoidal category is *compact closed* if every object A has a left and right adjoint, denoted as A^l, A^r respectively, for which the following special morphisms exist:

$$\eta^l : I \to A \otimes A^l \qquad \eta^r : I \to A^r \otimes A \tag{3.4}$$

$$\epsilon^l : A^l \otimes A \to I \qquad \epsilon^r : A \otimes A^r \to I \tag{3.5}$$

These adjoints are unique up to isomorphism. Furthermore, the above maps need to satisfy the following four special conditions (the so-called *yanking conditions*), which again ensure that all relevant diagrams commute:

$$\begin{aligned} (1_A \otimes \epsilon^l_A) \circ (\eta^l_A \otimes 1_A) = 1_A \qquad (\epsilon^r_A \otimes 1_A) \circ (1_A \otimes \eta^r_A) = 1_A \\ (\epsilon^l_A \otimes 1_{A^l}) \circ (1_{A^l} \otimes \eta^l_A) = 1_{A^l} \qquad (1_{A^r} \otimes \epsilon^r_A) \circ (\eta^r_A \otimes 1_{A^r}) = 1_{A^r} \end{aligned} \tag{3.6}$$

For the case of a *symmetric compact closed category*, the left and right adjoints collapse into one, so that $A^* := A^l = A^r$. As we will see, both a pregroup grammar and the category of finite-dimensional vector spaces over a base field conform to this abstract definition of a compact closed category. Before we proceed to this, though, we need to introduce another important concept, that of a strongly monoidal functor.

In general, a *functor* is a morphism between categories that preserves identities and composition. That is, a functor $\mathcal{F}$ between two categories $\mathcal{C}$ and $\mathcal{D}$ maps every object A in $\mathcal{C}$ to some object $\mathcal{F}(A)$ in $\mathcal{D}$, and every morphism $f : A \to B$ in $\mathcal{C}$ to a morphism $\mathcal{F}(f) : \mathcal{F}(A) \to \mathcal{F}(B)$ in $\mathcal{D}$, such that $\mathcal{F}(1_A) = 1_{\mathcal{F}(A)}$ for all objects in $\mathcal{C}$ and $\mathcal{F}(g \circ f) = \mathcal{F}(g) \circ \mathcal{F}(f)$ for all morphisms $f : A \to B, g : B \to C$. Note that in the case of monoidal categories, a functor must also preserve the monoidal structure. Taking any two monoidal categories $(\mathcal{C}, \otimes, I_\mathcal{C})$ and $(\mathcal{D}, \bullet, I_\mathcal{D})$, a *monoidal functor* is a functor that comes with a natural transformation (a structure-preserving morphism between functors) $\phi_{A,B} : \mathcal{F}(A) \bullet \mathcal{F}(B) \to \mathcal{F}(A \otimes B)$ and a morphism $\phi : I_\mathcal{D} \to \mathcal{F}(I_\mathcal{C})$, which as usual satisfy certain coherence conditions. Finally, preserving the compact structure between two categories so that $\mathcal{F}(A^l) = \mathcal{F}(A)^l$ and $\mathcal{F}(A^r) = \mathcal{F}(A)^r$ requires a special form of a monoidal functor, one whose $\phi_{A,B}$ and ϕ morphisms are invertible. Note that in this case, we get:

$$\mathcal{F}(A^l) \bullet \mathcal{F}(A) \xrightarrow{\phi_{A^l,A}} \mathcal{F}(A^l \otimes A) \xrightarrow{\mathcal{F}(\epsilon^l_{\mathcal{C}})} \mathcal{F}(I_{\mathcal{C}}) \xrightarrow{\phi^{-1}} I_{\mathcal{D}}$$
$$I_{\mathcal{D}} \xrightarrow{\phi} \mathcal{F}(I_{\mathcal{C}}) \xrightarrow{\mathcal{F}(\eta^l_{\mathcal{C}})} \mathcal{F}(A \otimes A^l) \xrightarrow{\phi^{-1}_{A,A^l}} \mathcal{F}(A) \bullet \mathcal{F}(A^l)$$

Since adjoints are unique, the above means that $\mathcal{F}(A^l)$ must be the left adjoint of $\mathcal{F}(A)$; the same reasoning applies for the case of the right adjoint. We will use such a *strongly monoidal functor* in §3.3 as a means for translating a grammatical derivation to a linear-algebraic formula between vectors and tensors.

3.2 Pregroup grammars

A *pregroup grammar* [59] is a type-logical grammar built on the rigorous mathematical basis of a pregroup algebra, i.e. a partially ordered monoid with unit 1, whose each element p has a left adjoint p^l and a right adjoint p^r. In the context of a pregroup algebra, Eqs 3.4 and 3.5 are transformed to the following:

$$p^l \cdot p \leq 1 \leq p \cdot p^l \quad \text{and} \quad p \cdot p^r \leq 1 \leq p^r \cdot p \tag{3.7}$$

where the dot denotes the monoid multiplication. It also holds that $1 \cdot p = p = p \cdot 1$. A pregroup grammar is the pregroup freely generated over a set of atomic types, for example n for well-formed noun phrases and s for well-formed sentences. Atomic types and their adjoints can be combined to form compound types, e.g. $n^r \cdot s \cdot n^l$ for a transitive verb. This type reflects the fact that a transitive verb is an entity expecting a noun to its right (the object) and a noun to its left (the subject) in order to return a sentence. The rules of the grammar are prescribed by the mathematical properties of pregroups, and specifically by the inequalities in (3.7) above. We say that a sequence of words $w_1 w_2 \dots w_n$ forms a grammatical sentence whenever we have:

$$t_1 \cdot t_2 \cdot \ldots \cdot t_n \leq s \tag{3.8}$$

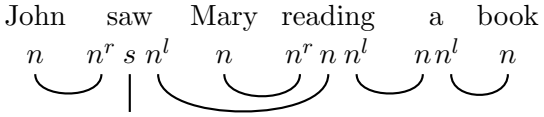
where t_i is the type of the ith word in the sentence. As an example, consider the following type dictionary:

John	saw	Mary	reading	a	book
n	$n^r \cdot s \cdot n^l$	n	$n^r \cdot n \cdot n^l$	$n \cdot n^l$	n

The juxtaposition of these words in the given order would form a grammatical sentence, as the derivation below shows:

$$
\begin{aligned}
n \cdot (n^r \cdot s \cdot n^l) \cdot n \cdot (n^r \cdot n \cdot n^l) \cdot (n \cdot n^l) \cdot n &= \\
(n \cdot n^r) \cdot s \cdot n^l \cdot (n \cdot n^r) \cdot n \cdot (n^l \cdot n) \cdot (n^l \cdot n) &\leq && \text{(associativity)} \\
1 \cdot s \cdot n^l \cdot 1 \cdot n \cdot 1 \cdot 1 &= && \text{(partial order)} \\
s \cdot (n^l \cdot n) &\leq && \text{(unitors)} \\
s & && \text{(partial order)}
\end{aligned}
\tag{3.9}
$$

Derivations in a pregroup grammar are usually shown as reduction diagrams, where contractions to the unit object due to inequalities $p^l \cdot p \leq 1$ and $p \cdot p^r \leq 1$ are depicted by lines connecting the corresponding elements:

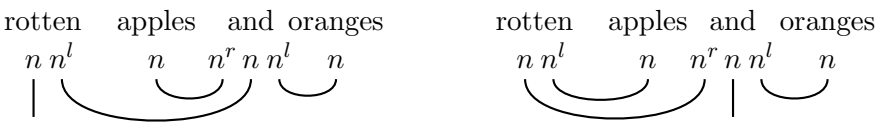


(3.10)

A pregroup grammar forms a compact closed category, to which I will refer as $\mathbf{Preg}_F$. The objects of this category are the basic types, their adjoints, and all tensor products of them, while the morphisms are given by the partial ordering, i.e. we take $p \leq q$ to denote the morphism $p \to q$. Note that this means that $\mathbf{Preg}_F$ is a *posetal* category; that is, a category between any two objects of which there exists *at most* one morphism. The tensor product is the monoid multiplication: for two objects p and q we have $p \cdot q$, and for two morphisms $p \leq q$ and $r \leq t$ we get $p \cdot r \leq q \cdot t$ by monotonicity of the monoid multiplication. Furthermore, from Eq. 3.7 we derive:

$$
\begin{aligned}
\epsilon^l &: p^l \cdot p \leq 1 \qquad \epsilon^r : p \cdot p^r \leq 1 \\
\eta^l &: 1 \leq p \cdot p^l \qquad \eta^r : 1 \leq p^r \cdot p
\end{aligned}
\tag{3.11}
$$

The category $\mathbf{Preg}_F$ is an obvious choice for accommodating our grammar, but it imposes a simplification: its posetal nature means that all possible derivations between two grammatical types would be identified, which is clearly an unwelcome property. Consider, for example the following case:



(3.12)

We obviously need a way to distinguish the two cases, since they lead to seman-

tically different results. The solution is to replace the free pregroup with the *free compact-closed category* over a partially ordered set of basic grammar types, as described by Preller and Lambek [96], abandoning the assumption that there is only one morphism for each domain and co-domain pair. This category, which is referred to by $\mathbf{C}_F$ in this work, has form identical to $\mathbf{Preg}_F$ with the important difference that two objects might be connected with more than one morphism; for the example above, the *hom-set*[1] between $n \cdot n^l \cdot n \cdot n^r \cdot n \cdot n^l \cdot n$ and n becomes:

$$\left\{(1_n \cdot \epsilon_n^l) \circ (1_{n \cdot n^l} \cdot \epsilon_n^l \cdot 1_n \cdot \epsilon_n^r), (\epsilon_n^r \cdot 1_n \cdot \epsilon_n^l) \circ (1_n \cdot \epsilon_n^l \cdot 1_{n^r \cdot n \cdot n^l \cdot n})\right\}$$

in which the first element corresponds to the left-hand derivation in (3.12), and the second element to the right-hand derivation. In the next section we will use $\mathbf{C}_F$ as the domain of our syntax-to-semantics functor.

3.3 Quantizing the grammar

Finite-dimensional vector spaces over a field also form a compact closed category, where the vector spaces are the objects, linear maps are the morphisms, the main operation is the tensor product between vector spaces, and the field over which the spaces are formed, in our case $\mathbb{R}$, is the unit. Following a category-theoretic convention, in this setting vectors $\overrightarrow{v} \in V$, $\overline{u} \in V \otimes W$ etc. are considered specific *states*[2] of their corresponding vector spaces, and are represented as morphisms from the unit object:

$$\overrightarrow{v} : I \to V \qquad\qquad \overline{u} : I \to V \otimes W \tag{3.13}$$

In contrast to $\mathbf{Preg}_F$ and $\mathbf{C}_F$, the category of finite-dimensional vector spaces and linear maps is symmetric. The adjoint of each vector space is its dual which, in the presence of a fixed basis, is isomorphic to the space itself. Note that any vector space with fixed basis (as it is always the case in distributional models of meaning) comes with an inner product; this provides a canonical way to model the behaviour of the ϵ-maps, which become the inner product between the involved vectors:

$$\epsilon^l = \epsilon^r : W \otimes W \to \mathbb{R} :: \sum_{ij} c_{ij}(\overrightarrow{w_i} \otimes \overrightarrow{w_j}) \mapsto \sum_{ij} c_{ij} \langle \overrightarrow{w_i} | \overrightarrow{w_j} \rangle \tag{3.14}$$

[1] The set of all morphisms between the two objects.

[2] This view of vectors as states will take a more concrete form in Chapter 6, where I resort to quantum mechanics for modelling ambiguity in language.

On the other hand, η-maps define identity matrices, which by linearity generalize to scalar products of these:

$$\eta^l = \eta^r : \mathbb{R} \to W \otimes W :: 1 \mapsto \sum_i \overrightarrow{w_i} \otimes \overrightarrow{w_i} \tag{3.15}$$

Let us refer to the category of finite-dimensional vector spaces and linear maps over a field as **FVect**. In the original paper [25], the unification of grammar and vector spaces was achieved by working on the product category $\mathbf{FPreg}_F \times \mathbf{FVect}$; for a type reduction $p_1 \cdot p_2 \cdot \ldots \cdot p_n \leq x$, the transition to vector spaces had the form of a map $(\leq, f)$ such that:

$$(\leq, f) : (p_1 \cdot p_2 \cdot \ldots \cdot p_n, W_1 \otimes \ldots \otimes W_n) \to (x, W) \tag{3.16}$$

with f representing the syntax-to-semantics morphism. Later, this was recast into a functorial passage from a pregroup dictionary to **FVect** from Preller and Sadrzadeh [95] and Kartsaklis, Sadrzadeh, Pulman and Coecke [53], while in [22] Coecke, Grefenstette and Sadrzadeh define a functor from Lambek monoids. The main difference of this latter work is that the grammar is expressed as a *monoidal bi-closed category*, a structure that is closer than a pregroup to the traditional form of a categorial grammar as this was formalized in the context of Lambek calculus [60] and the work of Ajdukiewicz [2] and Bar-Hillel [5]. For the purposes of this thesis I will not consider this approach, and I will define the transition from grammar type reductions to vector spaces along the lines of [95] and [53].

Assuming the canonical choices of Eq. 3.14 and Eq. 3.15 for the structural ϵ and η morphisms, respectively, defining a strongly monoidal functor:

$$\mathcal{F} : \mathbf{C}_F \to \mathbf{FVect} \tag{3.17}$$

culminates in choosing an appropriate vector space for each atomic type $x \in \mathcal{T}$ (where $\mathcal{T}$ is the partially ordered set of the atomic pregroup types) so that $\mathcal{F}(x) = X$. Note that since the set $\mathrm{Ob}(\mathbf{C}_F)$[3] is freely generated over $\mathcal{T}$ means that we can think of the objects in functor's co-domain as also being freely generated over a set $\{\mathcal{F}(t) | t \in \mathcal{T}\}$—which provides vector space counterparts for all complex types in our grammar. I proceed to make the above precise and give the basic properties of $\mathcal{F}$.

Firstly, it maps the pregroup unit to the field of our vector spaces:

[3]The set $\mathrm{Ob}(\mathcal{C})$ is the set of objects in category $\mathcal{C}$.

$$\mathcal{F}(1) = \mathbb{R} \tag{3.18}$$

As discussed above, for each atomic grammar type x, the functor assigns a vector space X so that $\mathcal{F}(x) = X$. For example:

$$\mathcal{F}(n) = N \qquad \mathcal{F}(s) = S \tag{3.19}$$

Note that in principle the assigned vector space does not have to be different for each grammatical type. Later, in Chapter 4, we will see that using the same vector space for nouns and sentences leads to interesting consequences.

Regarding the mappings of the adjoints, we first notice that since **FVect** is symmetric, it is the case that:

$$\mathcal{F}(x)^l = \mathcal{F}(x)^r = \mathcal{F}(x)^* \tag{3.20}$$

Furthermore, the fact that we work with fixed basis means that $\mathcal{F}(x)^* \cong \mathcal{F}(x)$. As a result, we have:

$$\mathcal{F}(x^l) = \mathcal{F}(x)^l = \mathcal{F}(x)^* = \mathcal{F}(x) \tag{3.21}$$

and similarly for the right adjoint. From strong monoidality, it follows that the complex types are mapped to tensor products of vector spaces:

$$\mathcal{F}(x \cdot y) = \mathcal{F}(x) \otimes \mathcal{F}(y) \tag{3.22}$$

Putting together all the above, we get type mappings of the following form:

$$\mathcal{F}(n \cdot n^l) = \mathcal{F}(n^r \cdot n) = N \otimes N \qquad \mathcal{F}(n^r \cdot s \cdot n^l) = N \otimes S \otimes N \tag{3.23}$$

The structural morphisms ϵ and η are translated as follows:

$$\mathcal{F}(\epsilon^l_{\mathbf{C}_F}) = \mathcal{F}(\epsilon^r_{\mathbf{C}_F}) = \epsilon_{\mathbf{FVect}} \qquad \mathcal{F}(\eta^l_{\mathbf{C}_F}) = \mathcal{F}(\eta^r_{\mathbf{C}_F}) = \eta_{\mathbf{FVect}} \tag{3.24}$$

with $\epsilon_{\mathbf{FVect}}$ and $\eta_{\mathbf{FVect}}$ defined as in Eqs 3.14 and 3.15, respectively. Finally, each other morphism in $\mathbf{C}_F$ is mapped to a linear map in **FVect**. I will use the case of a transitive sentence as an example. Here, the subject and the object have the type n, whereas the type of the verb is $n^r \cdot s \cdot n^l$, as described above. The derivation in pregroups form proceeds as follows:

$$n \cdot (n^r \cdot s \cdot n^l) \cdot n = (n \cdot n^r) \cdot s \cdot (n^l \cdot n) \leq 1 \cdot s \cdot 1 = s \tag{3.25}$$

which corresponds to the following morphism in $\mathbf{C}_F$:

$$\epsilon_n^r \cdot 1_s \cdot \epsilon_n^l : n \cdot n^r \cdot s \cdot n^l \cdot n \to s \tag{3.26}$$

Applying our functor $\mathcal{F}$ to this will give:

$$\begin{aligned} \mathcal{F}(\epsilon_n^r \cdot 1_s \cdot \epsilon_n^l) &= \mathcal{F}(\epsilon_n^r) \otimes \mathcal{F}(1_s) \otimes \mathcal{F}(\epsilon_n^l) \\ &= \epsilon_{\mathcal{F}(n)} \otimes 1_{\mathcal{F}(s)} \otimes \epsilon_{\mathcal{F}(n)} \\ &= \epsilon_N \otimes 1_S \otimes \epsilon_N : N \otimes N \otimes S \otimes N \otimes N \to S \end{aligned} \tag{3.27}$$

The function f in Eq. 1.1, suggested as a way of calculating the meaning of a sentence, now takes a concrete form strictly based on the grammar rules that connect the individual words. I will now proceed and formally define the meaning of a sentence as follows:

Definition 3.3.1. Take $\overrightarrow{w_i}$ to be a vector $I \to \mathcal{F}(p_i)$ corresponding to word w_i with pregroup type p_i in a sentence $w_1 w_2 \dots w_n$. Given a type-reduction $\alpha : p_1 \cdot p_2 \cdot \dots \cdot p_n \to s$, the meaning of the sentence is defined as follows:

$$\mathcal{F}(\alpha)(\overrightarrow{w_1} \otimes \overrightarrow{w_2} \otimes \dots \otimes \overrightarrow{w_n}) \tag{3.28}$$

Let us see what Def. 3.3.1 means from a linear-algebraic perspective. Working on the transitive case a little further, we observe that the map of pregroup types to tensors prescribes that the subject and object are vectors (Eq. 3.19) while the verb is a tensor of order 3 (Eq. 3.23). Hence, for a simple sentence such as 'dogs chase cats' we have the following geometric entities:

$$\overrightarrow{dogs} = \sum_i c_i^{dogs} \overrightarrow{n_i} \qquad \overrightarrow{chase} = \sum_{ijk} c_{ijk}^{chase} (\overrightarrow{n_i} \otimes \overrightarrow{s_j} \otimes \overrightarrow{n_k}) \qquad \overrightarrow{cats} = \sum_k c_k^{cats} \overrightarrow{n_k}$$

where c_i^v denotes the ith component in vector $\overrightarrow{v}$, $\overrightarrow{n_i}$ is a basis vector of N and $\overrightarrow{s_i}$ a basis vector of S. Applying Def. 3.3.1 on this sentence will give:

$$
\begin{aligned}
\mathcal{F}(\epsilon_n^r \cdot 1_s \cdot \epsilon_n^l)(\overrightarrow{dogs} \otimes \overline{chase} \otimes \overrightarrow{cats}) = \\
(\epsilon_N \otimes 1_S \otimes \epsilon_N)(\overrightarrow{dogs} \otimes \overline{chase} \otimes \overrightarrow{cats}) = \\
\sum_{ijk} c_{ijk}^{chase} \langle \overrightarrow{dogs} | \overrightarrow{n_i} \rangle \overrightarrow{s_j} \langle \overrightarrow{n_k} | \overrightarrow{cats} \rangle = \\
\sum_{ijk} c_{ijk}^{chase} c_i^{dogs} c_k^{cats} \overrightarrow{s_j} = \\
\overrightarrow{dogs}^{\mathsf{T}} \times \overline{chase} \times \overrightarrow{cats}
\end{aligned}
\tag{3.29}
$$

where the symbol $\times$ denotes tensor contraction, and the result lives in S. Thus we have arrived at Eq. 2.14, presented in §2.3.3 as a means for calculating the vector of a transitive sentence in the context of a tensor-based model.

The significance of the categorical framework lies exactly in this fact, that it provides an elegant mathematical counterpart of the formal semantics perspective as expressed by Montague [77], where words are represented and interact with each other according to their type-logical identities. Furthermore, it seems to imply that approaching the problem of compositionality in a tensor-based setting is a step in the right direction, since the linear-algebraic manipulations are a direct consequence of the grammatical derivation.

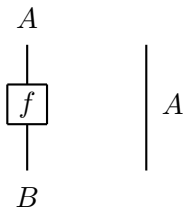
3.4 Pictorial calculus

Compact closed categories are a special case of monoidal categories, which are *complete* with regard to a convenient pictorial calculus [54, 109]. This means that an equation between morphisms follows from the axioms of monoidal categories if and only if it holds, up to *planar isotopy*, in the pictorial calculus. Selinger [109] explains the "planar isotopy" requirement as follows:

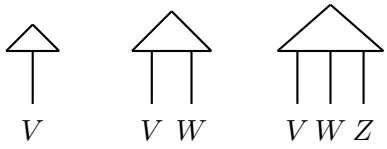
> [...] two diagrams drawn in a rectangle in the plane [...] are equivalent if it is possible to transform one to the other by continuously moving around boxes in the rectangle, without allowing boxes or wires to cross each other or to be detached from the boundary of the rectangle during the moving.

The diagrammatic language visualizes the derivations in an intuitive way, focusing on the interactions between the various objects, and I will use it extensively for demonstrating the main points of this thesis. In this section I introduce the fragment

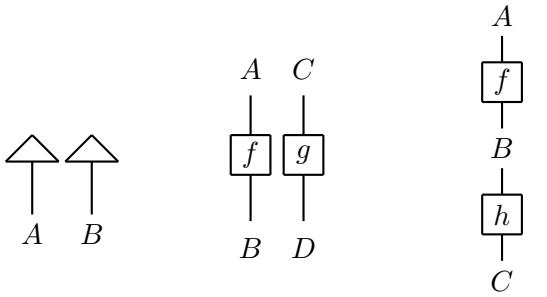
of the calculus that is relevant to the current work. A morphism $f : A \to B$ is depicted as a box with incoming and outgoing wires representing the objects; the identity morphism $1_A : A \to A$ is just a straight line.



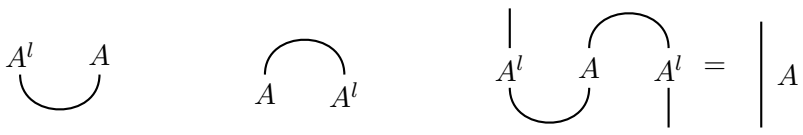
Recall that a vector is represented as a morphism from the unit object. In our graphical calculus, the unit object is depicted as a triangle, while the number of wires emanating from it denotes the order of the corresponding tensor:



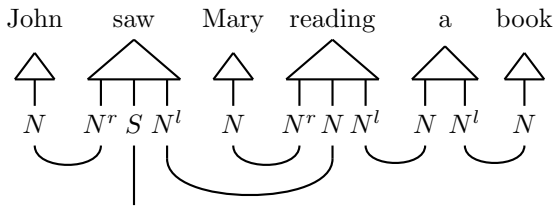
Tensor products of objects and morphisms are depicted by juxtaposing the corresponding diagrams side by side. Composition, on the other hand, is represented as a vertical superposition. For example, from left to right, here are the pictorial representations of the tensor of a vector in A with a vector in B, a tensor of morphisms $f \otimes g : A \otimes C \to B \otimes D$, and a composition of morphisms $h \circ f$ for $f : A \to B$ and $h : B \to C$.



The ϵ-maps are represented as cups ($\cup$) and the η-maps as caps ($\cap$). Equations such as $(\epsilon^l_A \otimes 1_{A^l}) \circ (1_{A^l} \otimes \eta^l_A) = 1_A$ (one of the four yanking conditions of compact closed categories) now get an intuitive visual justification:



Let us now see how the grammatical derivation in (3.10) is translated to **FVect**:[4]



(3.30)

Categorically, the derivation of a general transitive sentence of the form *subject-verb-object* can be depicted by the following commutative diagram:

$$\begin{array}{ccc} I \otimes I \otimes I & \xrightarrow{\overrightarrow{subj} \otimes \overline{verb} \otimes \overrightarrow{obj}} & N \otimes N^r \otimes S \otimes N^l \otimes N \\ \wr\| & & \downarrow \epsilon^r_N \otimes 1_S \otimes \epsilon^l_N \\ I & \xrightarrow[(\epsilon^r_N \otimes 1_S \otimes \epsilon^l_N) \circ (\overrightarrow{subj} \otimes \overline{verb} \otimes \overrightarrow{obj})]{} & S \end{array} \tag{3.31}$$

which demonstrates the fact that the composition morphism is a state vector in the sentence space.

3.5 Frobenius algebras

I will close this chapter by introducing the notion of a Frobenius algebra, which is going to play an important role for the rest of this thesis. Frobenius algebras were named after the German mathematician Ferdinand Georg Frobenius, and have been studied extensively from the 1930s onwards. The basis of a solid duality theory was set by Nakayama in 1939 [81] and 1941 [82]. Since then, they have found applications in various fields of mathematics and physics, e.g. in topological quantum field theories [56] and in categorical quantum mechanics [26, 24]. The general categorical definitions recalled below are due to Carboni and Walters [18], whereas their concrete instantiations to algebras over vector spaces were developed by Coecke and Pavlovic [26].

A Frobenius algebra over a symmetric monoidal category $(\mathcal{C}, \otimes, I)$ is a tuple $(F, \Delta, \iota, \mu, \zeta)$, where for an F object of $\mathcal{C}$ the triple (F, Δ, ι) is an associative coalgebra; that is, the following are morphisms of $\mathcal{C}$, satisfying the coalgebraic associativity and unit conditions:

[4]Despite the fact that our functor sends atomic types and their adjoints to the same base vector space (Eq. 3.21), in my diagrams I will occasionally take the liberty to refer to the tensored vector spaces with names denoting their original grammatical type for enhanced readability.

$$\Delta \colon F \to F \otimes F \qquad\qquad \iota \colon F \to I \tag{3.32}$$

The triple (F, μ, ζ) is an associative algebra, which means that the following are morphisms of $\mathcal{C}$, satisfying the algebraic associativity and unit conditions:

$$\mu \colon F \otimes F \to F \qquad\qquad \zeta \colon I \to F \tag{3.33}$$

Moreover, the above Δ and μ morphisms satisfy the following *Frobenius* condition:

$$(\mu \otimes 1_F) \circ (1_F \otimes \Delta) \;=\; \Delta \circ \mu \;=\; (1_F \otimes \mu) \circ (\Delta \otimes 1_F) \tag{3.34}$$

A Frobenius algebra is *commutative* if it satisfies the following two conditions for $\sigma : A \otimes B \to B \otimes A$, the symmetry morphism of $(\mathcal{C}, \otimes, I)$:

$$\sigma \circ \Delta = \Delta \qquad\qquad \mu \circ \sigma = \mu \tag{3.35}$$

Finally, a Frobenius algebra is *isometric* or *special* if it satisfies the following condition:

$$\mu \circ \Delta = 1 \tag{3.36}$$

In the category **FVect**, any vector space V with a fixed basis $\{\overrightarrow{v_i}\}_i$ has a commutative special Frobenius algebra over it, explicitly given as follows:

$$\begin{aligned} \Delta :: \overrightarrow{v_i} \mapsto \overrightarrow{v_i} \otimes \overrightarrow{v_i} \qquad\qquad & \iota :: \overrightarrow{v_i} \mapsto 1 \\ \mu :: \overrightarrow{v_i} \otimes \overrightarrow{v_j} \mapsto \delta_{ij} \overrightarrow{v_i} := \begin{cases} \overrightarrow{v_i} & i = j \\ \overrightarrow{0} & i \neq j \end{cases} \qquad\qquad & \zeta :: 1 \mapsto \sum_i \overrightarrow{v_i} \end{aligned} \tag{3.37}$$

In such Frobenius algebras, therefore, the operation Δ corresponds to *copying* of the basis and its unit ι corresponds to deleting of the basis. They enable one to faithfully encode vectors of V into spaces with higher tensor orders, such as $V \otimes V, V \otimes V \otimes V, \cdots$. I will use this fact extensively in Chapter 4 in order to encode relational tensors of lower order into higher order spaces. In linear algebraic terms,

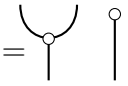
for $\overrightarrow{v} \in V$, we have that $\Delta(\overrightarrow{v})$ is a diagonal matrix whose diagonal elements are the weights of $\overrightarrow{v}$. As an example, consider V to be a 2-dimensional space with basis $\{\overrightarrow{v_1}, \overrightarrow{v_2}\}$. A vector $\overrightarrow{v} = a\overrightarrow{v_1} + b\overrightarrow{v_2}$ then can be encoded into $V \otimes V$ as follows:

$$\Delta(\overrightarrow{v}) = \Delta \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} a & 0 \\ 0 & b \end{pmatrix} = a\overrightarrow{v_1} \otimes \overrightarrow{v_1} + b\overrightarrow{v_2} \otimes \overrightarrow{v_2} \tag{3.38}$$

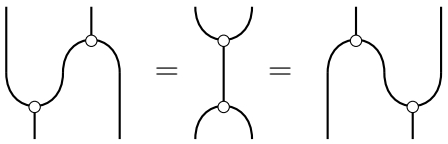
The operation μ is referred to as *uncopying* of the basis. In linear algebraic terms, for $\overline{z} \in V \otimes V$, we have that $\mu(\overline{z})$ is a vector consisting only of the diagonal elements of $\overline{z}$, hence losing the information encoded in the non-diagonal part. Concretely, if $\overline{z} = a\overrightarrow{v_1} \otimes \overrightarrow{v_1} + b\overrightarrow{v_1} \otimes \overrightarrow{v_2} + c\overrightarrow{v_2} \otimes \overrightarrow{v_1} + d\overrightarrow{v_2} \otimes \overrightarrow{v_2}$ in $V \otimes V$, this gets the following form:

$$\mu(\overline{z}) = \mu \begin{pmatrix} a & b \\ c & d \end{pmatrix} = \begin{pmatrix} a \\ d \end{pmatrix} = a\overrightarrow{v_1} + d\overrightarrow{v_2} \tag{3.39}$$

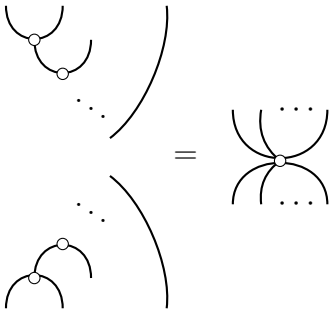
Computations with Frobenius algebras can also be represented within the more general diagrammatic calculus introduced in §3.4. Specifically, the linear maps of the coalgebra and algebra are depicted by:

$(\Delta, \iota) =$  $(\mu, \zeta) =$  (3.40)

which means that the Frobenius condition takes this form:

 (3.41)

Finally, the Frobenius condition guarantees that any diagram depicting a Frobenius algebraic computation can be reduced to a normal form that only depends on the number of input and output wires of the nodes, provided that the diagram of computation is connected. This property (a high level abstract account of which is due to Lack [58]) is referred to as the *spider equality* and is depicted as follows:

 (3.42)

This completes the presentation of the compositional model of Coecke et al. and the related categorical notions. At this point I should note, though, that as this thesis progresses new concepts will be introduced along the way. In Chapter 6, for example, the idea of an inner product space, which currently is only implicitly defined by the presence of fixed basis in our vector spaces, will be formalized in the context of *dagger compact closed categories* and specifically the category of finite-dimensional Hilbert spaces and linear maps (**FHilb**). Dagger compact closed categories (§6.3.3), as well as *dagger Frobenius algebras* (§6.5), constitute the fundamental structures of categorical quantum mechanics as set by Abramsky and Coecke [1], to which I will resort later in order to provide an extension of the original model capable of handling lexical ambiguity.

We are now ready to proceed to Part II, which details the theory behind the contributions of this thesis. Towards this purpose, the diagrammatic calculus presented in §3.4 will be one of my main tools for demonstrating the various concepts and ideas. When I find appropriate, I will also resort to the use of standard categorical commutative diagrams, similar to (3.31), for additional clarity. Before closing this chapter I would like to refer the reader who is interested in a more complete introduction to monoidal categories, Frobenius algebras, and their diagrammatic calculi, to the work of Coecke and Paquette [23].

Part II

Theory

Chapter 4

Building Verb Tensors using Frobenius Algebras

Chapter Abstract

I begin my analysis by providing an intuition of tensors as a means for modelling relations. Then I proceed and present an implementation framework for tensors of relational words based on Frobenius algebras that extends and improves previous proposals in a number of ways. After I provide a linguistic intuition of the suggested framework, I conclude the chapter by presenting a new CDM hierarchy that incorporates the newly devised class of models. Material is based on [53, 49, 51].

To a great extent, the essence of any tensor-based model lies in the functions that produce the sentences, i.e. the verbs. As we saw in §2.3.5, the purpose of such a function is to take a number of inputs (vectors representing nouns or noun phrases) and transform them to a sentence. The important question of how a function of this form can be constructed can obviously find more than one answer, many of which might depend on the underlying task at hand or the available resources. A goal of this thesis, though, is to not be dependent on factors of this sort; I am interested in reasoning about natural language meaning at a generic level, with the hope that the findings would have something more to say than how effective a specific CDM can be on some specific task. So let me start "by the book" and adopt a formal semantics perspective, where the meaning of a verb can be seen as the set of all elements in the universe of discourse who perform the action that the verb describes. As will become evident, this approach leads naturally to the tensor construction method of

Grefenstette and Sadrzadeh [40], which will serve as a starting point for the models presented in this thesis.

4.1 Verb tensors as relations

Recall from the discussion in §2.1 that under an extensional perspective of semantics the meaning of an intransitive verb is the set of individuals who perform the action denoted by the verb:

$$[\![sleep]\!] = \{x | x \in \mathcal{U} \wedge x \text{ sleeps}\} \tag{4.1}$$

where $\mathcal{U}$ denotes the universe of discourse. I will now start to transfer this setting to a vector space environment, adopting a truth-theoretic instantiation similar to the example in §2.3.5. As in that case, the noun space N is spanned by a set of mutually orthogonal vectors $\{\overrightarrow{n_i}\}_i$ denoting individuals (John is represented by $\overrightarrow{n_1}$, Mary by $\overrightarrow{n_2}$ and so on). I adopt the generic convention that maps individual n_i to $\overrightarrow{n_i}$.[1] So the entirety of the universe of discourse is given by a vector of 1s $\overrightarrow{U} = \sum_{n \in \mathcal{U}} \overrightarrow{n}$. The vectorial representation of a set of individuals $A \subseteq \mathcal{U}$ is just the sum of the corresponding noun vectors; naturally, the ith element of such a vector will be 1 if the individual who is denoted by $\overrightarrow{n_i}$ is included in set A, and 0 otherwise. Continuing on the intransitive verb example, let A be the set of all individuals whose denotations occur at least once as a subject of the specific verb in some corpus. Then, according to the definition above, the vectorial representation of this set is the following:

$$\overrightarrow{verb}_{IN} = \sum_{n \in A} \overrightarrow{n} \tag{4.2}$$

We have now an intuitive way to represent the meaning of a relational word, which in our vector space semantics takes the form of a summation over the vectors denoting the arguments of that word. Inevitably, though, this leads to a result living in the noun space N, seemingly contradicting the theoretical basis laid out in Chapter 3, which clearly states that words of type $n^r \cdot s$ (such as an intransitive verb) should live in the tensor product space $N \otimes S$. The functorial relation between grammatical types and tensors can be restored if we observe that any set of individuals is in one-to-one correspondence with a *characteristic function* that maps elements of the domain $\mathcal{U}$ to truth values; in other words, Eq. 4.1 can also be given as:

[1] In other words, n_i refers to an actual person such as John or Mary, while $\overrightarrow{n_i}$ is the model-specific representation of that person.

$$[\![sleep]\!] = \{(x,t)|x \in \mathcal{U}, t = \top \text{ if } x \text{ sleeps and } \bot \text{ o.w.}\} \tag{4.3}$$

Take χ_A to be the characteristic function of set A; furthermore, let the sentence space S be the one-dimensional space spanned by the single vector $\overrightarrow{1}$, which represents the truth value $\top$, whereas value $\bot$ is given by vector $\overrightarrow{0}$. Then, by mapping each pair (x,t) of χ_A to the tensor product of the vectors denoting x and t, we can re-define the meaning representation of the verb as below:

$$\overline{verb}_{IN} = \sum_{(n_i,[\![\overrightarrow{s_i}]\!])\in\chi_A} \overrightarrow{n_i} \otimes \overrightarrow{s_i} \tag{4.4}$$

with $\overrightarrow{s_i}$ to be $\overrightarrow{1}$ if the individual n_i occurs as a subject of the intransitive verb and $\overrightarrow{0}$ otherwise, and $[\![\overrightarrow{s_i}]\!]$ the corresponding truth value. The resulting tensor lives in $N \otimes S$ as dictated by the grammar; furthermore, and due to the form of S, it is isomorphic to the vector of Eq. 4.2. Assuming now that individual Mary is denoted by $\overrightarrow{n_2}$, we can compute the meaning of the sentence "Mary sleeps" as follows:

$$\begin{aligned}
(\epsilon_N^r \otimes 1_S)\left(\overrightarrow{n_2} \otimes \sum_i (\overrightarrow{n_i} \otimes \overrightarrow{s_i})\right) &= (\epsilon_N^r \otimes 1_S)\left(\sum_i \overrightarrow{n_2} \otimes \overrightarrow{n_i} \otimes \overrightarrow{s_i}\right) \\
&= \sum_i \langle \overrightarrow{n_2}|\overrightarrow{n_i}\rangle \overrightarrow{s_i} = \sum_i \delta_{2i}\overrightarrow{s_i} \\
&= \overrightarrow{s_2}\begin{cases} \overrightarrow{1} & \text{if } n_2 \in A \\ \overrightarrow{0} & o.w. \end{cases}
\end{aligned} \tag{4.5}$$

Note that in practice this is just the inner product of the vector of Mary with the vector representing the subset of individuals who sleep, given by Eq. 4.2. Let us now examine the case of a word with two arguments, such as a transitive verb. From an extensional perspective, this will be a relation of the following form:

$$[\![likes]\!] = \{(x,y)|x,y \in \mathcal{U} \wedge x \text{ likes } y\} \tag{4.6}$$

Let $\overrightarrow{n_i}, \overrightarrow{n_j}$ be the vectors denoting the two individuals in a pair of that relation (so that n_i likes n_j), and take $\overrightarrow{n_i} \otimes \overrightarrow{n_j}$ to be the semantic representation of that pair. This will be a matrix of size $|\mathcal{U}| \times |\mathcal{U}|$ with all zeros except a single 1 at position (i,j). It is now straightforward to model the relation of Eq. 4.6 as the sum of all its pairs, in the following manner:

$$\overline{verb}_{TR} = \sum_{(\overrightarrow{n_s},\overrightarrow{n_o})\in R} \overrightarrow{n_s} \otimes \overrightarrow{n_o} \tag{4.7}$$

where R is the underlying relation and $\overrightarrow{n_s}$, $\overrightarrow{n_o}$ denote individuals occurring as subject and object, respectively, of the specific verb in the training corpus. The result will be a matrix with an element in position (i, j) to be 1 if (n_i, n_j) has occurred as context of the verb (i.e. with n_i to be the subject of the verb and n_j its object), and 0 otherwise. As in the one-argument case, the functorial relation with the grammatical type of the verb ($n^r \cdot s \cdot n^l$) can be retained by replacing the matrix of Eq. 4.7 with the following isomorphic order-3 tensor:

$$\overline{verb}_{TR} = \sum_{((n_i,n_j),[\![\overrightarrow{s_{ij}}]\!])\in\chi_R} \overrightarrow{n_i} \otimes \overrightarrow{s_{ij}} \otimes \overrightarrow{n_j} \tag{4.8}$$

where this time χ_R is the characteristic function of the relation defining the contexts of the transitive verb, and $\overrightarrow{s_{ij}}$ is a one-dimensional vector denoting the inclusion or the absence of the corresponding pair of individuals in that relation. It is trivial to show that computing the meaning of a sentence such as "n_i likes n_j" will always result in the corresponding $\overrightarrow{s_{ij}}$ vector, i.e. the (i, j)th element of the matrix created by Eq. 4.7.

The discussion so far suggests that the "argument tensoring-and-summing" process described above is a valid way for one to model the truth-theoretic nature of formal approaches in natural language semantics, when using a vector space setting. In the first concrete implementation of the categorical framework from Grefenstette and Sadrzadeh [40], the authors suggest a generalization of this method to high dimensional real-valued vectors. Specifically, for the case of a transitive verb they propose the creation of a matrix as follows:

$$\overline{verb} = \sum_i (\overrightarrow{subj_i} \otimes \overrightarrow{obj_i}) \tag{4.9}$$

where $\overrightarrow{subj_i}$ and $\overrightarrow{obj_i}$ are the distributional vectors of subject and object, respectively, created as described in §2.2, and i iterates over all contexts of the specific verb in the training corpus. As explained in the doctoral thesis of Grefenstette [38], this process results in a structural mixing of the argument vectors, which reflects the extent to which each component of the subject vector is compatible with all the components of the object vector, and vice versa.

For the purposes of this thesis, the argument summing procedure of [40] has many important advantages. As demonstrated above, it is obviously aligned with the formal semantics perspective, which makes perfect sense for a model that can be seen as a mathematical counterpart of this traditional view of meaning in language.

Furthermore, it is generic enough not to be tied to any particular task or methodology, while at the same time its implementation and testing remain straightforward. Last, but not least, it creates *reduced representations* for relational words, since in all cases the order of the produced tensor is lower by one than the order dictated by the grammatical type. This is a quite important achievement, since for tensor-based models space complexity is always an issue. If, for example, we assume noun and sentence vectors with 300 dimensions, a transitive verb (a tensor of order 3) will require 27 million parameters, while for the case of a ditransitive verb this number is increased to 8.1 billion. Overall, I consider the argument summing procedure an interesting test-bed and an appropriate tool for demonstrating the ideas presented in this thesis.

4.2 Introducing Frobenius algebras in language

Despite the many benefits of the reduced tensors presented in §4.1, there is still an important problem we have to solve in order to make this method viable for real-world tasks. Recall that in its current form, our sentence space is the unit (the field). The transition from the truth-theoretic setting to real-valued vector spaces means that the single element that will result from a composition would not be restricted to 0 or 1, but it would be some number that, under proper normalization conditions, can show how *probable* is the specific subject-object pair for the verb at hand. Our composition function in this case is nothing more than an endomorphism of the field, as shown below for a transitive verb matrix created by Eq. 4.9:

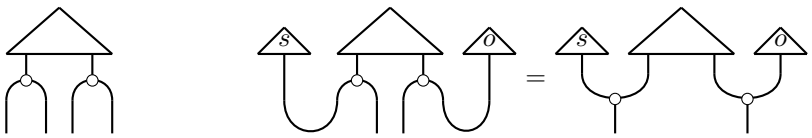
$$I \cong I \otimes I \otimes I \xrightarrow{\overrightarrow{s} \otimes \overline{v} \otimes \overrightarrow{o}} N \otimes N \otimes N \otimes N \xrightarrow{\epsilon_N^r \otimes \epsilon_N^l} I \otimes I \cong I \tag{4.10}$$

For any practical application, this is of course inadequate. We need a way to expand the verb matrix into a tensor of higher order, thus providing somehow the sentence dimension that is now missing. Recall from our discussion in §3.5 that every vector space with a fixed basis comes with a Frobenius algebra over it, which provides canonical ways for *copying* or *deleting* elements of that basis. Since in this case we are interested in copying, our tool will be a Frobenius map $\Delta : N \to N \otimes N$, applied on one or more of the original dimensions of our matrix, which currently lives in $N \otimes N$.

As it turns out, the decision of which (and how many) dimensions of the verb matrix to copy has some important consequences. We will start by examining the case of copying both dimensions, in which our inflated tensor takes the following form:

$$I \xrightarrow{\overline{verb}} N \otimes N \xrightarrow{\Delta_N \otimes \Delta_N} N \otimes N \otimes N \otimes N \tag{4.11}$$

How does this tensor form fit to the required grammatical type $n^r \cdot s \cdot n^l$? What we implicitly assume here is that $S = N \otimes N$, which means that the composition of a transitive verb with a specific subject-object pair will result in a matrix. In fact, this is exactly the composition model proposed in [40] and [38], where the meaning of a sentence is represented as an order-n tensor, with n to be the number of arguments for the head word of the sentence. In other words, when following this approach an intransitive sentence lives in a space $S = N$, a transitive one in $S = N \otimes N$ and so on. Let us show this graphically; in our diagrammatic calculus, the inflated tensor is depicted by the left-hand diagram below. When substituting the verb in Def. 3.3.1, the composition proceeds as in the right-hand diagram:



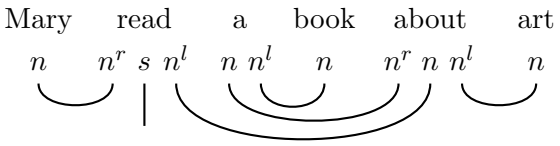
(4.12)

Verb Sentence

It is immediately obvious that the involvement of Δ-maps in the creation of the verb tensor imposes a restriction: since now our sentence space is produced by copying basis elements of the noun space, our functor $\mathcal{F}$ cannot any more apply different mappings on the two atomic pregroup types s and n; both of these should be mapped onto the same basic vector space, bringing the model closer to vector mixtures than originally aimed at by the theory. Indeed, it is easy to show that in the extreme case that one copies *all* the dimensions of a relational tensor, as we do here, the model reduces itself to a simple point-wise multiplication of the arguments with the verb tensor itself. This is exactly what diagram (4.12) reflects, and it is also obvious from the closed-form formula produced by the composition: Let $\overrightarrow{subj} = \sum_i s_i \overrightarrow{n_i}$, $\overrightarrow{obj} = \sum_j o_j \overrightarrow{n_j}$ be the vectors for the context of a transitive verb, the initial matrix of which is given by $\overline{verb} = \sum_{ij} v_{ij} \overrightarrow{n_i} \otimes \overrightarrow{n_j}$. Taking $\overline{verb}\,' = (\Delta_N \otimes \Delta_N)(\overline{verb}) = \sum_{ij} v_{ij} \overrightarrow{n_i} \otimes \overrightarrow{n_i} \otimes \overrightarrow{n_j} \otimes \overrightarrow{n_j}$ leads to the following composition:

$$\begin{aligned} (\epsilon_N^r \otimes 1_{N\otimes N} \otimes \epsilon_N^l)(\overrightarrow{subj} \otimes \overline{verb}\,' \otimes \overrightarrow{obj}) &= \\ \sum_{ij} v_{ij} \langle \overrightarrow{subj} | \overrightarrow{n_i} \rangle \langle \overrightarrow{n_j} | \overrightarrow{obj} \rangle \overrightarrow{n_i} \otimes \overrightarrow{n_j} &= \\ \sum_{ij} v_{ij} s_i o_j \overrightarrow{n_i} \otimes \overrightarrow{n_j} = \overline{verb} \odot (\overrightarrow{subj} \otimes \overrightarrow{obj}) & \end{aligned} \tag{4.13}$$

Hence, what we actually get is a sophisticated version of a vector mixture model (although not exactly in the word order forgetting sense of it), where the representation of a sentence is always the tensor product of the arguments point-wise multiplied by the tensor of the head verb. Which now makes very clear the reason behind the other unwelcome property of this formulation, that every sentence has to live in a space of a different tensor power, equal to that of the head verb. Especially this latter side-effect is in direct conflict with the premises of the categorical framework, since there is no way for one to compare sentences of different structures, say an intransitive one with a transitive one. Even more importantly, it means that the model is unable to assign a meaning to sentences or phrases with nested grammatical structures, such as in the following case:



(4.14)

Due to the mismatch between the logical and the concrete types, the translation of the above derivation to vector spaces is not possible; the multi-linear map for 'read' expects a vector in N as a semantic representation for its object, but what it actually receives is a tensor of order 2 (a matrix), since preposition 'about' is a function of two arguments. The next section deals with all the above issues, building on and extending my previously published work with Sadrzadeh, Coecke and Pulman [51, 53] on this topic.

4.3 Unifying the sentence space

The remedy for the problems raised with the approach of [40] is simple and stems directly from the two main limitations we have to take into account: (a) S must be equal to N; (b) a tensor with n arguments will always return a vector in N iff its order is $n+1$. Let me enforce the first condition by modifying the syntax-to-semantics functor so that the assignments of the basic grammar types in Eq. 3.19 are replaced with the following:

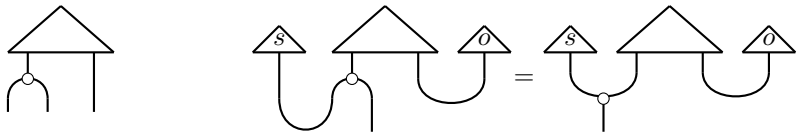
$$\mathcal{F}(n) = N \qquad \mathcal{F}(s) = N \tag{4.15}$$

Since we start from reduced tensors, the order of which is equal to the number of the verb arguments, we can conform to the second condition and properly restore the

functorial passage from grammar to vector spaces by restricting ourselves in copying only *one* of the tensor dimensions. As we will see later, this leads to interesting consequences that allow intuitive linguistic interpretations of the resulting framework.

I will continue using the case of a transitive verb as a running example. We start from a matrix created as dictated by Eq. 4.9; this is a tensor living in $N \otimes N$ that has to be encoded in $N \otimes N \otimes N$. A transitive verb is a function of two arguments, so there are two different ways to apply the Frobenius Δ-map:

Copy-Subject The first option is to copy the "row" dimension of the matrix corresponding to the verb. This dimension is the one that interacts with the subject noun during the composition. In diagrammatic form, the tensor and the composition are as follows:



(4.16)

Verb Sentence

The compositional morphism in this case is the following:

$$(\epsilon^r_N \otimes 1_N \otimes \epsilon^l_N) \circ (1_N \otimes \Delta_N \otimes 1_N \otimes 1_N)(\overrightarrow{subj} \otimes \overline{verb} \otimes \overrightarrow{obj}) = (\mu_N \otimes \epsilon^l_N)(\overrightarrow{subj} \otimes \overline{verb} \otimes \overrightarrow{obj}) \tag{4.17}$$

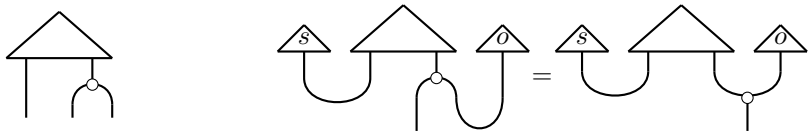
as the normal form in (4.16) and the commutative diagram below makes clear:

$$\begin{array}{ccccc} I \otimes I \otimes I & \xrightarrow{\overrightarrow{s} \otimes \overline{v} \otimes \overrightarrow{o}} & N \otimes N \otimes N \otimes N & \xrightarrow{1_N \otimes \Delta_N \otimes 1_N \otimes 1_N} & N \otimes N \otimes N \otimes N \otimes N \\ & & & \searrow^{\mu_N \otimes \epsilon^l_N} & \downarrow{\scriptstyle \epsilon^r_N \otimes 1_N \otimes \epsilon^l_N} \\ & & & & N \end{array} \tag{4.18}$$

Linear-algebraically (and using the subject, verb, and object vectors of our previous example), the computation proceeds as follows:

$$
\begin{aligned}
(\mu_N \otimes \epsilon_N^l)(\overrightarrow{subj} \otimes \overline{verb} \otimes \overrightarrow{obj}) = \sum_{ij} v_{ij} s_i o_j \mu_N(\overrightarrow{n_i} \otimes \overrightarrow{n_i}) \epsilon_N^l(\overrightarrow{n_j} \otimes \overrightarrow{n_j}) \quad &= \\
\sum_{ij} v_{ij} s_i o_j \overrightarrow{n_i} \langle \overrightarrow{n_j} | \overrightarrow{n_j} \rangle = \sum_{ij} v_{ij} s_i o_j \overrightarrow{n_i} \quad &= \\
\overrightarrow{subj} \odot (\overline{verb} \times \overrightarrow{obj})
\end{aligned}
\tag{4.19}
$$

Copy-Object Our other option is to copy the "column" dimension of the verb matrix, the one that interacts with objects:

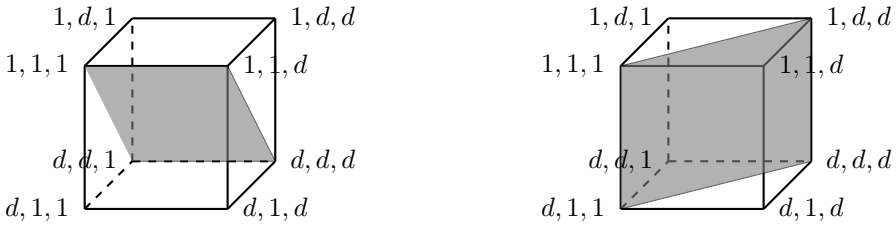


Verb Sentence (4.20)

Eq. 4.21 provides the categorical morphism and the linear-algebraic form (which I am not going to derive explicitly this time):

$$
\begin{aligned}
(\epsilon_N^r \otimes 1_N \otimes \epsilon_N^l) \circ (1_N \otimes 1_N \otimes \Delta_N \otimes 1_N)(\overrightarrow{subj} \otimes \overline{verb} \otimes \overrightarrow{obj}) \quad &= \\
(\epsilon_N^r \otimes \mu_N)(\overrightarrow{subj} \otimes \overline{verb} \otimes \overrightarrow{obj}) \quad &= \\
\overrightarrow{obj} \odot (\overline{verb}^{\mathsf{T}} \times \overrightarrow{subj})
\end{aligned}
\tag{4.21}
$$

From a geometric perspective, the two Frobenius models correspond to different ways of "diagonally" placing a plane into a cube. This is shown in Fig. 4.1.

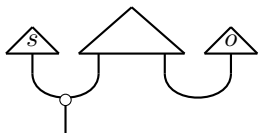


Copy subject Copy object

Figure 4.1: Geometric interpretation of the Frobenius models.

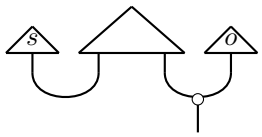
In contrast to the original model of [40], the interpretation of the Copy-Subject and Copy-Object models is quite different. Although there is still an element of point-

wise multiplication, now part of the sentence is computed by tensor contraction. For the case of the Copy-Subject model, the composite part corresponds to the verb phrase (verb-object) and is produced by matrix-multiplying the verb matrix with the object vector. In other words, what actually is achieved is a merging of the two parts of the sentence (subject and verb phrase), by point-wise multiplying the vector of the subject with the composite vector of the verb phrase.



(4.22)

The situation is very similar for the Copy-Object model, with the difference that now a vector is "categorically" composed for the intransitive part of the sentence (subject-verb), and then this vector is merged with the vector of the object:



(4.23)

The particular perspective of breaking the sentence into two parts with the verb as a common joint provides another important advantage: it actually *justifies* the somewhat ad-hoc decision to use a matrix in the compositional equations, since in both the cases of a verb phrase and an intransitive sentence the verb is a function of a single argument—that is, it can be canonically represented as a tensor of order 2. As we will see, my analysis suggests that the matrix in the Copy-Subject model must be different from the matrix of the Copy-Object model, since these two tensors represent different functions. I will return to this topic in §4.8, when I will discuss statistical techniques for building the relational tensors. Furthermore, in §4.5 we will see that the proposed model finds an appealing interpretation related to the linguistic phenomenon known as intonation. For the moment, though, I will show how the Frobenius setting can be extended to tensors of any arity.

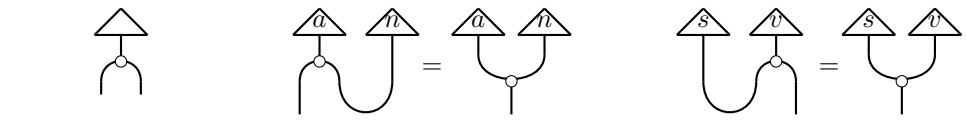
4.4 Extending the setting to any tensor

The application of Frobenius operators extends to any relational word, providing a generic treatment for all aspects of language. According to the "argument summing"

procedure of [40], for example, the linear maps of adjectives and intransitive verbs (both functions of a single argument) will be elements of N created as follows:

$$\overrightarrow{adj} = \sum_i \overrightarrow{noun_i} \qquad \overrightarrow{verb}_{IN} = \sum_i \overrightarrow{subj_i} \tag{4.24}$$

with i iterating through all relevant contexts. In both cases the application of a Δ operation will create a diagonal matrix, which when substituted in Def. 3.3.1 will produce the result shown below.



(4.25)

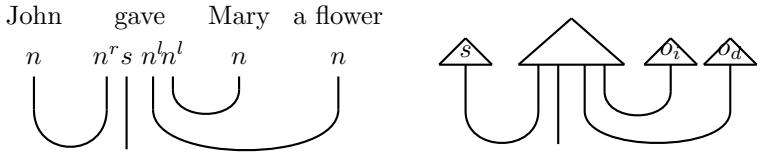
Adjective/Intr. verb Adjective-noun Intransitive sentence

Note that this is nothing more than another case of copying "all" dimensions of a tensor (here just one), that as we discussed in §4.3 results in a degeneration to an element-wise multiplicative model. Specifically, the meaning of adjective-noun compounds and intransitive sentences reduces to the following:

$$\overrightarrow{adj\ noun} = \mu(\overrightarrow{adj} \otimes \overrightarrow{noun}) = \overrightarrow{adj} \odot \overrightarrow{noun} \tag{4.26}$$

$$\overrightarrow{s_{IN}} = \mu(\overrightarrow{subj} \otimes \overrightarrow{verb}) = \overrightarrow{subj} \odot \overrightarrow{verb} \tag{4.27}$$

I will now proceed to the more interesting case of a ditransitive sentence, the derivation of which is depicted here:

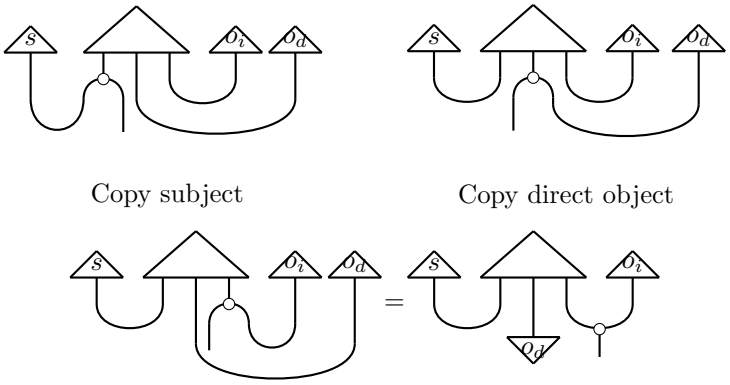


(4.28)

A ditransitive verb can be seen as a function of 3 arguments; hence, we can start by creating a tensor of order 3 that represents the verb, as follows:

$$\overrightarrow{verb}_{DT} = \sum_i \overrightarrow{subj_i} \otimes \overrightarrow{iobj_i} \otimes \overrightarrow{dobj_i} \tag{4.29}$$

where $\overrightarrow{iobj}$ and $\overrightarrow{dobj}$ refer to the vectors of the indirect and direct object, respectively. In this case, the Frobenius operators offer to us three alternatives, all of which are shown below:



(4.30)

Copy indirect object

Note that the third case (copy indirect object) makes use of the symmetry of the category. In all cases the result is a vector, computed as below:

$$\text{Copy subject: } \overrightarrow{s_{DT}} = \overrightarrow{subj} \odot \left((\overline{verb} \times \overrightarrow{iobj}) \times \overrightarrow{dobj} \right) \tag{4.31}$$

$$\text{Copy direct object: } \overrightarrow{s_{DT}} = \overrightarrow{dobj} \odot \left((\overline{verb} \times \overrightarrow{iobj})^{\mathsf{T}} \times \overrightarrow{subj} \right) \tag{4.32}$$

$$\text{Copy indirect object: } \overrightarrow{s_{DT}} = \overrightarrow{iobj} \odot \left((\overline{verb} \times \overrightarrow{dobj})^{\mathsf{T}} \times \overrightarrow{subj} \right) \tag{4.33}$$

As in the transitive case, the sentence is broken into two parts: one part for the copied argument and one part for the rest of the sentence. In all cases, the argument vector is merged by point-wise multiplication with the composite vector of the context. Let me now formally define the concept of a *Frobenius vector*:

Definition 4.4.1. Let a text constituent with head word w of arity n and grammatical derivation α; the ith Frobenius vector of this constituent is defined as follows:

$$\overrightarrow{v}_i^{\mathsf{F}} = \mathcal{F}(\alpha)(\overrightarrow{arg_1} \otimes \ldots \otimes \Delta_i(\overline{w}) \otimes \ldots \otimes \overrightarrow{arg_n}) \tag{4.34}$$

where $\Delta_i(\overline{w})$ represents the application of a Frobenius Δ-map on the ith dimension of the reduced tensor $\overline{w}$ for that word.

If w is a verb, then the Frobenius vectors $\overrightarrow{v}_i^{\mathsf{F}}$ will represent different aspects of the meaning of a sentence. In the next section I will provide an intuitive interpretation of these different aspects in the context of intonation.

4.5 Modelling intonation

The term *intonation* refers to variations of spoken pitch, the purpose of which is to emphasize parts of the utterance that might be important for the conveyed message. Consider again the ditransitive sentence of §4.4:

(3) John gave Mary a flower

This statement might have been used in response to a number of questions, each one of which requires the use of a different intonational emphasis from the speaker. For example:

(4) a. *Who gave Mary a flower?*
John gave Mary a flower

b. *What did John give to Mary?*
John gave Mary **a flower**

c. *To whom did John give a flower?*
John gave **Mary** a flower

The bold-faced part above is what Steedman [115] calls *rheme*—the information that the speaker wishes to make common ground for the listener. The rest of the sentence, i.e. what the listener already knows, is called *theme*. In the same paper, Steedman argues that the different syntactic derivations that one can get even for very simple sentences when using Combinatory Categorial Grammar (CCG) [114] (referred to with the somewhat belittling term "spurious readings"), actually serve to reflect changes in the intonational patterns. In CCG, our example sentence in (3) has a number of different syntactic derivations, two of them are the following:

$$\dfrac{\dfrac{\dfrac{\text{John}}{\text{NP}} \quad \dfrac{\dfrac{\text{gave}}{((\text{S}\backslash\text{NP})/\text{NP})/\text{NP}} \quad \dfrac{\text{Mary}}{\text{NP}}}{(\text{S}\backslash\text{NP})/\text{NP}}>}{\text{S}\backslash\text{NP}}> \quad \dfrac{\text{a flower}}{\text{NP}}}{\text{S}}< \tag{4.35}$$

$$\dfrac{\dfrac{\dfrac{\text{John}}{\text{NP}}}{\text{S}/(\text{S}\backslash\text{NP})}\,{>}\mathbf{T} \quad \dfrac{\dfrac{\text{gave}}{((\text{S}\backslash\text{NP})/\text{NP})/\text{NP}} \quad \dfrac{\text{Mary}}{\text{NP}}}{(\text{S}\backslash\text{NP})/\text{NP}}\,{>}}{\text{S}/\text{NP}}\,{>}\mathbf{B} \quad \dfrac{\text{a flower}}{\text{NP}}}{\text{S}}\,{>} \tag{4.36}$$

Note that (4.35) proceeds by first composing the part of the verb phrase ("gave Mary a flower"); later, in the final step, the verb phrase is composed with the subject 'John'. The situation is reversed for (4.36), where the use of type-raising and composition rules of CCG allow the construction of the fragment "John gave Mary" as valid grammatical text constituent, which is later combined with the direct object of the sentence ('a flower'). According to Steedman, each one of these derivations expresses a different intonational pattern, distinguishing the rheme from the theme when the sentence is used for answering different questions: (4.35) answers to "Who gave Mary a flower?", whereas (4.36) to "What did John give to Mary?".

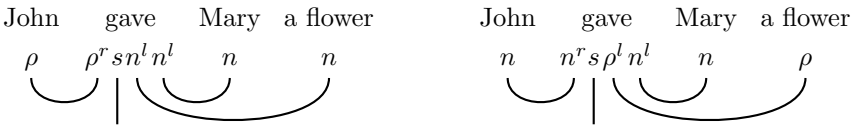
It is interesting to note that in our grammatical formalism, pregroup grammars, variations in a grammatical derivation similar to above are only implicitly assumed, since the order of composition between a relational word and its argument remains unspecified. This fact is apparent in the pregroup derivation of our sentence in (4.28), and is directly reflected in our semantic space through the functorial passage:

$$\begin{aligned} \overrightarrow{john}^{\mathsf{T}} \times \left((\overline{gave} \times \overrightarrow{mary}) \times \overrightarrow{flower} \right) &= \\ \left((\overrightarrow{john}^{\mathsf{T}} \times \overline{gave}) \times \overrightarrow{mary} \right) \times \overrightarrow{flower} &= \\ \left((\overrightarrow{john}^{\mathsf{T}} \times \overline{gave})^{\mathsf{T}} \times \overrightarrow{flower} \right) \times \overrightarrow{mary} & \end{aligned} \tag{4.37}$$

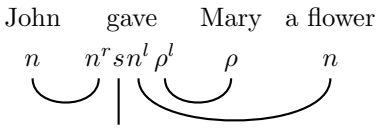
Eq. 4.37 constitutes a natural manifestation of the *principle of combinatory transparency* [114]: no matter in what order the various text constituents are combined, the semantic representation assigned to the sentence is always the same. However, if the claim that different grammatical derivations (thus different ways of combining constituents) subsume different intonational patterns is true, then this must also be reflected somehow in the semantic form of the sentence; in our case, the sentence vector. In other words, the claim here is that the meaning of sentence "**Mary** wrote a book about bats" is slightly different of that of sentence "Mary wrote a book about

bats". Indeed, if one uses the former form instead of the latter to answer the question "I *know* that Mary wrote a book, but what was this book about?", the person who posed the question would require a moment of mental processing to translate the semantics of the response in her mind.

Let me try making this idea more concrete by extending our pregroup grammar in order to reflect intonational information; specifically, I will introduce a new atomic type $\rho \leq n$ which will mark the part of a sentence that corresponds to rheme. The different intonational patterns in our example sentence now get the following distinct derivations:

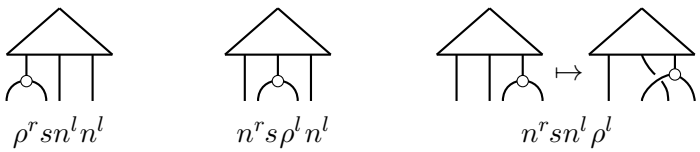


Who gave Mary a flower? What did John give to Mary? (4.38)



To whom did John give a flower?

Each intonational pattern requires a variation of the basic $n^r \cdot s \cdot n^l \cdot n^l$ grammatical type which, when composed with the rest of the context, leads to slightly different versions of the final result. The Frobenius setting detailed in this chapter provides a natural way to incorporate this kind of variation in the meaning of the relational tensors, and in turn in the meaning of sentences. Recall that each one of the alternative options for copying a dimension of a tensor results in (a) splitting the sentence in two parts, one of which consists of a single argument (the one corresponding to the copied dimension) and one for the rest of the sentence; (b) composing a vector for the rest of the sentence; and (c) point-wise multiplying the vector of the argument with the composite vector. I argue that the isolated argument corresponds to the rheme, while the composite part serves as the theme. In other words, each one of the types of 'gave' in (4.38) must be assigned to a tensor produced by copying a different argument, as below:



(4.39)

The point-wise multiplication of the theme with the rheme provides a unified meaning for the sentence, in which however the rheme will play a very important role: the nature of the compositional operator (a vector mixture) guarantees that the resulting vector will be affected *equally* from both arguments, rheme and theme. This puts the necessary focus on the appropriate part of the sentence, reflecting the variation in semantics intended by the intonational pattern (see also Diagram 4.30):

(5) a. Who gave Mary a flower?
$\overrightarrow{john} \odot \overrightarrow{gave\ mary\ a\ flower}$

b. What did John give to Mary?
$\overrightarrow{john\ gave\ mary} \odot \overrightarrow{a\ flower}$

c. To whom did John give a flower?
$\overrightarrow{john\ gave\ a\ flower} \odot \overrightarrow{(to)\ mary}$

The following definition makes this idea precise:

Definition 4.5.1. The meaning of a text constituent with head word w and arity n carrying intonational information on the ith argument is given by its ith Frobenius vector.

There is a different kind of question that has not been addressed yet:

(6) What just happened?
John gave Mary a flower

For simple declarative cases such that the above, where the theme is unmarked, I will define the meaning of a sentence as the sum of all intonational patterns. Since each intonational pattern corresponds to a different Frobenius vector, as these defined in Def. 4.4.1, we conclude to the following:

Definition 4.5.2. The meaning of a sentence with an unmarked theme is defined as the sum of all its Frobenius vectors:

$$\overrightarrow{s_U} = \overrightarrow{v}^{\mathsf{F}}_1 + \overrightarrow{v}^{\mathsf{F}}_2 \ldots + \overrightarrow{v}^{\mathsf{F}}_n \tag{4.40}$$

The material presented in this section is admittedly only the first step towards a complete treatment of intonation in the context of CDMs.[2] An interesting point that requires further study, for example, is how the above setting behaves in the presence of focus-sensitive elements, such as adverbs like 'only' and 'even'. Dealing

[2]A more extended version of this study can be found in [50] (joint work with M. Sadrzadeh).

with questions like these, together with a practical evaluation explicitly focused on the topic of intonation, is deferred for future research. For my current purposes, the model of Def. 4.5.2 (to which I will refer as **Frobenius additive**) and the other Frobenius models presented in this chapter will be evaluated on a number of generic tasks in Chapter 7. Furthermore, in the next section I provide a theoretical example of the Frobenius setting that hopefully will make the concept better understood to the reader.

4.6 A truth-theoretic instantiation

The purpose of this section is to provide some intuition for the Frobenius framework by showing how it behaves in a simple truth-theoretic setting similar to that of §4.1. Recall that a relation corresponding to the meaning of a transitive verb, such as 'likes', can be given by a matrix $\overline{likes} = \sum likes_{ij} \overrightarrow{n_i} \otimes \overrightarrow{n_j}$ where $likes_{ij}$ is the scalar 1 if individual n_i likes individual n_j and 0 otherwise. For individuals n_1 and n_3, copying the subject dimension of this matrix proceeds as follows:

$$\begin{aligned} (\mu_N \otimes \epsilon_N^l)\left(\overrightarrow{n_1} \otimes \left(\sum_{ij} likes_{ij} \overrightarrow{n_i} \otimes \overrightarrow{n_j}\right) \otimes \overrightarrow{n_3}\right) &= \\ \sum_{ij} likes_{ij} \mu(\overrightarrow{n_1} \otimes \overrightarrow{n_i}) \langle \overrightarrow{n_j} | \overrightarrow{n_3} \rangle = \sum_{ij} likes_{ij} \delta_{1i} \delta_{j3} \overrightarrow{n_i} &= \\ likes_{13} \overrightarrow{n_1} = \begin{cases} \overrightarrow{n_1} & \text{if } likes_{13} = 1 \\ \overrightarrow{0} & o.w. \end{cases} & \end{aligned} \tag{4.41}$$

The above example illustrates clearly the necessity of a shared space between sentences and words when using Frobenius operators, since what we get back is a vector in N; in case that the sentence is true, this vector corresponds to the subject of the verb, otherwise it is the zero vector. However, how can one interpret the result of (4.41) in a way consistent with the model? Let us repeat the derivation step-wise, starting from the composition of the verb with the object:

$$\begin{aligned} (1_N \otimes \epsilon_N^l)\left(\left(\sum_{ij} likes_{ij} \overrightarrow{n_i} \otimes \overrightarrow{n_j}\right) \otimes \overrightarrow{n_3}\right) = \sum_{ij} likes_{ij} \overrightarrow{n_i} \langle \overrightarrow{n_j} | \overrightarrow{n_3} \rangle &= \\ \sum_{ij} likes_{ij} \overrightarrow{n_i} \delta_{j3} = \sum_i likes_{i3} \overrightarrow{n_i} & \end{aligned} \tag{4.42}$$

The vectorial representation of the verb phrase "likes n_3" gets a very intuitive

explanation: it is the sum of all individuals who like the person denoted by vector $\overrightarrow{n_3}$. Since in our setting individuals form an orthonormal basis, the ith element of this vector corresponds to the person n_i, and has value 1 if this person likes n_3 or 0 otherwise. Indeed, as we discussed in §4.1 this sum can be seen as a *subset* of elements in our universe of discourse, the entirety of which is represented by the vector $\sum_i \overrightarrow{n_i}$.

What differentiates the Frobenius setting from the standard categorical composition is the second step, in our case that of composing the meaning of the verb phrase with the subject. Recall that this is just the point-wise multiplication of the vector of the verb phrase with the vector of the subject, which in our setting can be interpreted as the *intersection* of the singleton set formed by the subject with the set of people who like individual n_3.

$$\overrightarrow{n_1} \odot \overrightarrow{likes\ n_3} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} \odot \begin{pmatrix} 1 \\ 1 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} \tag{4.43}$$

This is just a test of set membership: if the result is the subject vector, this means that the corresponding individual is a member of the set of people who like n_3; in the opposite case, the result will be the zero vector (the empty set). Let me summarize these observations using as an example the concrete sentence "John likes Mary". In the traditional categorical composition, the meaning of this sentence is given by:

$$\text{John} \ \in \ \{x | x \text{ likes Mary}\} \tag{4.44}$$

and the result can be one of the two values in our sentence space, i.e. $\top$ or $\bot$. In the Frobenius setting, the meaning of the sentence changes to:

$$\{\text{John}\} \cap \{x | x \text{ likes Mary}\} \tag{4.45}$$

and the result is {John} if John is among the people who like Mary, or the empty set otherwise. Hence, while the goal is still the same (to state the truth or falseness of the sentence), now we have a different way to express the result.

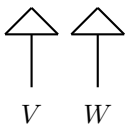
Even more interestingly, the above analysis provides a direct justification of why the Frobenius models constitute an appropriate way of modelling intonation: when answering the question "Who likes Mary?", the correct answer is "John" and not "true" or "false". Furthermore, it is not difficult to show that when the object dimension of the verb matrix is copied, the evaluation of the sentence takes the following form:

$$\{\text{Mary}\} \cap \{x | x \text{ is liked by John}\} \tag{4.46}$$

A positive result ($\{\text{Mary}\}$) answers directly to the question "Whom John likes?", setting the focus on the appropriate part of the sentence.

4.7 Frobenius operators and entanglement

In this section I will shortly demonstrate the benefits of using Frobenius operators for constructing relational tensors with regard to *entanglement*. As in quantum mechanics, entanglement is a necessary requirement for tensor-based models to allow the unhindered flow of information between the different parts of the sentence. Recall that a word tensor living in vector space W is seen as a state of W. This state is *separable* if it can be expressed as the tensor product of two or more vectors. In our graphical calculus, these objects are depicted by the juxtaposition of two or more triangles:

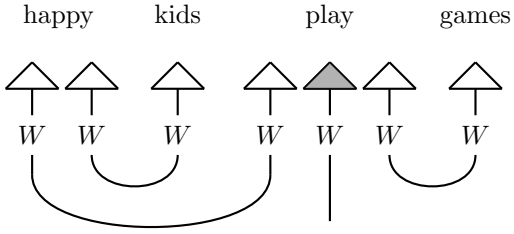


(4.47)

In general, a state is not separable if it is a linear combination of many separable states. The number of separable states needed to express the original tensor is equal to the *tensor rank*. Graphically, a tensor of this form is shown as a single triangle with two or more legs:

(4.48)

In categorical quantum mechanics terms, entangled states are necessary to allow the flow of information between the different subsystems; this is exactly the case for linguistics as well. Consider the diagram below, in which all relational words are represented by separable tensors (in other words, no entanglement is present).



(4.49)

In this version, the ϵ-maps are completely detached from the components of the relational tensors that carry the results (left-hand wire of the adjective and middle wire of the verb); as a consequence, flow of information is obstructed, all compositional interactions have been eliminated, and the meaning of the sentence is reduced to the middle component of the verb (shaded vector) multiplied by a scalar, as follows (superscripts denote the left-hand, middle, and right-hand components of separable tensors):

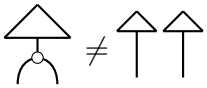
$$\langle \overrightarrow{happy}^{(r)} | \overrightarrow{kids} \rangle \langle \overrightarrow{happy}^{(l)} | \overrightarrow{play}^{(l)} \rangle \langle \overrightarrow{play}^{(r)} | \overrightarrow{games} \rangle \overrightarrow{play}^{(m)}$$

Depending on how one measures the distance between two sentences, this is a very unwelcome effect, to say the least. When using cosine distance, the meaning of all sentences with 'play' as the verb will be exactly the same and equal to the middle component of the 'play' tensor. For example, the sentence "trembling shadows play hide-and-seek" will have the same meaning as our example sentence. Similarly, the comparison of two arbitrary transitive sentences will be reduced to comparing just the middle components of their verb tensors, completely ignoring any surrounding context. The use of Euclidean distance instead of cosine would slightly improve things, since now we would be at least able to also detect differences in the magnitude between the two middle components. Unfortunately, this metric has been proved not very appropriate for distributional models of meaning, since in the vastness of a highly dimensional space every point ends up to be almost equidistant from all the others. As a result, most implementations of distributional models prefer the more relaxed metric of cosine distance which is length-invariant. Table 4.1 presents the consequences of separability in a number of grammatical constructs.

Structure	**Simplification**	**Cos-measured**
adjective-noun	$\overrightarrow{adj} \times \overrightarrow{noun} = (\overrightarrow{adj}^{(l)} \otimes \overrightarrow{adj}^{(r)}) \times \overrightarrow{noun} = \langle \overrightarrow{adj}^{(r)} \vert \overrightarrow{noun} \rangle \cdot \overrightarrow{adj}^{(l)}$	$\overrightarrow{adj}^{(l)}$
intrans. sentence	$\overrightarrow{subj} \times \overrightarrow{verb} = \overrightarrow{subj} \times (\overrightarrow{verb}^{(l)} \otimes \overrightarrow{verb}^{(r)}) = \langle \overrightarrow{subj} \vert \overrightarrow{verb}^{(l)} \rangle \cdot \overrightarrow{verb}^{(r)}$	$\overrightarrow{verb}^{(r)}$
verb-object	$\overrightarrow{verb} \times \overrightarrow{obj} = (\overrightarrow{verb}^{(l)} \otimes \overrightarrow{verb}^{(r)}) \times \overrightarrow{obj} = \langle \overrightarrow{verb}^{(r)} \vert \overrightarrow{obj} \rangle \cdot \overrightarrow{verb}^{(l)}$	$\overrightarrow{verb}^{(l)}$
transitive sentence	$\overrightarrow{subj} \times \overrightarrow{verb} \times \overrightarrow{obj} = \overrightarrow{subj} \times (\overrightarrow{verb}^{(l)} \otimes \overrightarrow{verb}^{(m)} \otimes \overrightarrow{verb}^{(r)}) \times \overrightarrow{obj} = \langle \overrightarrow{subj} \vert \overrightarrow{verb}^{(l)} \rangle \cdot \langle \overrightarrow{verb}^{(r)} \vert \overrightarrow{obj} \rangle \cdot \overrightarrow{verb}^{(m)}$	$\overrightarrow{verb}^{(m)}$

Table 4.1: Consequences of separability in various grammatical structures. Superscripts (l), (m) and (r) refer to left-hand, middle, and right-hand component of a separable tensor.

This aspect of tensor-based models of meaning is quite important and, surprisingly, almost completely neglected by the current research. In fact, the only relevant work I am aware of comes from personal research (joint with Sadrzadeh) [49]. In the context of current discussion, the interesting point is that no matter the actual level of entanglement in the relational tensors, the Frobenius framework detailed in this chapter provides an additional layer that prevents the compositional process from degrading to the invariant vectors showed in (4.49). The reason behind this is that, given a vector $\overrightarrow{v}$, the diagonal matrix that results from $\Delta(\overrightarrow{v})$ cannot be expressed as a product state:

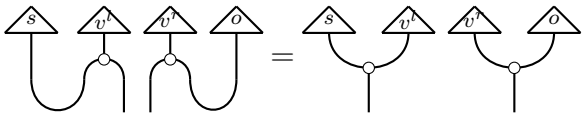


(4.50)

A tensor like this is essentially constructed from a *single* wire, providing an analogy with a *maximally entangled state* in quantum mechanics. I give an example of this inseparability for a two dimensional space, where we have:

$$\Delta \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} a & 0 \\ 0 & b \end{pmatrix} \quad \neq \quad \begin{pmatrix} v_1 \\ v_2 \end{pmatrix} \otimes \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} = \begin{pmatrix} v_1 w_1 & v_1 w_2 \\ v_2 w_1 & v_2 w_2 \end{pmatrix} \tag{4.51}$$

since the only case in which the above would be trivially satisfied is when $a = b = 0$. Let us now see how this property affects the various models presented in this chapter in the case of a separable verb matrix. When one copies both dimensions of the original verb matrix the result is the following:

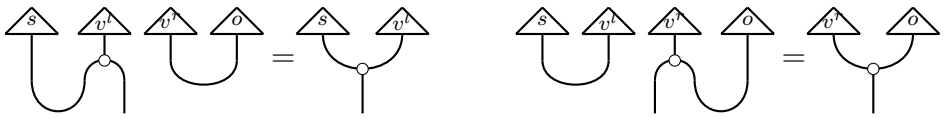


(4.52)

which means that, linear-algebraically, the meaning of a transitive sentence becomes:

$$\overrightarrow{subj\ verb\ obj} = (\overrightarrow{subj} \odot \overrightarrow{verb}^{(l)}) \otimes (\overrightarrow{verb}^{(r)} \odot \overrightarrow{obj}) \tag{4.53}$$

Furthermore, the Copy-Subject and Copy-Object models simplify to the following:



(4.54)

Therefore, the actual equation behind the Frobenius additive model becomes:

$$\overrightarrow{subj\ verb\ obj} = (\overrightarrow{subj} \odot \overrightarrow{verb}^{(l)}) + (\overrightarrow{verb}^{(r)} \odot \overrightarrow{obj}) \tag{4.55}$$

Despite the simplifications presented above, note that none of these models degenerates to the level of producing "constant" vectors or matrices. The reason behind this lies in the use of Frobenius Δ operators for copying the original dimensions of the verb matrix, a computation that equips the fragmented system with flow, although not in the originally intended sense. Actually, the results of our aforementioned study in [49] suggest that this might be very important when one constructs verb matrices by using the argument summing procedure of [40]. In that work we empirically evaluate the level of entanglement in verb matrices created according to Eq. 4.9 by comparing them with two separable versions. In the first case, the verbs matrices were created by a straightforward variation of the original version that results in a product state:

$$\overline{verb} = \left(\sum_i \overrightarrow{subj_i} \right) \otimes \left(\sum_i \overrightarrow{obj_i} \right) \tag{4.56}$$

with i iterating as usual over all instances of the verb in the corpus. Furthermore, each verb matrix was compared with its rank-1 approximation, which is again a product state constructed by using only the highest eigenvalue and the related left and right singular vectors of the original matrix, as produced by applying singular value decomposition. In both cases, our findings suggest a very high *cosine* similarity (up to 0.97) between the original matrix and the separable version, probably due to a high level of linear dependence between our word vectors. Although such a result depends heavily on the form of vector space used for constructing the verb tensors, our discussion above shows clearly that the Frobenius framework offers a safety net that can help us avoid these kinds of pitfalls. I refer the interested reader to [49] for more details about this topic.

4.8 From argument summing to linear regression

Most of my analysis in this chapter assumes as a starting point the reduced verb representations produced by applying Eq. 4.9. I will now explore a different approach based on statistical learning that, as we are going to see, fits quite nicely to the Frobenius setting, and even extends its power. Recall that the Frobenius model essentially acts on two steps: (a) it prepares a composite vector for the theme by applying tensor contraction; and (b) point-wise multiplies the result with the rheme. When the rheme is the subject of a transitive verb, this is translated to the following:

$$\overrightarrow{subj} \odot (\overline{verb} \times \overrightarrow{obj}) \tag{4.57}$$

with the meaning of the verb-object part to be computed "categorically", as the matrix multiplication of the verb with the object vector.

Let me now attempt for a moment a twist of perspective, and forget any assumptions about using Frobenius operators. How can one describe the compositional model depicted in Eq. 4.57 in that case? This is obviously a vector mixture model, where the operands are always some rheme and its corresponding theme; furthermore, the vector of the theme has been prepared "categorically". However, there is an important difference between this new view and the Frobenius one we were discussing so far: note that now the notion of a well-defined sentence does not exist any more—actually, it has been replaced by the notion of a *well-defined theme*. Let me formalize this observation, by replacing type s with a new atomic type, call it θ. The pregroup derivation for the theme part of Eq. 4.57 becomes:

$$(\theta \cdot n^l) \cdot n = \theta \cdot (n^l \cdot n) \leq \theta \tag{4.58}$$

The new view reveals two important things. First, it is now evident why the representation of the verb as a tensor of order 2 (a matrix) makes sense in this setting: it is just a function that inputs a noun and outputs a well-defined theme, that is, $\overline{verb} : N \to \Theta$, which can canonically be expressed as a matrix in $N \otimes \Theta$. More importantly, computing the meaning of the theme that results in by copying the object dimension, in $(\overrightarrow{subj}^{\mathsf{T}} \times \overline{verb}) \odot \overrightarrow{obj}$, will naturally require a different version of $\overline{verb}$, since this time the output of our function is a syntactically different construct (subject-verb).

Assuming now we have a means to prepare these two different aspects of our verb tensor, we can think of a transitive verb as being represented by a tuple $(\overline{verb}_s, \overline{verb}_o)$. This form has the advantage that, while it retains the benefits of our previous reduced representation (we still do not need to create a verb tensor of order 3), it is more expressive since the interaction of the verb with each argument is handled by a specialized function. Using this perspective in our Frobenius additive model results in the following equation:

$$\overrightarrow{subj\ verb\ obj} = \overrightarrow{subj} \odot (\overline{verb}_o \times \overrightarrow{obj}) + (\overrightarrow{subj}^{\mathsf{T}} \times \overline{verb}_s) \odot \overrightarrow{obj} \tag{4.59}$$

At this stage, the separation of a transitive sentence into two distinct themes by the Frobenius operators makes it easier for us to apply more sophisticated statistical methods in order to create the corresponding verb tensors. Imagine for example the case of the left-hand theme (subject-verb); what we want is a linear map $\overline{verb}_s$ that,

given an input noun, will approximate some ideal distributional behaviour of the subject-verb construct in a large corpus. Assuming that, for the ith occurrence of the verb in the corpus, this "ideal behaviour" is denoted by a vector $\overrightarrow{subj_i\ verb}$, with $subj_i$ referring to the corresponding subject noun, the matrix we seek is the following:

$$\hat{\overrightarrow{verb}}_s = \arg\min_{\mathbf{W}} \sum_i \left(\mathbf{W} \times \overrightarrow{subj_i} - \overrightarrow{subj_i\ verb} \right)^2 \tag{4.60}$$

Learning the matrix $\mathbf{W}$ that minimizes the sum in Eq. 4.60 is a *multi-linear regression* problem, a technique that obviously constitutes the most appropriate way for training a model based on linear and multi-linear maps. What remains to be answered is how one should create the vectors that will be used as a "gold reference" in the optimization process. In one of the first applications of linear regression for CDMs, Baroni and Zamparelli [6] propose the creation of a distributional vector for every two-word construct using the same techniques as for single words (i.e. by collecting word co-occurrences from the context). The goal of the model, then, would be to create a matrix that when multiplied with a subject vector will approximate the corresponding co-occurrence vector of the subject-verb construct. Of course, exactly the same method can be applied on the right-hand theme of a transitive sentence, learning a verb matrix that produces as output a verb-object construct.

Interestingly, if we discard the point-wise multiplications in Eq. 4.59, what we get back is very close to a recent trend in constructing tensor-based compositional models where each argument of a multi-linear map is addressed separately by learning a specialized matrix [85, 93]. These models follow a methodology very similar to what described in this section, in the sense that the composition function does not aim to produce a sentence, but a specific part of it; the individual parts are somehow combined (usually by summing their vectors) in a later stage to produce a vectorial representation for the whole sentence.

Despite the convenience that comes with such methods, the decoupling of the arguments in the above way hides an important caveat: although the objective function of Eq. 4.59 guarantees that the result will be a reliable (to the extent that the training corpus allows) map which, given an input, will produce an appropriate meaning vector for the left-hand (or right-hand) part of the sentence, there is no guarantee whatsoever that the addition of these two vectors reflects appropriately the meaning of the sentence in its entirety. The solution to this problem is to restore our initial perspective and start again working with sentences instead of themes. From a statistical point of view, the proper way to achieve that is to *jointly* learn the two matrices

of a transitive verb; in the case of the Frobenius additive model, this could be done by training a regression model that directly minimizes the following quantity:

$$\sum_i \Big((\mathbf{W}_s \times \overrightarrow{subj_i}) \odot \overrightarrow{obj_i} + (\mathbf{W}_o \times \overrightarrow{obj_i}) \odot \overrightarrow{subj_i} - \overrightarrow{subj_i\ verb\ obj_i} \Big)^2 \tag{4.61}$$

or, for the simpler version without the point-wise interaction:

$$\sum_i \Big(\mathbf{W}_s \times \overrightarrow{subj_i} + \mathbf{W}_o \times \overrightarrow{obj_i} - \overrightarrow{subj_i\ verb\ obj_i} \Big)^2 \tag{4.62}$$

In the above equations, $\overrightarrow{subj\ verb\ obj}$ refers to a distributional vector created from the contexts of the specific triplet across the corpus. Inevitably, this might lead to data sparsity problems when creating the vectors for the triplets, and certainly is not applicable for training argument matrices in cases of relational words of higher arity (e.g. ditranstive verbs). For these cases, we have to adopt the less generic solution to optimize the objective function directly on the goal set by the underlying task—for example, on some form of classification as in the case of sentiment analysis.

The application of the linear regression method can be seen in practice in §8.5, where I use Eq. 4.60 in order to train matrices for verbs taking one argument.

4.9 A revised taxonomy of CDMs

The Frobenius framework presented in this chapter has the unique characteristic that it allows the combination of two quite different compositional models: a vector mixture model based on point-wise multiplication, where both operands have equal contribution to the final composite result; and a tensor-based model in which relational words are tensors of higher order acting on vectors. As we saw in the previous pages, this flexibility can be used to model linguistic phenomena, such as intonation, or to provide solutions that lead to models more robust against physical properties of the verb tensors, as in the case of separability. In Chapter 5 I move one step further, showing that appropriate use of Frobenius operators can help us to reduce the space complexity imposed by the higher order tensors of functional words such as prepositions and conjunctions in a way that makes linguistic sense.

In general, the application of Frobenius algebras to language constitutes an interesting novel compositional methodology, resulting in a class of CDMs that deserves its own place in the taxonomy we put together in Fig. 2.7. I will now call this class

partial tensor-based models[3], and I will introduce it in Fig. 4.2 below. The practical evaluation of the new class of models is deferred for Chapter 7, where a series of experiments on various tasks will reveal steady and robust performance against previous implementations and baselines.

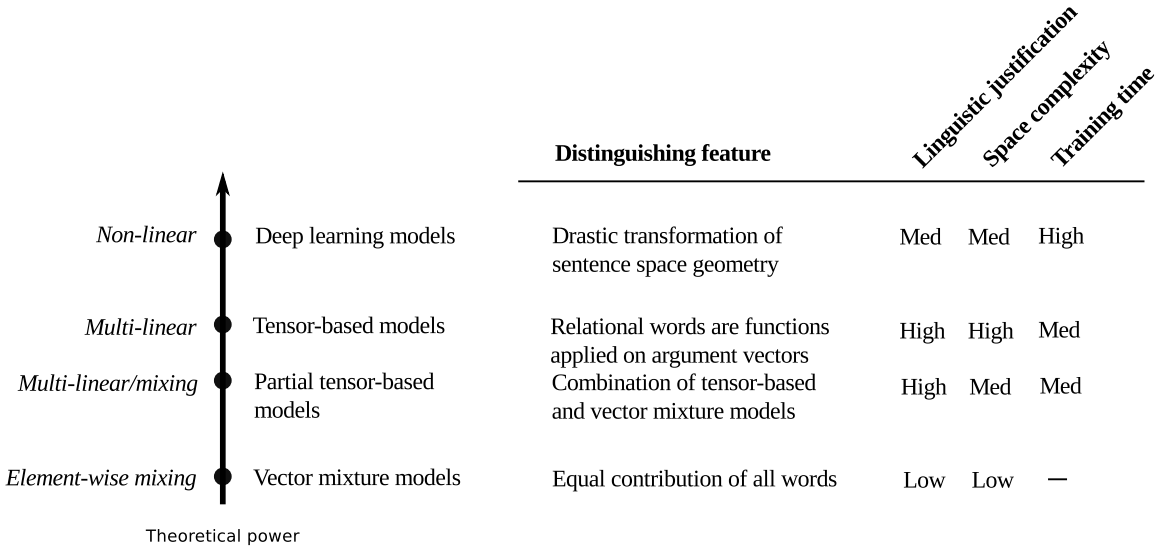


Figure 4.2: A revised taxonomy of CDMs.

[3]Thanks to a suggestion by Mehrnoosh Sadrzadeh.

Chapter 5

Covering Larger Fragments of Language

Chapter Abstract

In contrast to other classes of CDMs, a tensor-based model has to assign a form of meaning in every kind of functional word, such as prepositions and relative pronouns. In this chapter I use Frobenius operators in conjunction with unit maps in ***FVect*** *in order to provide intuitive linear-algebraic interpretations for a number of functional words, extending the applicability of the categorical framework of [25] to larger fragments of language.*

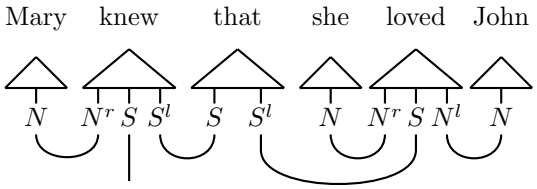
Functional words, such as prepositions, determiners, or relative pronouns, form a special category of lexical items that pose an important problem for any CDM. In the context of a distributional model, the problem arises from the ubiquitous nature of these words, which means that creating a context vector for preposition 'in', for example, would not be especially useful, since this word can be encountered in almost every possible context. Vector mixture models "address" this problem by considering functional words as semantically vacuous and ignoring them for any practical purpose. This solution seems questionable since it essentially discards specific relations between the various text constituents. At the other end of the scale, deep learning techniques usually rely on their brute force and make no distinction between functional words and content words, hoping that the learning process will eventually capture at least some of the correct semantics.

For tensor-based compositional models, the problem is even more complicated. Models like these, being based on deep semantic analysis, do not allow us to arbitrarily drop a number of words from a sentence based on subjective criteria regarding

their meaning, and for a good reason: there is simply no linguistic justification behind such decisions. Even in cases where dropping a word might seem justified (e.g. for a determiner), we need to proceed in a canonical way that does not violate the underlying theory. Even more importantly, the grammatical types of some functional words usually translate to tensors of very high order (for example, of order 6 for the case of verb-phrase coordinating conjunctions), a fact that makes the efficient treatment of the relevant constructs impossible. In this chapter I build on and extend previous work of Preller, Sadrzadeh and others [95, 103, 104] by exploiting the compact closed structure of the semantic space and the convenient tool of Frobenius algebras in order to address the above issues. Specifically, I provide canonical ways to create compact representations for various functional words, which result in compositions that reflect the essence hidden behind the complex grammatical interactions. As we will see, a valuable tool for achieving this will be a morphism of a compact closed category that we have not used until now: the η-map.

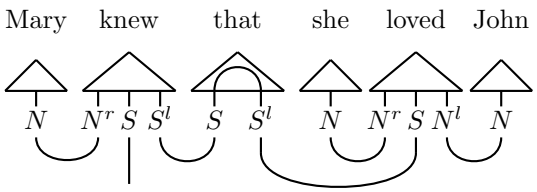
5.1 Omitting words

In English language there exist certain cases where one can omit a word without altering the meaning of the overall sentence. Imagine, for example, a typical case of a *sentential complement*:



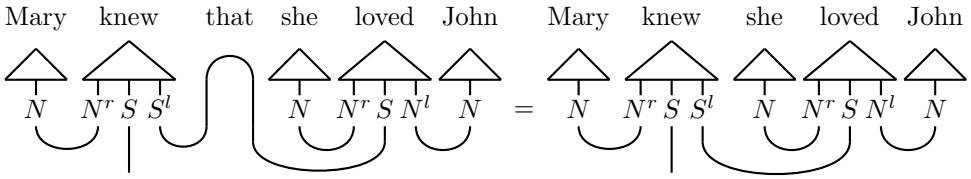
(5.1)

To comply with the English grammar, we would like the vector of the above sentence to be equal to that of "Mary knew she loved John", where the word 'that' has been omitted. As first showed by Preller and Sadrzadeh [95], compact closed categories provide a means to achieve this through the unit morphism η (Eq. 3.4). In the specific case, we can translate the $s \cdot s^l$ type of the complementizer to **FVect** by using a morphism $\eta_S^l : I \to S \otimes S^l$ in the following manner:



(5.2)

The interaction depicted above takes an intuitive form: the matrix of 'that' allows the information from the right-hand sentence to *flow* through it, acting as an identity function:

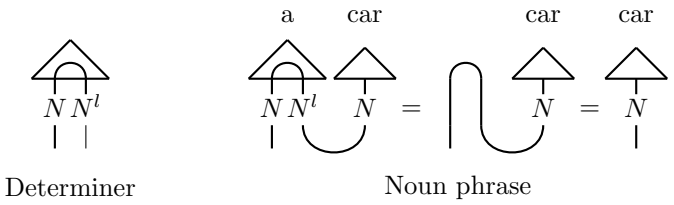


(5.3)

Indeed, in categorical terms the composition of 'that' with sentence "she loved John" becomes an instance of a yanking condition, and takes place as below:

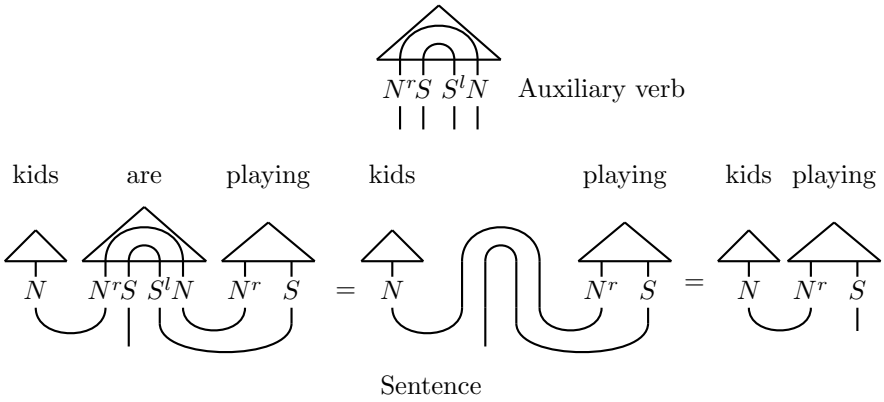
$$(1_S \otimes \epsilon^l_S) \circ (\eta^l_S \otimes 1_S)(\overrightarrow{she\ loved\ john}) = 1_S(\overrightarrow{she\ loved\ john}) \tag{5.4}$$

We have now a canonical way to "ignore" a word[1]—or, much more interestingly, parts of the internal structure of words that correspond to redundant interactions, as we will see later in this chapter. For the moment, I would like to note that for certain functional words that have little meaning of their own, such as copulas, auxiliaries and determiners, the decision to ignore them is an acceptable compromise that can save us from trying to provide answers to perhaps unanswerable questions (e.g. what is the distributional meaning of 'some'?). Applying the above method to the case of a determiner, for example, we get the following:



(5.5)

while for the slightly more complicated case of an auxiliary verb, we have:



(5.6)

[1]Note that technically we do not ignore anything; we have just decided to assign a particular meaning to the complementizer, and we construct its linear map in a way to reflect this meaning.

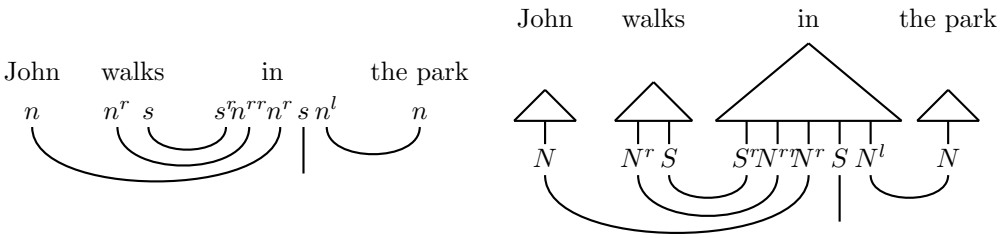
This time we create the tensor by nesting an η-map within another, getting the following composition (better depicted by a commutative diagram):

$$\begin{array}{ccccc}
N \otimes N^r \otimes I \otimes N \otimes N^r \otimes S & & \xrightarrow{1_{N\otimes N^r} \otimes \eta^l_S \otimes 1_{N\otimes N^r\otimes S}} & & N \otimes N^r \otimes S \otimes S^l \otimes N \otimes N^r \otimes S \\
\| & & & & \\
N \otimes N^r \otimes N \otimes N^r \otimes S & & & & \\
\uparrow{\scriptstyle 1_N \otimes \eta^r_N \otimes 1_{N^r} \otimes 1_S} & & & & \downarrow{\scriptstyle \epsilon^r_N \otimes 1_S \otimes 1_{S^l} \otimes \epsilon^r_N \otimes 1_S} \\
N \otimes I \otimes N^r \otimes S & & & & \\
\| & & & & \\
N \otimes N^r \otimes S & \xrightarrow{\epsilon^r_N \otimes 1_S} & S & \xleftarrow{1_S \otimes \epsilon^l_S} & S \otimes S^l \otimes S
\end{array} \tag{5.7}$$

Despite the usefulness of the above method in the aforementioned cases, it should be evident that it would not be adequate for words that occur in more complex grammatical interactions within a sentence. In what follows I show how to extend these ideas to various other functional words of the English language in a linguistically justified way, often resorting to using η-maps in combination with Frobenius operators.

5.2 Prepositions

Not all functional words can be ignored in the sense described in §5.1. Prepositions, for example, signify specific relations between text constructs that need to be semantically reflected: our model must be capable of expressing that the meaning of "John put the bag *on* the table" is different from that of "John put the bag *under* the table". At the same time, we need to find a way to deal with the fact that a verb-modifying (for example) preposition, is a tensor of order 5:



(5.8)

From a formal semantics perspective, a transitive preposition is seen as a restriction applied to the set of individuals performing a specific action. Consider, for example, the logical form for the above sentence:

$$\exists x[park(x) \wedge in(x)(walks)(john)] \tag{5.9}$$

Taking the meaning of predicate *walks* to be the set of individuals who generally walk, *in* is a function that takes as input a place x and applied to this set of individuals who walk, eventually returning a subset with all individuals who walk in x [31, p. 244]. Thus, a transitive preposition can be seen as a function with three arguments: a subject (*John*), an action (*walks*), and an object (*park*). This is directly reflected in the right-hand diagram of (5.8), where all this information is fed to the tensor of the preposition for processing.

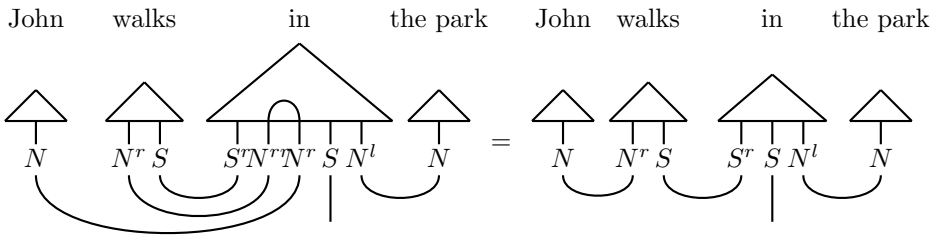
Again, the compact closed structure provides us some options. I will begin by noticing that the type $s^r \cdot s \cdot n^l$ is a more generic case of the type $s^r \cdot n^{rr} \cdot n^r \cdot s \cdot n^l$.[2] Indeed,

$$s^r \cdot s \cdot n^l = s^r \cdot 1 \cdot s \cdot n^l \leq s^r \cdot n^{rr} \cdot n^r \cdot s \cdot n^l \tag{5.10}$$

This type form denotes a function of two arguments which, in the case of verb-modifying prepositions, expects a sentence to the left (an intransitive verb *already combined* with a subject noun phrase), and an object noun phrase at the right in order to return a sentence. The translation of Eq. 5.10 to vector spaces is:

$$S^r \otimes S \otimes N^l \cong S^r \otimes I \otimes S \otimes N^l \xrightarrow{1_{S^r} \otimes \eta^r_{N^r} \otimes 1_S \otimes 1_{N^l}} S^r \otimes N^{rr} \otimes N^r \otimes S \otimes N^l \tag{5.11}$$

Using this tensor for the composition will produce the following normal form:



(5.12)

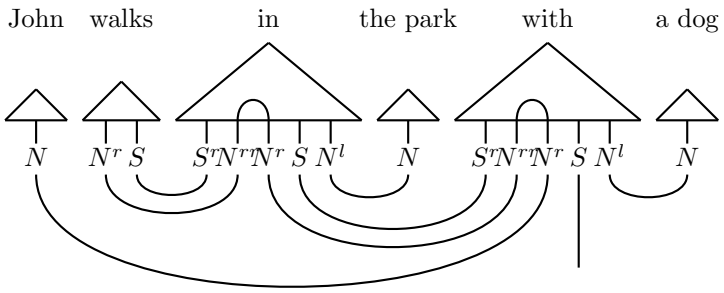
The representation of a verb-modifying preposition as a function of two arguments is a very convenient outcome, since we already know how to handle such a function from our work with transitive verbs in Chapter 4. Indeed, now we can proceed one step further and adopt the argument summing process of [40], by defining the meaning of a preposition as follows:

$$\overline{pr}_{TR} = \sum_i \overrightarrow{verb_i} \otimes \overrightarrow{noun_i} \tag{5.13}$$

[2]Note that the atomic type n^{rr} expresses an *iterated adjoint*: the right adjoint of a right adjoint.

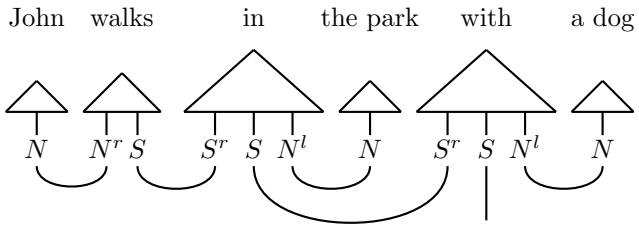
where $\overrightarrow{verb_i}$ corresponds to the verb modified by the preposition, and $\overrightarrow{noun_i}$ the object noun from the right. The usual Frobenius manipulations apply here; copying the object dimension of $\overline{in}$, for example, will compute the meaning of the sentence as $(\overline{in}^{\mathsf{T}} \times (\overline{walks}^{\mathsf{T}} \times \overrightarrow{john})) \odot \overrightarrow{park}$, which brings the same intonational interpretation as discussed in §4.5 by answering the question "where does John walk in?".

The above treatment scales naturally to cases where a verb is modified by more than one prepositional phrase, satisfying in this way one of the basic requirements of a PP-modification analysis. Extending further our previous example:



(5.14)

which produces the following normal form:



(5.15)

Before closing this section, I would like to note that in general noun-modifying prepositions, as in "a man in uniform", do not need any special treatment since they are already in a manageable 2-argument form, as it is evident by their type $(n^r \cdot n \cdot n^l)$. Obviously, the matrix of a noun-modifying preposition must be different of that corresponding to the verb-modifying variant; with $noun^{(s)}$ and $noun^{(o)}$ denoting the nouns at the "subject" and "object" position, we get:

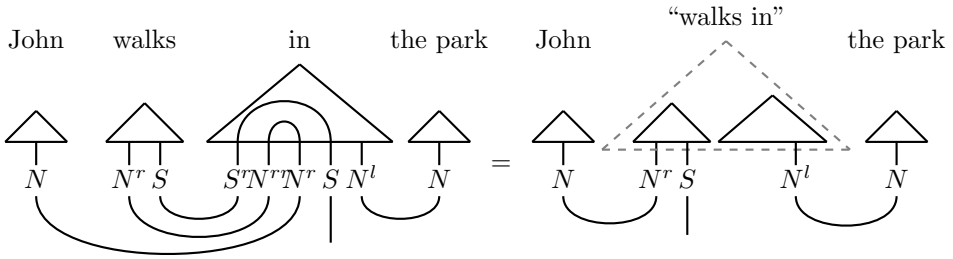
$$\overline{pr}_N = \sum_i \overrightarrow{noun}_i^{(s)} \otimes \overrightarrow{noun}_i^{(o)} \tag{5.16}$$

5.3 Phrasal verbs

Interestingly, the type $s^r \cdot s \cdot n^l$ we derived in §5.2 for a verb-modifying preposition can be further reduced if we create the preposition tensor by using a second η-map:

$$N^l \cong I \otimes N^l \xrightarrow{\eta_S^r \otimes 1_{N^l}} S^r \otimes S \otimes N^l \cong \tag{5.17}$$
$$S^r \otimes I \otimes S \otimes N^l \xrightarrow{1_{S^r} \otimes \eta_{N^r}^r \otimes 1_S \otimes 1_{N^l}} S^r \otimes N^{rr} \otimes N^r \otimes S \otimes N^l$$

Doing this might seem not very helpful in the beginning, since as it is evident in (5.18) below what we have "achieved" is to cancel out the interaction of the object ("the park") with the rest of the sentence. However, the interesting point of this treatment is that it provides a justification for viewing a preposition from a different perspective: as *part* of the preceding verb. Indeed, it looks like the sole purpose of the reduced preposition tensor is to provide another argument to the adjacent verb, creating a new *transitive* version of it:



(5.18)

This second model resembles a pre-Davidsonian way of working with prepositions, when the standard practice was to treat them as integral parts of the verbs. In his seminal paper, "The Logical Form of Action Sentences" [30], Davidson introduces the representation of prepositions as distinct predicates, explaining his reasoning as follows:

> In general, we conceal logical structure when we treat prepositions as integral parts of verbs; it is a merit of the present proposal that it suggests a way of treating prepositions as contributing structure.

However, the model of (5.18) still carries an important advantage, since it provides a canonical way to treat special cases where actually the contribution of prepositions to structure is not very strong, such as phrasal verbs ('look after', 'run into', 'stand by', 'hand in', etc). Creating a verb matrix for a phrasal verb is a procedure slightly more complicated than in the case of a normal verb. Specifically, given a text corpus that has been *dependency-parsed* into a set of relations (w_D, w_H, R), where w_H is the head word, w_D the dependent, and R the grammatical relation that holds between the two words, we can construct the reduced tensor of a phrasal verb *verb prep* by looking for chains of the following form:

$$(noun_1, verb, \text{SUBJ}) \rightarrow (verb, prep, \text{MOD}) \rightarrow (noun_2, verb, \text{OBJ}) \tag{5.19}$$

Here, MOD denotes a modifier. Then, $noun_1$ and $noun_2$ are the subject and the object, respectively, of the "phrasal" version of the verb, produced by attaching the preposition to it; the application of Eq. 4.9, by tensoring the vectors of the two nouns and adding them to a running total, will create a reduced tensor as required.

5.4 Coordination

Two coordinated phrases or sentences can be seen as contributing equally to the final outcome; we would expect, for example, the vector of the sentence "John reads and Mary sleeps" to reflect equally the vectors of the two coordinated sentences, "John reads" and "Mary sleeps". This is exactly one of the cases in which the necessity of a compositional model with the characteristics of our Frobenius framework (categorical composition in combination with point-wise multiplication) is very clear; the tensors of the coordinators must be created in a way to allow categorical composition for the conjuncts, and then merge the two composite vectors using point-wise multiplication. Motivated by the work of Sadrzadeh and colleagues on relative pronouns[3] [103, 104], in this section I will show how this can be achieved by a combination of Frobenius operators and η-maps.

For the analysis that follows I consider the usual ternary rule X CONJ $X \rightarrow X$, which states that coordination always take place between conjuncts of the same type and produces a result that again matches that specific type. In pregroups terms, this is achieved by assigning the type $x^r \cdot x \cdot x^l$ to the conjunction (where x can be an atomic or a complex type), which leads to the following generic derivation:

$$x \cdot (x^r \cdot x \cdot x^l) \cdot x \leq 1 \cdot x \cdot 1 = x \tag{5.20}$$

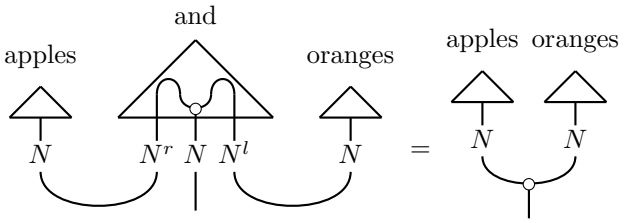
5.4.1 Coordinating noun phrases

I will start with the simple case of a noun-phrase coordinating conjunction, the type of which is $n^r \cdot n \cdot n^l$. In order to achieve the desired result, I will create the corresponding construction in **FVect** using the following morphism:

$$I \xrightarrow{\eta_N^r \otimes \eta_N^l} N^r \otimes N \otimes N \otimes N^l \xrightarrow{1_{N^r} \otimes \mu_N \otimes 1_{N^l}} N^r \otimes N \otimes N^l \tag{5.21}$$

[3]Which is quite relevant to the models of Chapter 4 and will be presented in more detail in §5.7.

The composition now of a coordinated noun phrase takes this form:



(5.22)

As shown above, the closed-form formula of this construction is the morphism:

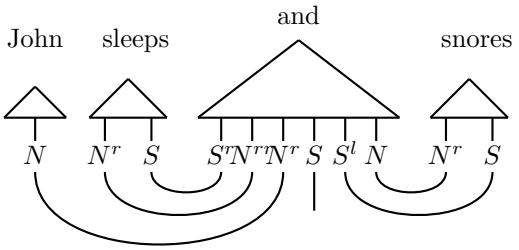
$$\mu(\overrightarrow{apples} \otimes \overrightarrow{oranges}) = \overrightarrow{apples} \odot \overrightarrow{oranges} \tag{5.23}$$

5.4.2 Coordinating verb phrases

The case of verb phrase coordination is more interesting and involved. Recall that the pregroup type of a verb phrase is $n^r \cdot s$; that is, something that expects a noun (a subject) from the left in order to return a sentence. Using the ternary rule in Eq. 5.20 and the properties of pregroups,[4] we derive the following type for a coordinator between two verb phrases:

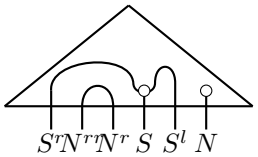
$$(n^r \cdot s)^r \cdot (n^r \cdot s) \cdot (n^r \cdot s)^l \leq s^r \cdot n^{rr} \cdot n^r \cdot s \cdot s^l \cdot n^{rl} \leq s^r \cdot n^{rr} \cdot n^r \cdot s \cdot s^l \cdot n \tag{5.24}$$

This requires a tensor of order 6, as in the following example:



(5.25)

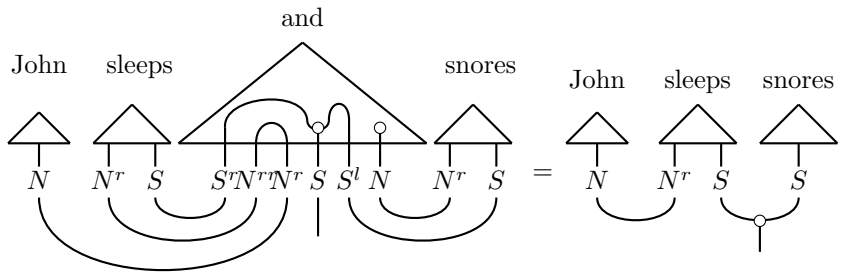
One intuitive way to model this so that it imposes "equal contribution" of the two verbs in the final outcome is to define the inner structure of the coordinator as follows:



(5.26)

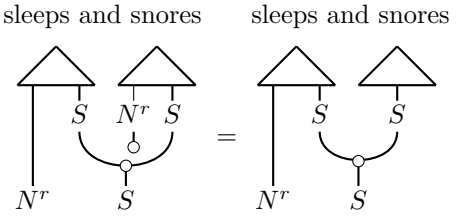
[4] It can be shown that $(x \cdot y)^l = y^l \cdot x^l$ and $(x \cdot y)^r = y^r \cdot x^r$; furthermore, it also holds that $x^{lr} = x = x^{rl}$.

Note that in this case we use for the first time the Frobenius ι-map, the purpose of which is to *delete* a wire. The motivation behind this becomes apparent in the normal form that results after the composition of the coordinator with the rest of the context:



(5.27)

Deleting the noun dimension of the second verb helps us to create a coherent compact representation for the coordinated structure "sleeps and snores", that can indeed be seen as a new verb phrase. Note that the sentence dimension of this structure is produced by "merging" the sentence dimensions of the two conjuncts:



(5.28)

Categorically, the original tensor of the coordinator is created by the following morphism:

$$I \xrightarrow{\eta^r_S \otimes \eta^l_S} S^r \otimes S \otimes S \otimes S^l \cong$$
$$S^r \otimes I \otimes S \otimes S \otimes S^l \xrightarrow{1_{S^r} \otimes \eta^r_{N^r} \otimes \mu_S \otimes 1^l_S} S^r \otimes N^{rr} \otimes N^r \otimes S \otimes S^l \quad (5.29)$$

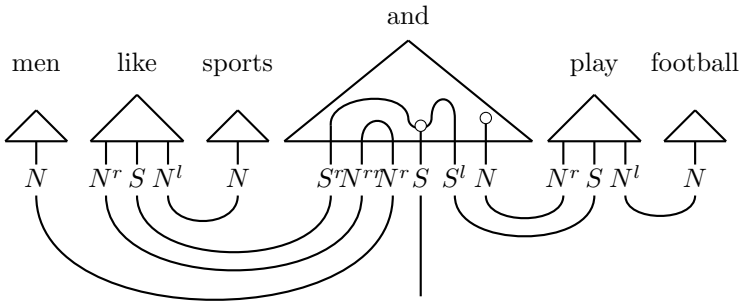
I will use the normal form of (5.27) to compute the meaning of the coordinated structure as below:

$$(\epsilon^r_N \otimes \mu_S)(\overrightarrow{john} \otimes \overline{sleeps} \otimes \overline{snores}) = (\overrightarrow{john}^{\mathsf{T}} \times \overline{sleeps}) \odot \overrightarrow{snores} \quad (5.30)$$

where $\overrightarrow{snores}$ is the vector produced by deleting the row dimension of the original verb matrix for 'snores'. Linear-algebraically (and when one uses full tensors, as opposed to the reduced Frobenius tensors presented in Chapter 4), the application of ι-map (Eq. 3.37) to a matrix has the following result:

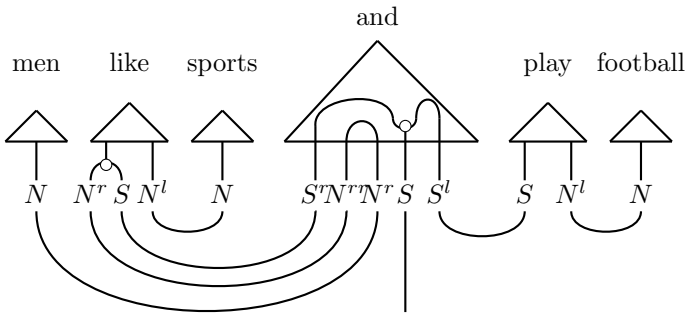
$$(\iota \otimes 1_N)\left(\sum_{ij} v_{ij}\overrightarrow{n_i} \otimes \overrightarrow{n_j}\right) = \sum_{ij} v_{ij}\overrightarrow{n_j} \tag{5.31}$$

This is exactly the vector created by summing the rows of the verb matrix. It would be also instructive to see how this method fits into the Frobenius framework of Chapter 4. Imagine the following more complicated case of verb phrase coordination, where the coordinated verbs occur with an object noun:



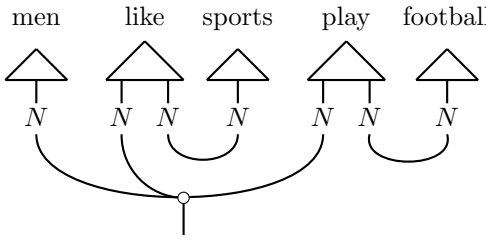
(5.32)

Let me first create the tensors for verbs 'like' and 'play' as reduced matrices (§4.1). I will expand the tensor of 'like' by copying its subject dimension; however, I will not apply any dimension copying on the 'play' tensor, since one of its dimensions needs to be deleted anyway. The above derivation gets the following form:



(5.33)

which has the following normal form, by the application of the spider theorem:



(5.34)

In linear-algebraic terms, the result is once again highly intuitive: the vectors of the two verb phrases are merged together (as required by the conjunction), and the result is further merged with the subject of the sentence—which has to contribute

equally to the final outcome, since according to the Copy-Subject model we applied it has the role of the rheme:

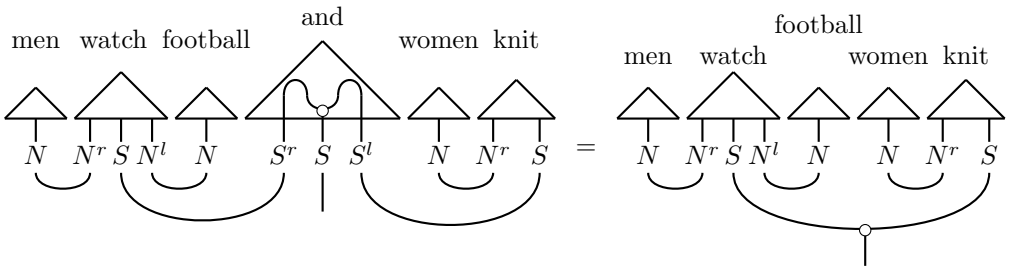
$$\overrightarrow{men} \odot (\overline{like} \times \overrightarrow{sports}) \odot (\overline{play} \times \overrightarrow{football}) \tag{5.35}$$

5.4.3 Coordinating sentences

The type of a sentence coordinator is $s^r \cdot s \cdot s^l$, leading to a situation very similar to that of the noun phrase case. We create the tensor as follows:

$$I \xrightarrow{\eta^r_S \otimes \eta^l_S} S^r \otimes S \otimes S \otimes S^l \xrightarrow{1_{S^r} \otimes \mu_S \otimes 1_{S^l}} S^r \otimes S \otimes S^l \tag{5.36}$$

and get the following generic derivation:

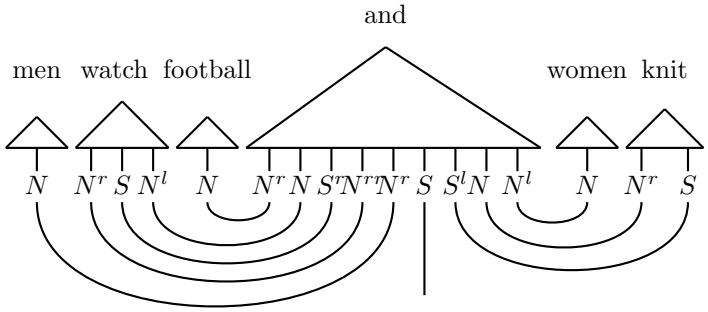


(5.37)

Linear-algebraically:

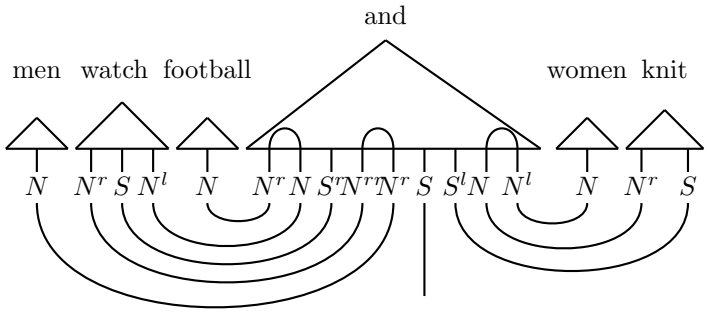
$$(\overrightarrow{men}^{\mathsf{T}} \times \overline{watch} \times \overrightarrow{football}) \odot (\overrightarrow{women}^{\mathsf{T}} \times \overline{knit}) \tag{5.38}$$

Note that in general the type of a sentence coordinator can occur in expanded form, as below:



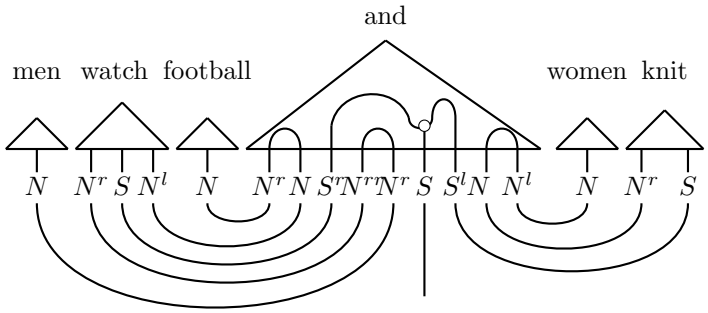
(5.39)

As in the case of verb-modifying prepositions (§5.2), it can be shown that all these cases reduce to the canonical form $s^r \cdot s \cdot s^l$. For the derivation in (5.39), we get:



(5.40)

Applying the previous Frobenius method to the coordinator tensor will give:



(5.41)

the normal form of which is exactly the same as the generic case depicted in (5.37).

5.4.4 Distributivity

The coordination treatment presented above fits nicely with the theoretical basis detailed in this thesis, but it has a drawback: the element-wise multiplications introduced by the Frobenius operators do not interact well with tensor contraction with regard to distributivity, which means that the proposed system cannot adhere to basic entailments of the following form:

(7) Mary studies philosophy and history $\models$ Mary studies philosophy and Mary studies history

(8) Men like sports and play football $\models$ Men like sports and men play football

where symbol $\models$ denotes semantic entailment. In (7) the subject-verb part distributes over noun phrases, while in (8) the subject distributes over verb phrases. In both cases the entailed part is a coordination of sentences. The obvious way to allow this kind of distributivity in a tensor-based setting is to replace the element-wise multiplication with vector addition, since for a tensor $\mathbf{W}$ and two vectors $\overrightarrow{v}, \overrightarrow{w}$ it is always the case that $\mathbf{W}(\overrightarrow{v} + \overrightarrow{w}) = \mathbf{W}\overrightarrow{v} + \mathbf{W}\overrightarrow{w}$. For our examples we get the out-of-the-box derivations shown below:

$$\overrightarrow{mary} \times \overline{studies} \times (\overrightarrow{phil} + \overrightarrow{hist}) = \overrightarrow{mary} \times \overline{studies} \times \overrightarrow{phil} + \overrightarrow{mary} \times \overline{studies} \times \overrightarrow{hist}$$
$$\overrightarrow{men} \times (\overline{like} \times \overrightarrow{sp} + \overline{play} \times \overrightarrow{fb}) = \overrightarrow{men} \times \overline{like} \times \overrightarrow{sp} + \overrightarrow{men} \times \overline{play} \times \overrightarrow{fb}$$

The result is a setting similar to the original, in which the multiplicative vector mixture model we initially used to represent coordination has been replaced by the additive variation. So why not prefer this method in the first place over the Frobenius alternative? There are a couple of reasons. First, from a linguistic perspective vector addition seems an operation much more appropriate for modelling *disjunctive* cases of coordination (a topic we did not address in this thesis), as explained by Widdows [121] in the context of quantum logic. There is a second very important reason: In the case of an intonational interpretation, as described in §4.5, the forms of basic entailment in (7) and (8) *do not* actually hold. Take for example the following Copy-Subject variation of the left-hand case in (8), as computed in Eq. 5.35:

(9) *Who likes sports and plays football?*
Men like sports and play football:
$\overrightarrow{men} \odot \overline{like} \times \overrightarrow{sports} \odot \overline{play} \times \overrightarrow{football}$

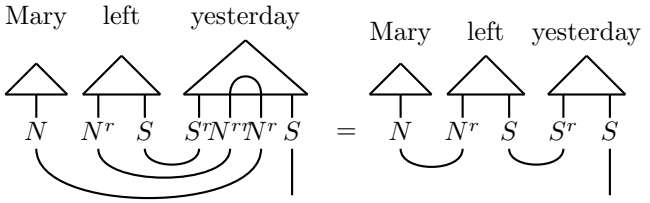
Note, however, that the Copy-Subject version of the right-hand part in (8) answers to a slightly different question—or, more accurately, to a conjunction of two *distinct* questions:

(10) *Who likes sports and who plays football?*
Men like sports and **men** play football:
$(\overrightarrow{men} \odot \overline{like} \times \overrightarrow{sports}) \odot (\overrightarrow{men} \odot \overline{play} \times \overrightarrow{football}) =$
$\overrightarrow{men}^2 \odot \overline{like} \times \overrightarrow{sports} \odot \overline{play} \times \overrightarrow{football}$

The two resulting vectors are quite similar, with the important and necessary difference that in the second case the contribution of word 'men' is intensified, exactly as required by the question at hand.

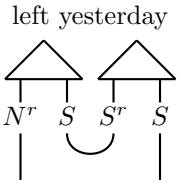
5.5 Adverbs

An adverb is something that usually modifies the meaning of a verb or an adjective. For temporal or spatial adverbs that occur *after* the modified verb, I propose the following representation:



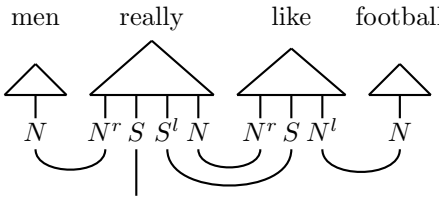
(5.42)

Intuitively, a verb modified by an adverb gets this new form:



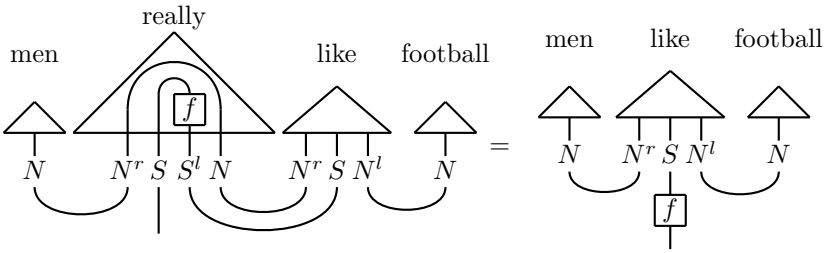
(5.43)

It is even possible for one to apply the argument summing procedure and create the meaning of the adverb as the sum of all verbs that the specific adverb modifies in the training corpus. The case of an adverb that occurs before the verb, as below:



(5.44)

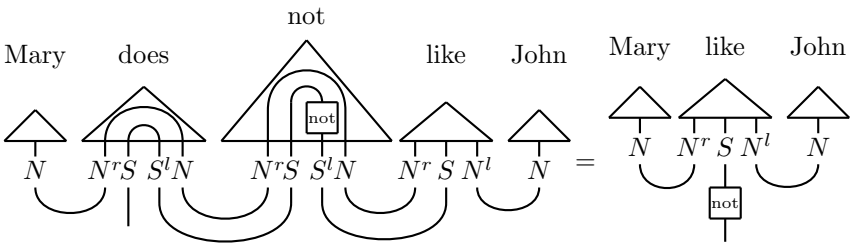
is more complicated, since the adverbial linear map must be applied on the sentence dimension of the verb. In order to achieve this we need to follow [95] and define the inner $S \otimes S^l$ part of the adverb tensor as an *operator*—that is, a matrix that corresponds to a linear map $f : S \to S$. In our graphical calculus, this is depicted as follows:[5]



(5.45)

Again in this case, our linear map can be created as the sum of all verbs that the specific adverb modifies in the training corpus. Note that the use of an operator is also the canonical way to define negation in a simple declarative sentence, assuming one has a way to create a map that "negates" the meaning of a sentence:

[5]The concept of an operator, as well as the specific graphical notation I used to denote it, will be explained in more detail in §6.3.3.

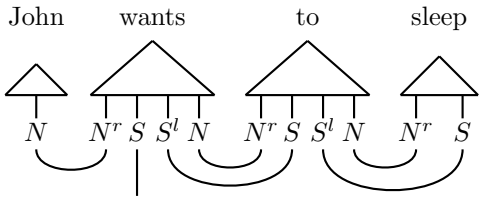


(5.46)

In a truth-theoretic setting, for example, similar to that of §2.3.5, $\overline{not}$ can simply be $\left(\begin{smallmatrix} 0 & 1 \\ 1 & 0 \end{smallmatrix}\right)$ as explained in [95]. Unfortunately, it is not quite clear how one can successfully generalize this map to high dimensional real-valued vector spaces; the work of Widdows [121] provides valuable insights on this in the context of quantum logic.

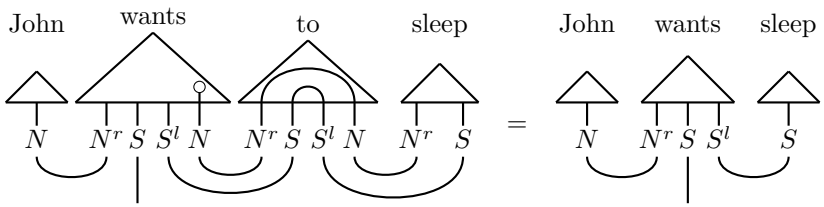
5.6 Infinitive phrases

An infinitive phrase in the English language is a verb phrase in which the verb occurs in its infinitive form, that is, following the particle *to*. Perhaps the most typical use of an infinitive verb phrase is to *complement* some other verb, as below:



(5.47)

For these cases, I propose the complete "bridging" of the infinitive particle in conjunction with the deletion of the noun dimension of the verb, as below:



(5.48)

Note that this approach assigns the type $n^r \cdot s \cdot s^l$ to the complemented verb 'want', which is now a function of two arguments: a subject noun from the left, and the bare infinitive form of the verb at the right. Since in the new formulation there is no direct connection between the noun ('John') and the bare infinitive ('sleep'), the interaction of those two words is assumed to be handled by the head word of the sentence ('wants'). This suggests that an appropriate method of constructing the tensors of the complemented verb according to the argument summing procedure is the following:

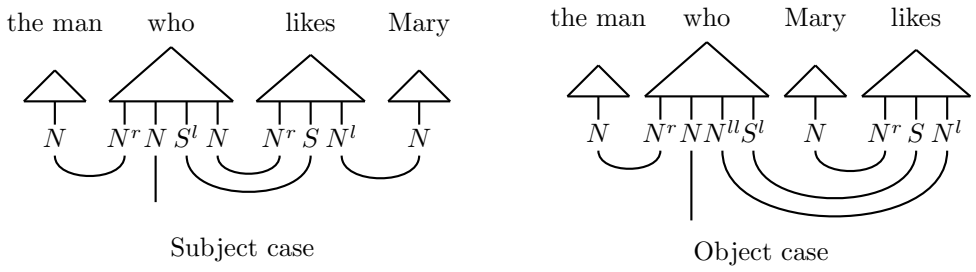
$$\overline{verb}_C = \sum_i \overrightarrow{subj}_i \otimes \overrightarrow{verb}_i^{BI} \tag{5.49}$$

where $\overrightarrow{verb}_i^{BI}$ denotes the distributional vector of a verb whose infinitive form complements $verb_C$ in some sentence.

5.7 Relative pronouns

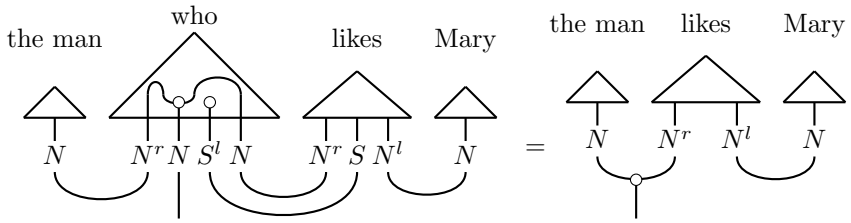
Recently, the treatment of relative pronouns in a tensor-based setting received a lot of attention from Sadrzadeh, Clark and Coecke, who provide intuitive insights regarding the "anatomy" (i.e. internal structure) of tensors in a series of papers [103, 104]. Their method is based on the application of Frobenius algebras, providing connections and common points with the way we model transitive verbs in Chapter 4. The work in [103, 104] is very relevant to the material presented in this thesis in general and an inspiration for some of the methods I used in this chapter in particular, and in this section I'm going to review its most important points and make the connection with the Frobenius framework of Chapter 4 explicit.

The purpose of a relative pronoun is to mark a relative clause, i.e. a text constituent that modifies the noun at the left (the *antecedent*). In its most common form, the antecedent can be the subject or the object of the relative clause's head verb, a fact that leads to slightly different grammatical types and derivations for the two cases:



(5.50)

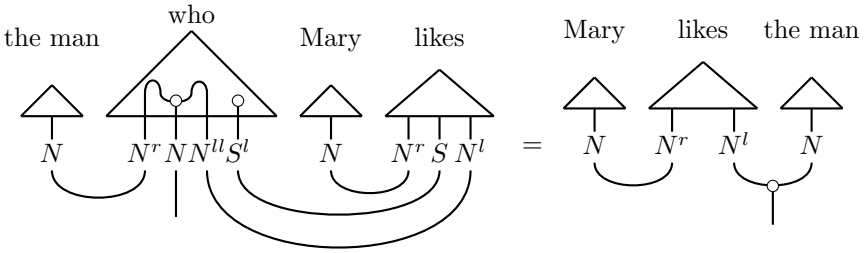
In [103], the copying abilities of Frobenius algebras are cleverly exploited to allow the passing of information of the antecedent from the left-hand part of the phrase to the right-hand part, in order to let it properly interact with the modifier part:



(5.51)

Given a reduced verb tensor created according to Eq. 4.9, the normal form of (5.51) above is identical to the Frobenius vector that results after copying the subject dimension of a transitive verb (Diagram 4.16). Recall from the discussion in §4.5 that this is exactly the vector that puts intonational emphasis on the subject, since this part of the sentence/phrase contributes equally with the verb phrase to the final outcome through point-wise multiplication. The interpretation in the context of a subject relative pronoun follows the same intuition: the subject still constitutes the most important part of the phrase, *modified* by the part that follows the relative pronoun. The relative pronoun itself does not appear in the final calculation.

Following the same idea, we can model an object relative pronoun as below:



(5.52)

which this time puts the emphasis on the object. Finally, for the case of possessive relative pronouns (e.g. "the man *whose* friend likes Mary"), which will not be discussed here, the reader may refer to [104].

When one tries to put together the Frobenius framework of Chapter 4 with the relative pronouns treatment presented in this section, one has to face the apparent inconsistency that the meaning of a noun phrase, say "men who like football", will be identified with the meaning of the sentence "**men** like football" (used as an answer to the question "Who likes football?"). It is interesting, though, to note that despite the obvious simplification that such an identification imposes, the underlying reasoning remains quite valid. First and foremost, we observe that the subject noun is the most important part in both constructs. What changes is the perspective from which we see the composition with the context: in the relative pronoun case, we consider the context as a modifier of the noun, whereas in the intonational case, we emphasize

the equal contribution of the two parts to the final composite vector. Setting aside the unfortunate effect that they produce the same concrete vectorial representation, these solutions are indeed intuitive and linguistically justified.

5.8 A step towards complete coverage of language

Back in §2.3.3, we identified as one of the most important challenges related to tensor-based models the fact that the designer has to devise maps for every word type of the language. In this chapter and in Chapter 4 we covered some of the necessary ground towards this purpose. The proposed methodology, as hopefully it is evident from the material presented in this thesis, allows for a scalable framework that is open to creative solutions and linguistically motivated reasoning. Besides a challenge, then, I consider the glass-box approach that this thesis adopts a very important advantage, one step further from running (for example) a deep net-based black box a number of times with different parameters until our numbers exceed some threshold. This is actually one of the most appealing characteristics of tensor-based models: they allow us to actively express our intuitions regarding both the compositional and distributional behaviour of the various relational and functional words.

The use of the Frobenius algebras in the way described in this chapter and Chapter 4 is a means of providing additional structure to the compact setting of the original model, which essentially comes with a rather limited set of tools consisting of a tensor operation and η and ϵ maps. As we saw, language can be more complex than this. The introduction of Frobenius operators provides a second form of composition which co-exists with tensor product and can help modelling more complex linguistic phenomena such as intonation and coordination. From this perspective, the research presented here can be seen as laying the groundwork for a novel form of a grammar enriched with Frobenius structure. In such a grammar, the appropriate translation from syntax to semantics would be explicitly handled in the functorial passage itself, rather than by direct tensor assignments as in this thesis. The intuition in both cases is the same and aims at providing a compositional model better tailored to the nature of vector space semantics than the current attempts. The goal is twofold: (a) to optimize the compositional process by eliminating redundant interactions (especially in the case of functional words); and (b) to circumvent tensor product and impose equal contribution between words when this makes linguistic sense.

A number of proposals detailed in this chapter have been applied in practice with positive results in the context of an experiment we are going to see later in §7.3. The

goal of the task is to classify simplified dictionary definitions to their corresponding terms based on the distance of their vectors. The experiment involves the creation of composite vectors for the definitions, a process that required linear maps for verb- and noun- modifying prepositions as well as for various types of conjunctions. However, a large scale evaluation of the categorical model on sentences of completely arbitrary structure remains still a long-standing goal to which hopefully the current work contributes positively.

Chapter 6

Dealing with Lexical Ambiguity

Chapter Abstract

This chapter investigates the relation between disambiguation and composition in the context of CDMs. A new compositional methodology is proposed which is capable of efficiently handling the different levels of lexical ambiguity in language by introducing an explicit disambiguation step before the composition. Furthermore, the categorical model of [25] is recast in a higher-level framework which incorporates ambiguity by following a quantum-theoretic perspective.

Section 2.3 provided a concise introduction to a number of CDMs, which were further categorized according to the method they utilize to represent relational words and the composition function they employ (§2.4). Regardless of their level of sophistication, however, most of these models do share a common feature: they all rely on ambiguous vector representations, where all the meanings of a homonymous word, such as the noun 'bank', are merged into the same vector or tensor. An interesting question, then, refers to the way in which CDMs affect (and get affected by) those ambiguous words; in other words, given a sentence such as "a man was waiting by the bank", we are interested to know to what extent a composite vector can appropriately reflect the intended use of word 'bank' in that context, and how such a vector would differ, for example, from the vector of the sentence "a fisherman was waiting by the bank".

Interestingly, from a formal semantics perspective this question is irrelevant, since the meanings of words are represented by logical constants that have been explicitly set before the compositional process. In the case of CDMs, however, ambiguity is a very important factor. When acting on an ambiguous vector space, a CDM seems to perform two tasks at the same time, composition *and* disambiguation, leaving the

resulting vector in a state that is hard to interpret. In the ideal case, the composition function should be able to perform both tasks in parallel, providing an equivalent of the notion of *continuous meaning*, as this is discussed in §6.1. In practice, though, as a series of experiments in Chapter 8 will reveal, this is hardly the case. I argue that the reason for this is that a CDM is not capable of treating all kinds of ambiguity in a uniform way. There should be a clear separation of cases of *homonymy*, where the same word has more than one completely disjunctive meaning, from *polysemous* cases, in which a word has multiple but still related senses.

The purpose of this chapter is to investigate these issues in the context of compositional models of meaning to an appropriate depth, and to provide insights that can lead to more efficient CDMs. I will start by discussing the different levels of lexical ambiguity that may occur in language and I will attempt a connection with two different notions of meaning originated in psycholinguistics, that of continuous and discretized meaning. Later, in §6.3, we will see how these concepts can be incorporated into the categorical framework of Coecke et al. [25] by exploiting the fact that both our model and quantum mechanics follow a vector space semantics.

6.1 Understanding lexical ambiguity

In order to deal with lexical ambiguity, as the title of this chapter claims it is aiming to do, we firstly need to understand its nature. In other words, we are interested to study in what way an ambiguous word differs from an unambiguous one, and what is the defining quality that makes this distinction clear. On the surface, the answer to these questions seems straightforward: an ambiguous word is one with more than one definition in the dictionary. Inevitably, though, this leads to another question, for which, as Hoffman et al. note [43], there exists no satisfactory answer:

What is the criterion of considering two word uses as separate senses?

For *homonymous* cases, in which due to some historical accident words that share exactly the same spelling and pronunciation refer to completely different concepts, the distinction is straightforward. One such case is the word 'bank', which can mean financial institution, land alongside river, or a sequence of objects in a row. However, for *polysemous* words things are not so simple. In contrast to homonymy, *polysemy* relates to subtle deviations between the different senses of the same word. Such a case for example would be 'bank' again as a financial institution and as the concrete building in which the financial institution is accommodated. And while it is not

wrong for one to claim that the number of meanings[1] of a homonymous word can be considered almost fixed across different dictionaries, the same is not true for the small and overlapping variations of senses that might be listed under a word expressing a polysemous case.

The approach of classifying a word as ambiguous or not based on the number of its discrete lexicographic entries in a dictionary is in line with most psycholinguistic studies of ambiguity (see, for example [12, 100]). However, a different perspective that is progressively becoming more popular among researchers rejects the assumption that a word is restricted to a limited number of predefined senses. Instead of this, the meaning of every word is considered dynamic, varying continuously depending on the context in which the specific word appears [70, 62, 43]. Under this perspective, two uses of the same word are never truly identical in meaning, as their exact connotations depend on the specific context in which they appear. Hence, the separation of a word's meaning to some predefined number of senses is nothing more than an artificial *discretization* of the underlying continuous spectrum of senses.

The concept of continuous meaning is very intuitive and fits naturally with the distributional semantics perspective, where a word is expressed in terms of all other words in the vocabulary. However, I argue that it can be only applied on polysemous cases, where a word still expresses a coherent and self-contained concept. Recall the example of the polysemous use of 'bank' as a financial institution and the building where the services of the institution are offered; when we use the sentence 'I went to the bank' (with the financial meaning of the word in mind) we essentially refer to both of the polysemous meanings of 'bank' at the same time—at a higher level, the word 'bank' expresses an abstract but concise concept that encompasses all of the available polysemous meanings. On the other hand, the fact that the same name can be used to describe a completely different concept (such as a river bank or a number of objects in a row) is nothing more than an unfortunate coincidence expressing lack of specification.

For cases like these, a combination of the notions of discretized and continuous meaning seems a better strategy. The lack of specification imposed by the homonymous meanings must be addressed by an initial discretization step, responsible for defining explicit semantic representations for every homonymous use of the word at hand; furthermore, the notion of continuous meaning will be still retained within the produced parts, which will now provide much more coherent and accurate semantic

[1]From this point on, I adopt the convention to refer to *meanings* of a word when talking about homonymous cases, and to *senses* of a word for polysemous cases.

representations for the corresponding concepts.

We have now arrived at a central claim of this thesis: when the use of the same word to describe two concepts is nothing more than an unfortunate historical coincidence, discretized meaning should be favoured over continuous meaning. Let me make this idea precise:

$$\begin{aligned} \text{homonymy} &= \text{discretized meaning} \\ \text{polysemy} &= \text{continuous meaning} \end{aligned} \tag{6.1}$$

In the next section I outline a concrete proposal for incorporating the above relationships into a CDM.

6.2 Prior word disambiguation and composition

Transferring the above discussion to high dimensional vector spaces, we start by observing that a vector space in which all the meanings of an ambiguous word are fused into the same vector representation is a setting far less than optimal. Indeed, having a single vector representing all meanings of a genuinely homonymous word does not make more sense than having a single vector representing the meanings of some other completely unrelated words, say 'book' and 'banana'. A vector like this would be the average of all senses, inadequate to reflect the meaning of any of them in a reliable way. This is demonstrated in Fig. 6.1 for the homonymous word 'organ'.

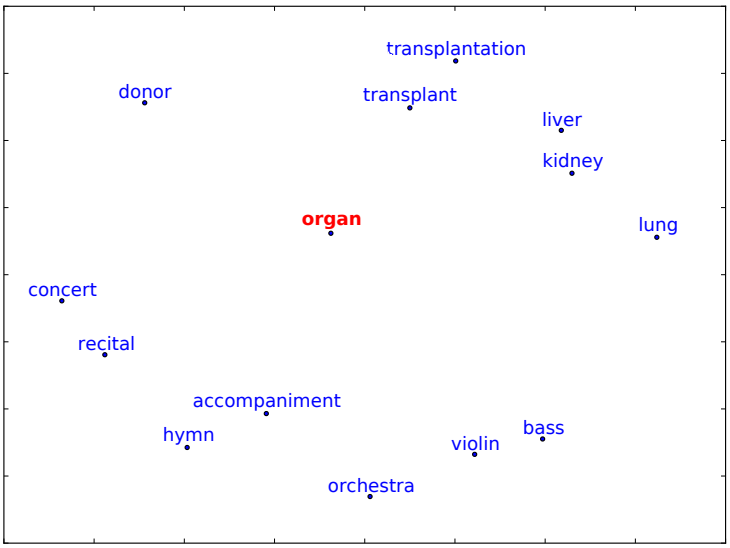


Figure 6.1: The vector of a homonymous word in a two-dimensional semantic space (projection of real word vectors using multi-dimensional scaling).

In order to avoid using such uninformative inputs in our compositional function, every distinct *meaning* of a word should be represented by a different vector. We can think of this requirement as the inevitable discretization related to the homonynous cases. On the other hand, we would expect that any composition function between word vectors would at least be able to adequately resolve the small deviations of senses due to polysemy, providing in this way an analogy of the continuous meaning concept.

I will now proceed and present a way to achieve the above. Following Pulman [97] and subsequent work I conducted with him and Sadrzadeh [52], I propose the introduction of an explicit disambiguation step that precedes the composition, which will be responsible for the homonymy-related discretization of the meaning. The disambiguated vectors will be used as inputs to the actual composition step, which will further refine the meaning of words to arbitrary levels of polysemy. The main idea is to replace Eq. 1.1, aiming at providing a meaning for a sentence by composing the vectors for the words therein, with the following:

$$\overrightarrow{s} = f\left(\phi(\overrightarrow{w_1}, C_1), \phi(\overrightarrow{w_2}, C_2), \ldots, \phi(\overrightarrow{w_n}, C_n)\right) \tag{6.2}$$

where f is our compositional function as before, and $\phi(\overrightarrow{w_i}, C_i)$ is a function that provides a disambiguated version of the vector (or tensor) for ith word given the context C_i of this word (e.g. all other words in the same sentence). We can think of function ϕ as a means to handle all the homonymous cases, when the gap in meaning is quite big to be handled by the composition function alone.

A reasonable question that the above proposal poses is how one can explicitly recognize and treat differently homonymous and polysemous cases of words, a problem that has been proved far from trivial in linguistics. The answer is that such an action is not necessary, given any rough discretization of meaning that at least captures the homonymous uses of the word to an adequate level. Assuming that a word w is polysemous, and not homonymous, we can expect that all "meaning" vectors that might be produced by $\phi(\overrightarrow{w}, C_w)$ will be very similar to each other as well as to the original ambiguous vector $\overrightarrow{w}$, reflecting the fact that there are only small deviations between the different senses of the word. Hence, the effect for the outer composition function f should be minimal (see also discussion in [7], p. 94).

The effectiveness of the methodology expressed by Eq. 6.2 has been experimentally tested and verified in a series of experiments, which will be presented in Chapter 8 along with concrete methods for performing sense induction and disambiguation of word vectors and tensors. My goal for the rest of the present chapter is to show

how the idea of treating each level of lexical ambiguity in a different way finds an intuitive manifestation in the context of the categorical model of [25]. Specifically, I will exploit the framework of quantum mechanics in order to provide an ambiguity-aware extension of the model that is capable of handing the notions of homonymy and polysemy in a unified and natural manner.

6.3 A quantum perspective in linguistics

Although seemingly unrelated, quantum mechanics and linguistics share a common link through the framework of compact closed categories, which has been used in the past to study and reason about phenomena related to both of these areas. The application of the categorical framework to linguistics is the subject of this thesis; even more interestingly, exactly the same machinery has been used in the past by Abramsky and Coecke [1] for providing structural proofs for a class of quantum protocols, essentially recasting the vector space semantics of quantum mechanics in a more abstract way. In this section I will try to make the connection between the two fields even more explicit, taking advantage of the fact that the ultimate purpose of quantum mechanics is to deal with uncertainty—and this is essentially what we need to achieve here in the context of language.

6.3.1 From simple vectors to quantum states

As we discussed in §2.1, the formal semantics perspective interprets the meanings of words as unique primitives. For a vocabulary V, let these primitives be given by a set of orthogonal vectors $\{|n_i\rangle\}_i$ with $0 \leq i < |V|$; the vector in which the ith element is 1 represents the ith word in V in a classical sense. Now let us use the distributional hypothesis in order to start moving towards the quantum realm. Take a semantic space fixing a basis as above; a vector in that space will be represented as the sum:[2]

$$\overrightarrow{w} = a_1|n_1\rangle + a_2|n_2\rangle + \ldots + a_n|n_n\rangle \tag{6.3}$$

Under this setting, we think of words as *quantum systems* and bases $|n_i\rangle$ as describing a specific unambiguous *state*: the state of word w occurring in the vicinity (i.e. same context) of the ith word in the vocabulary. Then Eq. 6.3 can be seen as a *superposition* of all these states, and the coefficients a_i as denoting the strength of

[2]Traditionally, vectors representing quantum states are given in *Dirac notation*, and in this chapter I am going to follow this convention.

every state. In quantum mechanics terms, we have a *pure state* that expresses our knowledge regarding the words with which w co-occurs in language (and how often), when w is taken out of context. That is, when w is seen alone, it exists in an abstract state where it co-occurs with *all* the words of the vocabulary with strengths denoted by the various coefficients.

Note that, in the ideal case, every disjoint meaning of a genuinely homonymous word must be represented by a different pure state. We expect the word *bank* to have at least two different meanings, which will correspond to distinct pure states:

$$\begin{aligned} \overrightarrow{bank}_{fin} &= a_1|n_1\rangle + a_2|n_2\rangle + \ldots + a_n|n_n\rangle \\ \overrightarrow{bank}_{riv} &= b_1|n_1\rangle + b_2|n_2\rangle + \ldots + b_n|n_n\rangle \end{aligned} \tag{6.4}$$

This is a direct consequence of the fact that *bank* as a land alongside a river is expected to be seen in drastically different contexts than as a financial institution, so that $\{b_i\}_i \neq \{a_i\}_i$—that is, it is essentially a different word. However, since the same "label" is used to express two different concepts, we need a way to express our genuine uncertainty (in a classical sense) regarding the state in which our system may be found (i.e. under which sense the specific word is used in a specific context). In quantum mechanics terms, this is achieved by the means of a *mixed state*, which is seen as a statistical ensemble of pure states. A mixed state is described by a *density matrix*: a self-adjoint ($\rho = \rho^\dagger$) positive semi-definite ($\langle v|\rho|v\rangle \geq 0$ for all $|v\rangle$) matrix of trace one, created as below:

$$\rho(w) = \sum_i p_i |s_i\rangle\langle s_i| \tag{6.5}$$

where $\{(p_i, |s_i\rangle)\}_i$ denotes an ensemble of pure states, each one of which can occur with probability p_i; we also have $\sum_i p_i = 1$. A density matrix can also be used for expressing a single pure state (that is, a word with a single definition); in that case we simply have:

$$\rho(w) = |w\rangle\langle w| \tag{6.6}$$

Before proceeding further I would now like to restate "Eq." 6.1, incorporating in it the quantum perspective we have just discussed. Under this new view, quantum superposition (pure states) accounts for the continuous meaning, while the discretization of meaning is achieved through density matrices (mixed states). That is, we have:

$$\begin{gathered}\text{homonymy} = \text{discretized meaning} = \text{mixed state (density matrix)} \\ \text{polysemy} = \text{continuous meaning} = \text{pure state (superposition)}\end{gathered} \tag{6.7}$$

The next section will hopefully make clear what this means in practice.

6.3.2 Quantum measurements

Density matrices interact with other self-adjoint operators known as *observables*, to produce quantum measurements. Note that, by the spectral theorem, every self-adjoint matrix A has an eigen-decomposition $A = \sum_i e_i |e_i\rangle\langle e_i|$, where e_i is an eigenvalue associated with eigenvector $|e_i\rangle$; furthermore, the eigenvectors of A constitute an orthonormal basis. An observable describes some quantity, such as spin or momentum; its eigenvalues correspond to possible values of this quantity, while the associated eigenvectors (or *eigenstates*) are the unambiguous states assigned to these values. Assuming a system in state $|\psi\rangle$, a measurement of A with respect to that (pure) state will return the expectation value of A in state ψ:

$$\langle A \rangle_\psi = \langle \psi | A | \psi \rangle \tag{6.8}$$

Note that when $|\psi\rangle$ corresponds to one of the unambiguous eigenstates, the above calculation returns the associated eigenvalue with absolute certainty, as claimed above:

$$\langle e_i | A | e_i \rangle = \langle e_i | e_i | e_i \rangle = e_i \tag{6.9}$$

Now, in the case of an arbitrary state vector $|\psi\rangle$, we observe that any such vector can be expressed in terms of the orthonormal basis $\{|e_i\rangle\}_i$ as $\sum_i \langle e_i | \psi \rangle |e_i\rangle$. Substituting this form into Eq. 6.8 above will give:

$$\langle A \rangle_\psi = \sum_i \langle \psi | e_i \rangle \langle e_i | A | e_i \rangle \langle e_i | \psi \rangle = \sum_i |\langle e_i | \psi \rangle|^2 e_i \tag{6.10}$$

According to the *Born rule*, the quantity $|\langle e_i | \psi \rangle|^2$ gives the probability of the measurement to return eigenvalue e_i; in other words, the result is the average of all potential values of the observable weighted by their probabilities with respect to $|\psi\rangle$. It can be shown that for a density matrix $\rho = \sum_j p_j |\psi_j\rangle\langle\psi_j|$ expressing a mixed state, and an observable A with eigen-decomposition $A = \sum_i e_i |e_i\rangle\langle e_i|$, the measurement takes the following form:

$$\langle A \rangle_\rho = \sum_j p_j \langle \psi_j | A | \psi_j \rangle = \sum_j \sum_i p_j e_i |\langle e_i | \psi_j \rangle|^2 = \mathrm{Tr}(\rho A) \tag{6.11}$$

where Tr denotes trace. From a statistical point of view, this is the average of the expectation values of A for each one of the pure states ψ_i, weighted by the corresponding probabilities.

Note that a density matrix, being a self-adjoint operator itself, can serve as an observable with respect to another density matrix describing the state of a different system. We can interpret such a measurement as the probability of the state expressed by the first matrix to be the configuration corresponding to the second one. On the linguistics side, taking the trace of $\rho_1 \rho_2$ when those two operators describe words as detailed above, can be seen as a measurement of how probable it is for the two words to occur in the same context. In other words, we have a similarity notion that will take the place of the usual inner product we have been using until now. Observe in Eq. 6.11 that during this computation two levels of averaging occur in parallel: the inner sum (indexed by i) is responsible for the superposition part over the pure states, hence our means to handle polysemy cases; on the other hand, the outer sum (indexed by j) carries the statistical average related to the homonymous cases.

There is one last concept to discuss before proceeding to formalize all these ideas in a more rigorous way; that of a *partial trace*. Assuming a composite quantum system represented by a density matrix $\rho \in \mathcal{H}_A \otimes \mathcal{H}_B$, we might want to obtain a measurement of an observable M with respect to one of the subsystems, say A. For these cases we compute the partial trace as $\mathrm{Tr}(\rho(M \otimes 1_B))$, an operation that "destroys" system A but leaves system B intact. Recall that, in our categorical framework, composite systems such as the above correspond to words with complex types, such as adjectives and verbs. It turns out that in the quantum setting this notion of partial trace serves as the composition function between words, replacing the ϵ-maps of the "classical" setting. Intuitively, a "measurement" of a noun serving as the subject of some intransitive verb, with respect to the density matrix of that verb, produces a new density matrix for the compound system that reflects the probability of the two words occurring together.

6.3.3 Operators and completely positive maps

I will now start to adapt the categorical framework of Chapter 3 in a way that adopts the quantum view of §6.3.1 and §6.3.2. First and foremost, this view requires us to be a little more precise regarding the form of our semantic space. We start from

the fact that, traditionally, a pure quantum state is a vector in a Hilbert space over the field of complex numbers $\mathbb{C}$.[3] Furthermore, for our purpose this space will be finite-dimensional. We refer to the category of finite-dimensional Hilbert spaces and linear maps as **FHilb**. This category is *dagger compact closed*, that is, equipped with an involutive contravariant functor $\dagger : \mathbf{FHilb} \to \mathbf{FHilb}$ with identity on objects. Specifically, every morphism $f : A \to B$ has a dagger which is the unique map $f^\dagger : B \to A$ satisfying $\langle f(u)|v\rangle = \langle u|f^\dagger(v)\rangle$ for all $u \in A, v \in B$. Additionally, the following diagram commutes for all objects:

$$\begin{array}{ccc} I & \xrightarrow{\epsilon_A^\dagger} & A \otimes A^* \\ & \searrow^{\eta_A} & \downarrow{\scriptstyle \sigma_{A,A^*}} \\ & & A^* \otimes A \end{array} \tag{6.12}$$

The existence of a dagger provides a categorical definition for the notion of inner product; for two state vectors $|\psi\rangle$ and $|\phi\rangle$ in a Hilbert space $\mathcal{H}$, we can now define $\langle\psi|\phi\rangle$ as the following morphism:

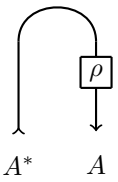
$$I \xrightarrow{\psi} \mathcal{H} \xrightarrow{\phi^\dagger} I \tag{6.13}$$

In category **FHilb**, a pure state is a morphism $\psi : I \to A$ such that $\psi^\dagger \circ \psi = 1_I$, quite similar to a word vector of meaning as those introduced in §2.2. However, our decision to adopt the density matrix representation results in a *doubling*; indeed, the density matrix associated with a pure state is the positive operator $\psi \circ \psi^\dagger : A \to A$ that sends states to states. Similarly, the density matrix of a mixed state is the operator $\rho : A \to A$. We can retrieve the original notation of representing vectors in some state space as morphisms from the unit by using the notion of *name*, which, for a mixed-state operator is defined as follows:

$$\ulcorner \rho \urcorner = I \xrightarrow{\eta^r} A^* \otimes A \xrightarrow{1_{A^*} \otimes \rho} A^* \otimes A \tag{6.14}$$

Intuitively, we can think of a name as the concrete matrix corresponding to the linear map of the specified morphism. In our graphical calculus, $\ulcorner \rho \urcorner$ has the following representation:

[3]This is because quantum mechanics deals with *wave functions*, the proper representation of which requires complex numbers. In practice, though, it should be noted that all semantic spaces for linguistic purposes are built over real numbers.

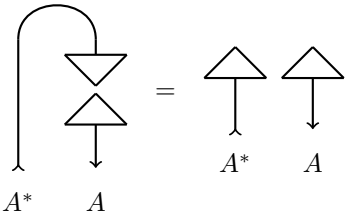


(6.15)

Note that an operator representing a pure state ψ has the following name:

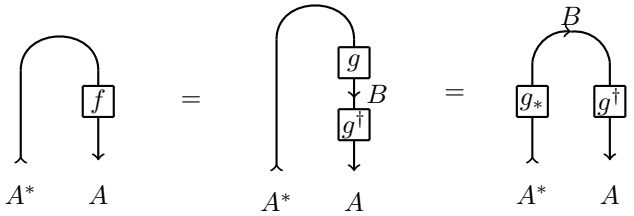
$$I \xrightarrow{\eta^r} A^* \otimes A \xrightarrow{1_{A^*} \otimes \psi^\dagger} A^* \otimes I \xrightarrow{1_{A^*} \otimes \psi} A^* \otimes A \tag{6.16}$$

which, as is evident in the graphical language, collapses to tensor product of the vector of ψ with itself.



(6.17)

Since we deal with positive operators (i.e. f is positive if it satisfies $\langle f(v)|v\rangle \geq 0$ for all v), it would be helpful to define the notion categorically: In a dagger compact closed category, a morphism $f : A \to A$ is positive when there exists an object B and a morphism $g : A \to B$ such that $f = g^\dagger \circ g$. Graphically:

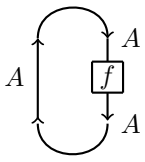


(6.18)

We also need to formally define the notion of trace, since, as we saw in §6.3.1, this will be our means to perform measurements between states; for a morphism $f : A \to A$, this takes the following form:

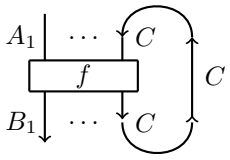
$$\mathrm{Tr}(f) = I \xrightarrow{\eta_A} A^* \otimes A \xrightarrow{\sigma_{A^*,A}} A \otimes A^* \xrightarrow{f \otimes 1_{A^*}} A \otimes A^* \xrightarrow{\epsilon_A} I \tag{6.19}$$

and, graphically:



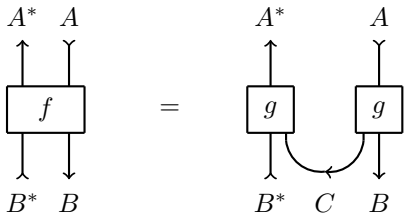
(6.20)

Similarly, we define what a partial trace is. As informally discussed before and will be more evident in §6.4, in the new framework we are introducing this takes the role of our composition function. For a morphism $A_1 \otimes \ldots \otimes A_n \otimes C \to B_1 \otimes \ldots \otimes B_n \otimes C$ it is defined as:



(6.21)

Recasting the categorical framework of [25] to the density matrices formalism requires another important step: the morphisms in **FHilb** are simple linear maps between Hilbert spaces, but what we actually need are maps that send positive operators to positive operators, while at the same time respect the monoidal structure. It turns out that the notion we are looking for is that of *completely positive maps*. Specifically, according to the *Stinespring dilation theorem* a morphism $f : A^* \otimes A \to B^* \otimes B$ is completely positive if there exists an object C and a morphism $g : A \to C \otimes B$ such that the following is true:



(6.22)

Thankfully, it has been shown that dagger compact closed categories stand in a certain correspondence with categories in which morphisms are completely positive maps, as required for our construction. This is the topic of the next section.

6.3.4 The CPM construction

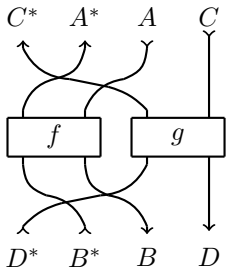
Selinger [108] has showed that any dagger compact closed category can be associated with a category in which the objects are the objects of the original category, but the maps are completely positive maps. Furthermore, he showed that the resulting category is again dagger compact closed. The *CPM construction* is exactly what we need in order to impose complete positivity in our model, and in this section I am going to detail how it can be applied to **FHilb**.

Definition 6.3.1. The category **CPM(FHilb)**:

- Has as objects finite-dimensional Hilbert spaces (exactly as **FHilb**),
- Each morphism $f : A \to B$ in **CPM(FHilb)** is a completely positive map $f : A^* \otimes A \to B^* \otimes B$ in **FHilb**; furthermore, composition of morphisms is as in **FHilb**,
- Tensor product of objects is as in **FHilb**; the tensor product of two morphisms $f : A^* \otimes A \to B^* \otimes B$ and $g : C^* \otimes C \to D^* \otimes D$ is given as below:

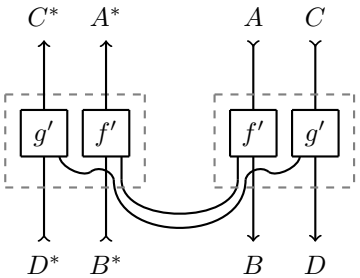
$$C^* \otimes A^* \otimes A \otimes C \xrightarrow{\cong} A^* \otimes A \otimes C^* \otimes C \xrightarrow{f \otimes g} B^* \otimes B \otimes D^* \otimes D \xrightarrow{\cong} D^* \otimes B^* \otimes B \otimes D \tag{6.23}$$

Graphically:



(6.24)

Eq. 6.23 and Diagram (6.24) above make use of the fact that **FHilb** is a symmetric dagger compact closed category, equipped with a natural isomorphism $\sigma_{A,B} : A \otimes B \to B \otimes A$. Note that the tensor product of morphisms is again a completely positive map according to Diagram (6.22); indeed, if we express the two tensored morphisms in their factorized form we get:



(6.25)

The resulting category is again dagger compact closed (see Theorem 4.20 and related proof in [108]). Table 6.1 shows how the CPM construction translates all the important morphisms we are going to need for our purposes (in general by doubling them).

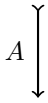
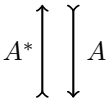
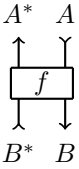
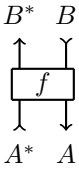
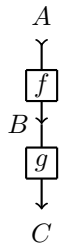
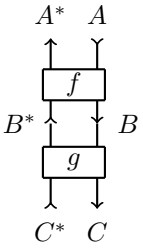
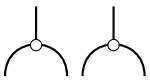
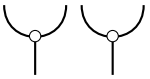
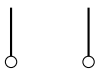
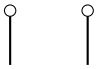
	CPM(FHilb)	FHilb		CPM(FHilb)	FHilb
1_A			f		
$f^\dagger$			$g \circ f$		
η			ϵ		
Frob. Δ			Frob. μ		
Frob. ι			Frob. ζ		

Table 6.1: The application of the CPM construction on **FHilb** using the graphical calculus. Tensor product is shown in (6.24).

6.4 An ambiguity-aware model of language

We are now ready to put together all the concepts introduced above in the context of a compositional model of meaning. We deal first with the generic case of using full (as opposed to reduced) tensors for relational words; §6.5 describes how one can apply the Frobenius framework of Chapter 4 to the new model.

Definition 6.4.1. Let $\mathcal{L} : \mathbf{FHilb} \to \mathbf{CPM(FHilb)}$ be the canonical functor applying the CPM construction as in Def. 6.3.1. Then, the passage from a grammar to distributional meaning is defined according to the following composition:

$$\mathbf{C}_F \xrightarrow{\mathcal{F}} \mathbf{FHilb} \xrightarrow{\mathcal{L}} \mathbf{CPM(FHilb)} \tag{6.26}$$

where $\mathcal{F}$ is a strongly monoidal functor that sends pregroup types from the free category generated over a partial set of atomic grammar types to finite-dimensional

Hilbert spaces in **FHilb**.

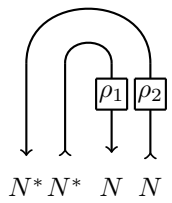
Let us now proceed and redefine what we would consider as "meaning" of a sentence in the new model.

Definition 6.4.2. Take $\rho(w_i)$ to be a state vector $I \to \mathcal{L}(\mathcal{F}(p_i))$ corresponding to operator $\mathcal{F}(p_i) \to \mathcal{F}(p_i)$ for word w_i with pregroup type p_i in a sentence $w_1 w_2 \dots w_n$. Given a type reduction $\alpha : p_1 \cdot p_2 \cdot \ldots \cdot p_n \to s$, the meaning of the sentence is defined as follows:

$$\mathcal{L}(\mathcal{F}(a))\left(\rho(w_1) \otimes \rho(w_2) \otimes \ldots \otimes \rho(w_n)\right) \tag{6.27}$$

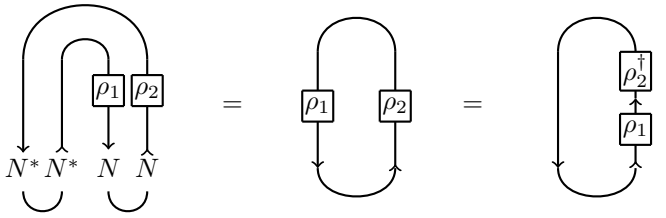
where $\otimes$ here denotes tensor product in **CPM(FHilb)**. We further require that $\rho(w_i)$ has been prepared as a proper density matrix from a statistical ensemble of pure states associated with the ith word in the sequence.

Eq. 6.27 is quite similar to our original equation of Def. 3.3.1; the major difference is that a sentence is not represented any more as the tensor product of vectors, but as the tensor product of the density matrices associated with them. It would be helpful to clarify at this point what this means exactly. Recall that tensoring any two positive maps $A^* \otimes A \xrightarrow{f} B^* \otimes B$ and $C^* \otimes C \xrightarrow{g} D^* \otimes D$ by a simple juxtaposition of them as done until now is not enough, since the resulting morphism $A^* \otimes A \otimes C^* \otimes C \xrightarrow{f \otimes g} B^* \otimes B \otimes D^* \otimes D$ will not be completely positive. Complete positivity requires from us to *embed* one object into another starting from $C^* \otimes A^* \otimes A \otimes C$, which, for two words with density matrices ρ_1 and ρ_2, leads to the following graphical representation:



(6.28)

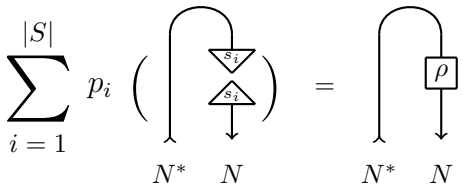
Exactly as in the original model, measuring the semantic similarity between two words is an action performed by the application of an ϵ-map, which in **CPM(FHilb)** is doubled as everything else. Note that for two words with operators ρ_1 and ρ_2, the result of this is nothing more than $\mathrm{Tr}(\rho_2^\dagger \rho_1)$:



(6.29)

I will now proceed to precisely define the semantic representations of nouns and relational words as density matrices. For what follows, I assume that every word is already associated with some statistical ensemble $S = \{(p_i, |s_i\rangle)\}_i$, where $|s_i\rangle$ is a state vector corresponding to a particular meaning that might occur in the current context with probability p_i. We can think of this ensemble as the product of a *preparation* step, a topic that we are going to examine in detail in Chapter 8.

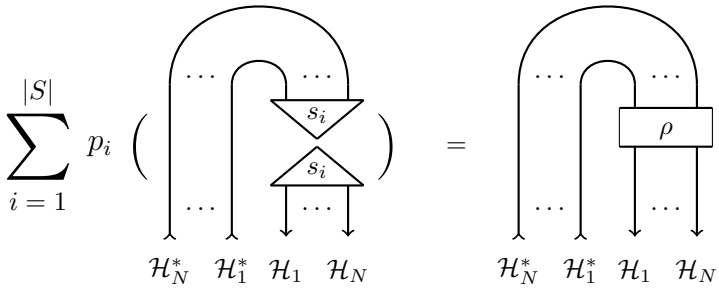
Nouns Nouns are represented as operators describing pure (if they are unambiguous) or mixed (if they have more than one meanings) states as below:



(6.30)

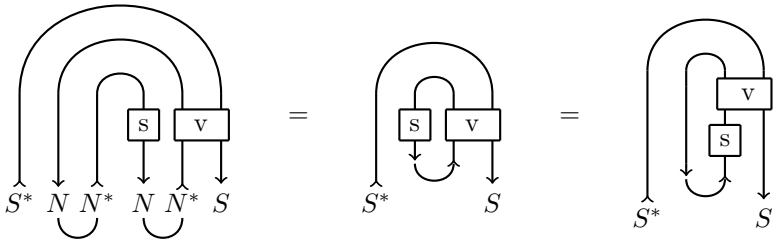
where $|S|$ is the size of the statistical ensemble associated with the word.

Relational words We think of a relational word as a N-partite system that originally lives in some tensor product of Hilbert spaces $\mathcal{H}_1 \otimes \ldots \otimes \mathcal{H}_N$. In case the word is ambiguous, the system as usual is represented by a mixed state:



(6.31)

We have now all the pieces we need to compute the meaning of a sentence or a phrase. For a simple intransitive sentence, the composition has the following form:

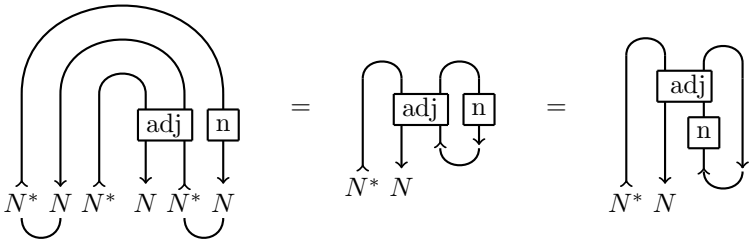


(6.32)

With the help of diagrammatic calculus, it is now clear why partial trace has the role of the composition function in the new setting. We can think of $\rho(s) \otimes 1_S$ as our "observable", which, when measured with respect to the composite system representing the verb, will return a new reduced state of $\rho(v)$ on the right-hand subsystem. This reduced $\rho(v)$ is calculated as:

$$\rho_{IN} = \mathrm{Tr}(\rho(v) \circ (\rho(s) \otimes 1_S)) \tag{6.33}$$

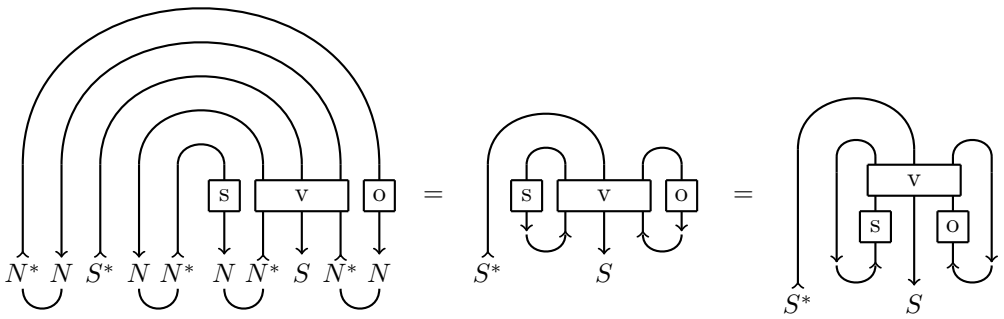
and serves as the "meaning" of the intransitive sentence. The case of an adjective-noun compound is quite similar, with the difference that now the partial trace takes place on the right:



(6.34)

$$\rho_{AN} = \mathrm{Tr}(\rho(adj) \circ (1_N \otimes \rho(n))) \tag{6.35}$$

Finally, the meaning of a transitive sentence is computed by taking two partial traces, one for the subject dimension of the verb and one of the object dimension:



(6.36)

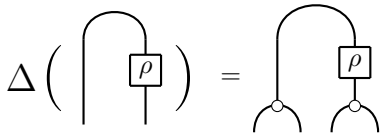
This results in the following computation:

$$\rho_{TS} = \mathrm{Tr}(\rho(v) \circ (\rho(s) \otimes 1_S \otimes \rho(o))) \tag{6.37}$$

6.5 Density matrices and Frobenius algebras

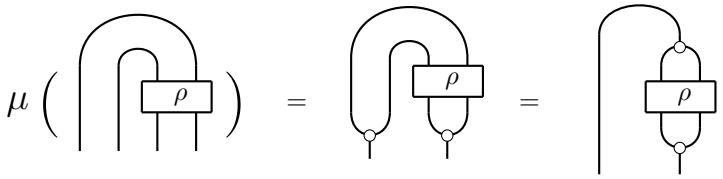
In my exposition so far I used the assumption that a verb tensor has been faithfully constructed according to its grammatical type (that is, a transitive verb tensor is of order 3, an intransitive verb tensor of order 2, and so on). In this section I will show how the Frobenius framework of Chapter 4 can be applied to our new quantum setting. Note that while the dimensionality reduction gained by applying the Frobenius operators on our original framework was a welcome extra convenience, for the quantum formulation it becomes a strictly mandatory implementation strategy: doubling an order-3 verb tensor when using a 100-dimensional *real-valued* vector space would require $100^6 \times 8 = 8 \times 10^{12}$ bytes, that is, 8 terabytes for a single density matrix. However, when one starts from a tensor of order 2 this number reduces to 800 megabytes, which, while still high, is definitely manageable. Furthermore, the Frobenius framework will also allow us to continue using the intuitive reasoning previously discussed in Chapters 4 and 5 regarding the meaning of relational and functional words.

In the context of a dagger monoidal category, a *dagger Frobenius algebra* is a Frobenius algebra in which $\Delta^\dagger = \mu$ and $\iota^\dagger = \zeta$. Extending our discussion in §3.5 from **FVect** to **FHilb**, an orthonormal basis $\{|i\rangle\}$ in a finite-dimensional Hilbert space induces a special dagger Frobenius algebra with $\Delta := |i\rangle \to |i\rangle \otimes |i\rangle$ and $\iota := |i\rangle \to 1$, while μ, ζ are obtained by taking the daggers of these, respectively. The image of this algebra in **CPM(FHilb)** is again a special dagger Frobenius algebra, which, as shown in Table 6.1, can be constructed by simply doubling the original maps. Copying a dimension from a density matrix, then, takes the following form:



(6.38)

Similarly, deletion of a dimension is performed as follows:

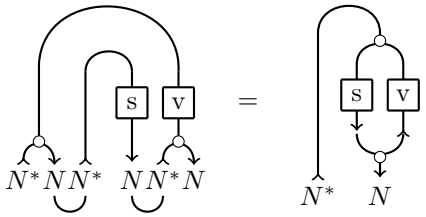


(6.39)

Assume now that we compute a density matrix of reduced dimensionality for an intransitive verb applying a variation of the argument summing procedure described in §4.1, as below:

$$\overline{verb}_{IN} = \sum_i \rho(subj_i) \tag{6.40}$$

where $\rho(subj_i)$ denotes the density matrix of the ith subject of that verb in the training corpus. The expansion of the density matrix to a tensor of order 4 to make it conform with the grammatical type $n^r \cdot s$ of the verb[4] and the subsequent composition with the subject proceeds as:



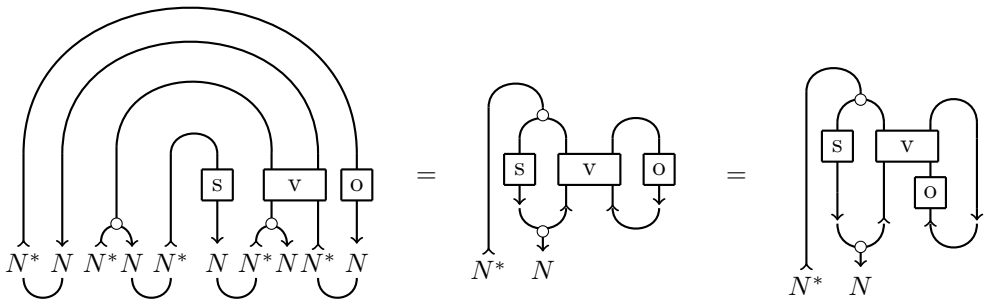
(6.41)

Note that linear-algebraically this is nothing more than the point-wise multiplication of the two density matrices $\rho(s) \odot \rho(v)$, which is exactly the behaviour of the original Frobenius formulation on tensors acting on a single argument (§4.4). Indeed, all the properties discussed in Chapter 4 that result from using Frobenius operators on simple meaning vectors are directly transferable to the new quantum setting. I will also examine the case of a transitive verb, the meaning of which can be defined as below:

$$\overline{verb}_{TR} = \sum_i \rho(subj_i) \otimes \rho(obj_i) \tag{6.42}$$

with i iterating through all occurrences of the verb in the training corpus, and $\rho(n)$ representing the density matrix produced for noun n. The application of the Copy-Subject model, then, will get the following form:

[4]Remember that in **FHilb** this would be a vector in $S^* \otimes N \otimes N^* \otimes S$.



(6.43)

From a linear algebraic perspective, we again faithfully recover the meaning of the original Copy-Subject model, since the density matrix of the subject contributes equally to the final result with the density matrix of the verb phrase:

$$\rho_{CS} = \rho(s) \odot \mathrm{Tr}(\rho(v) \circ (1_N \otimes \rho(o))) \tag{6.44}$$

Although I will not show it here, the computation proceeds similarly for the Copy-Object model with the difference that the dimension of the result is now attached to the object loop. The formula is given below:

$$\rho_{CO} = \mathrm{Tr}(\rho(v) \circ (\rho(s) \otimes 1_N)) \odot \rho(o) \tag{6.45}$$

The topic of applying Frobenius algebras on density matrices is not by any means exhausted in this section. In work that has been developed independently of (and about the same time with) the research presented in this chapter, Piedeleu [89] explores an alternative non-commutative Frobenius algebra for a density matrix formulation coming from the work of Coecke and Spekkens [27]. His analysis suggests that, for certain cases, this non-commutative version seems to be better suited to the density matrix formalism, compared to the algebra presented in this section. Furthermore, there are many other structures that adhere to the definition of a Frobenius algebra in the context of a CPM construction; this creates many opportunities for interesting combinations, the linguistic intuition of which still remains to be investigated.

6.6 Measuring ambiguity and entanglement

The quantum formulation offers to us a natural way to measure the degree of ambiguity expressed by a density matrix, in the form of *Von Neumann entropy*. For a density matrix ρ with eigen-decomposition $\rho = \sum e_i |e_i\rangle\langle e_i|$, this is given as:

$$S(\rho) = -\operatorname{Tr}(\rho \ln \rho) = -\sum_i e_i \ln e_i \tag{6.46}$$

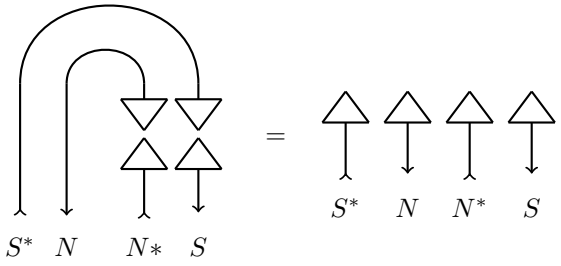
In the case of a pure state, entropy is always zero, while for a *maximally mixed state* it gets its maximum value, which is $\ln D$, with D denoting the dimension of the underlying Hilbert space. Intuitively, then, $S(\rho)$ provides an indication of the extent to which the density matrix ρ deviates from a pure state.

Recall that in our formalism a pure state corresponds to an *unambiguous* word with a single definition. For density matrices representing words, then, Von Neumann entropy offers a measure of the degree of lexical ambiguity that those words express. Even more importantly, we have now a way to measure the degree of ambiguity in "composite" density matrices representing phrases or sentences. Returning to the example given in the introduction of this chapter, measuring the level of ambiguity in the sentence "a fisherman was waiting by the bank" should return a number much lower than the one we get from the sentence "a man was waiting by the bank". This can be an insightful tool towards further research that aims at investigating the presence of ambiguity in compositional settings, and the various ways it affects the result of composition.

Regarding our discussion in §4.7, Von Neumann entropy also provides a canonical way for measuring the level of entanglement in a density matrix representing a relational word. Take $|\Psi\rangle \in A \otimes B$ to be the state of an *unambiguous* relational word and $\rho_{AB} = |\Psi\rangle\langle\Psi|$ the associated density matrix (expressing a pure state); then, the entanglement of ρ_{AB} is defined as the Von Neumann entropy of one of the two reduced matrices:

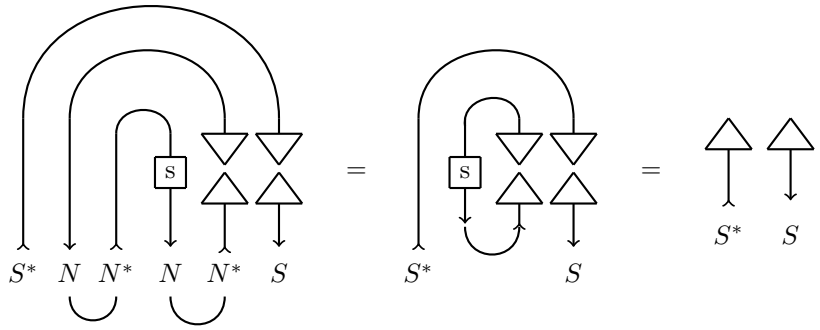
$$E(|\Psi\rangle) = S(\rho_A) = -\operatorname{Tr}(\rho_A \ln \rho_A) = -\operatorname{Tr}(\rho_B \ln \rho_B) = S(\rho_B) \tag{6.47}$$

with $\rho_A = \operatorname{Tr}_B(\rho_{AB})$ to be the partial trace of ρ_{AB} over the basis of B, and vice versa for ρ_B. Note that this is zero only when the initial state $|\Psi\rangle$ is a pure state—which would lead to the separability problems discussed in §4.7. The form of a separable intransitive verb, for example, is the following:



(6.48)

which, when composed with a subject noun, returns the following invariant outcome:



(6.49)

Computing the level of entanglement in a density matrix expressing a mixed state (in our setting, an ambiguous word) is a much more involved process. A mixed state $\rho_{AB} \in A \otimes B$ is separable if it can be factorized to the following form:

$$\rho_{AB} = \sum_i p_i \rho_i^A \otimes \rho_i^B \tag{6.50}$$

where $p_i > 0$ with $\sum_i p_i = 1$, and ρ^A, ρ^B are states of the respective subsystems. Unfortunately, finding such a factorization is an NP-hard problem, for which there exists no single standard solution. Possibilities include *distillable entanglement* and *entanglement cost*, both of which are very difficult to compute. For a concise survey of various entanglement measures, I refer the reader to the study of Plenio amd Virmani [92].

6.7 A demonstration

In this section I would like to present a small concrete example that hopefully will help readers to get a better understanding of the model. I assume a two-dimensional space with basis $\binom{0}{1}$ associated with 'finance' and $\binom{1}{0}$ associated with 'water'. We will work with the following vectors created from a hypothetical corpus:

$$|bank\rangle_f = \binom{1}{8} \quad |bank\rangle_r = \binom{9}{2} \quad |river\rangle = \binom{7}{1} \quad |fish\rangle = \binom{5}{2} \quad |money\rangle = \binom{2}{7}$$

Note that 'bank' is an ambiguous word with two meanings, one related to finance and one to water, whereas all other words are assumed unambiguous with only one definition. Normalizing those vectors will give:

$$|bank\rangle_f = \left(\begin{smallmatrix}0.12\\0.99\end{smallmatrix}\right) \quad |bank\rangle_r = \left(\begin{smallmatrix}0.98\\0.22\end{smallmatrix}\right) \quad |river\rangle = \left(\begin{smallmatrix}0.99\\0.14\end{smallmatrix}\right)$$
$$|fish\rangle = \left(\begin{smallmatrix}0.93\\0.37\end{smallmatrix}\right) \quad |money\rangle = \left(\begin{smallmatrix}0.27\\0.96\end{smallmatrix}\right)$$

Let us now prepare a density matrix for the ambiguous word 'bank', assigning equal probability to each one of the two possible meanings:

$$\rho_{bank} = \tfrac{1}{2}|bank\rangle_f\langle bank|_f + \tfrac{1}{2}|bank\rangle_r\langle bank|_r = \left(\begin{smallmatrix}0.48 & 0.17\\0.17 & 0.52\end{smallmatrix}\right)$$

We might want to measure the level of ambiguity in ρ_{bank}. Computing the Von Neumann entropy according to Eq. 6.46 gives:

$$S(\rho_{bank}) = -\operatorname{Tr}(\rho_{bank} \ln \rho_{bank}) = 0.64$$

Recall that the maximum entropy for a two-dimensional space is $\ln 2 = 0.69$. So in our example 'bank' is quite ambiguous, but not "maximally" ambiguous; in order to get a maximally mixed state, $|bank\rangle_f$ and $|bank\rangle_r$ should correspond to our basis vectors, in which case we get $\rho_{bank} = \left(\begin{smallmatrix}0.5 & 0\\0 & 0.5\end{smallmatrix}\right)$ and $S(\rho_{bank}) = 0.69$.

Let us now proceed to composing a density matrix for the compound noun 'river bank' using the Frobenius framework of §6.5. We will treat the (unambiguous) word 'river' as an adjective, and we will compose it with the (ambiguous) word 'bank' as follows:

$$\rho_{river\ bank} = |river\rangle\langle river| \odot \rho_{bank} = \left(\begin{smallmatrix}0.98 & 0.14\\0.14 & 0.02\end{smallmatrix}\right) \odot \left(\begin{smallmatrix}0.48 & 0.17\\0.17 & 0.52\end{smallmatrix}\right) = \left(\begin{smallmatrix}0.47 & 0.02\\0.02 & 0.01\end{smallmatrix}\right)$$

Measuring the entropy of the composite matrix gives $S(\rho_{river\ bank}) = 0.40$, which directly reflects the fact that the meaning of the compound noun is much clearer than that of word 'bank' alone, as expected. Furthermore, we would expect that 'river bank' is semantically closer to 'fish' than to 'money'. Recall that for any two operators ρ_1 and ρ_2 representing words, the trace of $\rho_1\rho_2$ gives a number that can be interpreted as the probability of one word occurring in the vicinity of the other. Applying this notion of similarity between 'river bank' and 'fish' results in the following computation:

$$\operatorname{Tr}(\rho_{river\ bank} \circ (|fish\rangle\langle fish|)) = \operatorname{Tr}\left[\left(\begin{smallmatrix}0.47 & 0.02\\0.02 & 0.01\end{smallmatrix}\right)\left(\begin{smallmatrix}0.86 & 0.35\\0.35 & 0.14\end{smallmatrix}\right)\right] = 0.43$$

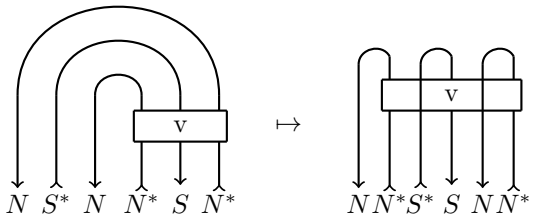
A similar computation for the similarity of 'river bank' and 'money' returns 0.06. Now assume an ambiguous vector $\overrightarrow{bank} = \overrightarrow{bank}_f + \overrightarrow{bank}_r = \binom{10}{10}$, and let us calculate the similarities between the same pairs of words when using directly the distributional vectors:

$$\cos(\overrightarrow{river} \odot \overrightarrow{bank}, \overrightarrow{fish}) = \cos\left[\binom{70}{10}, \binom{5}{2}\right] = 0.97$$
$$\cos(\overrightarrow{river} \odot \overrightarrow{bank}, \overrightarrow{money}) = \cos\left[\binom{70}{10}, \binom{2}{7}\right] = 0.41$$

Evidently, these numbers are not so sensible as the ones returned by the density matrix model. Although this toy example cannot be possibly used as an indication for the relative performance of the models, in general it is reasonable for one to expect that the richer representation imposed by the quantum model would be able to reflect better the semantic relationships between the various words, and in turn, of the various phrases and sentences after word composition.

6.8 A more convenient graphical notation

For complex sentences, the hemi-circular diagrammatic notation imposed by the special requirements of the CPM structure will quickly become inconvenient and non-informative. Following a suggestion by Bob Coecke, in this section I will show how we can almost achieve our neat original graphical representation by a change of perspective. Recall that this hemi-circular form of the diagrams is a direct consequence of the fact that the tensor product of completely positive maps imposes the embedding of one object into another; as a result, wires that belong to the same object end up to be far apart. We can avoid this problem by introducing a third dimension in our diagrammatic notation, which will allow us to represent the operators in the following manner:



(6.51)

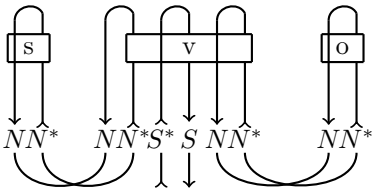
Drawing in three dimensions allows the luxury of placing objects and wires *in front* of other objects, as opposed to the strict side-by-side juxtaposition we were restricted until now. Under this new perspective, the two copies of wires that constitute the η- and ϵ-maps are not any more juxtaposed as before, but they overlap as follows (imagine that one copy has been placed in front of the other):

(6.52)

The same is true for the Frobenius operators:

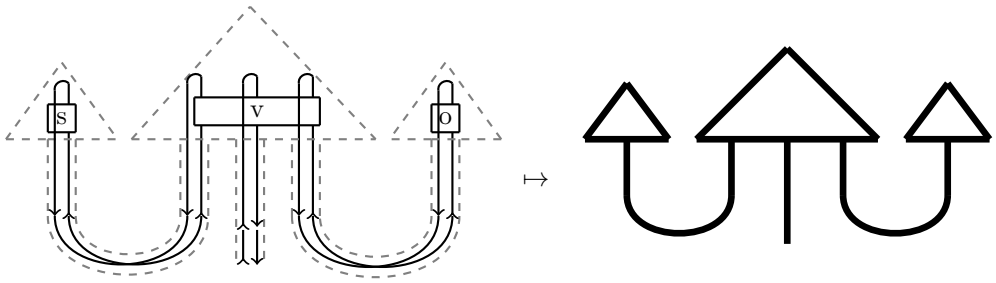
(6.53)

Composition in our new notation takes this form:



(6.54)

At this point it is not difficult to use a little visual abstraction, and essentially recover our previous notation:



(6.55)

A diagram such as (6.55) above reflects the nature of the underlying model to a much higher degree than the original notation. First, it emphasizes the most prominent characteristic of the new setting, which is the fact that everything is *doubled*. More importantly, it allows us to continue thinking about words as distinct systems that interact with each other through processes, a quality that is lost in the complex labyrinthine figures similar to that in (6.36).

6.9 Discussion

The quantum formulation presented in this chapter provides perhaps the most natural way to incorporate the notions of homonymy and polysemy in the categorical framework of [25]. The concept of an ambiguous word vector has been replaced by that of a density matrix, which expresses a probability distribution over the potential meanings of a word. While a density matrix expresses our uncertainty regarding the potential meaning of a homonymous word, smaller semantic deviations in meaning that correspond to cases of polysemy are left to be resolved by superpositions taking place over the pure states. In general, the probabilistic view imposed by the density matrix formalism can be seen as a step towards more robust tensor representations for relational words and, in turn, more effective compositional models.

Furthermore, the new perspective creates opportunities for novel interesting research. In §6.6 we briefly discussed the notion of Von Neumann entropy, and how this can be used to quantify the level of ambiguity in a single word or a composite text segment. This subject is closely related to another interesting question: in what ways can *entanglement* affect the result of composition? Or, to put things into a more practical perspective: How can we exploit this phenomenon in order to design more effective CDMs? Another very interesting characteristic of the new model is that it can accommodate more than one notion of ambiguity; this is a direct consequence of the fact that the CPM construction can be applied to *any* dagger compact closed category, even to **CPM(FHilb)** itself. In the new category, words will be represented by density matrices of density matrices; imagine, for example, that additionally to the first level of mixed states which represent a probability distribution over possible meanings of a word, we would like a second level that expresses our uncertainty regarding the syntactical use of that word (for example, 'fly' can be a noun or a verb).

The practical aspects of the material presented here will be the subject of Chapter 8. Specifically, in §8.1 and §8.2 we will see how to apply standard word sense induction and disambiguation techniques in order to create unambiguous tensors representing each distinct meaning of a homonymous word. Putting together those tensors into a statistical ensemble that can be used for preparing a density matrix for the specific word is straightforward. However, the space complexity problems posed by doubling the order of tensors (briefly discussed in §6.5) make a large-scale evaluation of the model not possible for the moment. In the second part of Chapter 8 we will conveniently side-step this difficulty, by evaluating our generic hypothesis regarding the separation of homonymy from polysemy directly on Eq. 6.2—that is, by simplify-

ing the process so that the disambiguation of a word takes place on the spot given the context, and the most probable meaning vector for each word is selected for the composition in a separate step. The results strongly suggest that the assumption upon which the quantum model was built is correct; a direct practical evaluation of the model would require the application of the Frobenius framework as proposed in 6.5, state vectors of reduced dimensionality compared to standard practice, and a carefully designed task capable of demonstrating the advantages of the formulation.

I would like to close this chapter by pointing out that the research presented here is not the only one that uses density matrices for linguistic purposes. In the context of the categorical framework of Coecke et al. [25], Piedeleu [89] independently developed a model that handles lexical ambiguity in about the same way as the current exposition; this parallel development led to a subsequent joint paper, co-authored with Piedeleu, Coecke and Sadrzadeh [90]. In yet another parallel development, Balkır [4] uses a different form of density matrices in order to provide a similarity measure that can be used for evaluating hyponymy-hypernymy relations. Blacoe et al. [10] describe a distributional (but not compositional) model of meaning based on density matrices created by grammatical dependencies. Finally, and in a more generic level not directly associated to density matrices, the application of ideas from quantum theory to language has been proved in the past a very popular field of research. Some interesting examples include the work of Bruza et al. [15] and Widdows [121].

Part III

Practice

Chapter 7

Evaluating the Frobenius Models

Chapter Abstract

We will now turn our attention to investigating the theoretical models presented in Part II from a practical perspective, providing insights about concrete implementations and discussing the inevitable difficulties that this transition from theory to practice poses. The aim of the current chapter is to provide a generic evaluation of the Frobenius framework of Chapter 4, whereas Chapter 8 deals with the more extensive topic of implementing and evaluating a prior disambiguation step in CDMs. Material is based on [53, 48, 51].

Compositional-distributional models of meaning constitute a relatively new concept in NLP research, and they inevitably lack any established evaluation procedures. In many cases (especially in deep learning settings), the CDM is trained in a way that maximizes the performance with a specific task in mind, e.g. paraphrase detection [111] or sentiment analysis [112], and is evaluated accordingly. This means that a vector representing the meaning of a sentence is much less useful for tasks other than the one it was trained for. As claimed before, though, one of the goals of this thesis is to study CDMs in a level more generic than that, independently of specific training procedures and, inevitably, of specific tasks; therefore, what we seek is the most "canonical" way to evaluate a CDM, which has to be detached from any assumptions regarding the usage of the produced sentence vectors. For a model whose purpose is to assign a certain form of meaning to a sentence, it is reasonable to claim that this task must be based on a direct evaluation of the semantic similarity between pairs of sentences or phrases.

The above reasoning presupposes something important: the existence of a set of gold-standard similarity scores, against which the computer performance can be evaluated. Usually these gold-standard scores come from a number of human annotators, the duty of which is to provide a number that, according to their opinion, reflects the semantic similarity of two sentences in a specific scale (e.g. from 1=dissimilar to 7=identical). The performance of the computer model, then, can be computed as some form of correlation (Spearman's ρ or Pearson's r) between the list of computer scores and the list of human annotations. This form of task and specific variations of it will be one of our main evaluation tools in this part of the thesis.

The models that are going to participate in the experiments are presented in Table 7.1. The Relational model corresponds to the non-scalable approach of [38], where a transitive sentence is represented by a matrix. The table also includes a non-compositional baseline (Only Verbs), in which the meaning of a sentence or phrase is just the distributional vector of the head verb. Furthermore, I test the performance of the two vector mixture models of [75], in which the meaning of a sentence is computed as the point-wise addition or multiplication of the vectors for the words therein.

Model	**Formula**
Only Verbs	$\overrightarrow{verb}$
Additive	$\overrightarrow{subj} + \overrightarrow{verb} + \overrightarrow{obj}$
Multiplicative	$\overrightarrow{subj} \odot \overrightarrow{verb} \odot \overrightarrow{obj}$
Relational	$\overline{verb} \odot (\overrightarrow{subj} \otimes \overrightarrow{obj})$
Copy-Subject	$\overrightarrow{subj} \odot (\overline{verb} \times \overrightarrow{obj})$
Copy-Object	$\overrightarrow{obj} \odot (\overline{verb}^{\mathsf{T}} \times \overrightarrow{subj})$
Frobenius additive	Copy-Subject + Copy-Object

Table 7.1: Compositional models for transitive verbs.

7.1 Head verb disambiguation

In one of the most common tasks for testing compositionality in distributional models of meaning, the sentence similarity process described in the introduction of this chapter is laid out around ambiguous verbs. Originally introduced by Kintsch [55], and later adopted by Mitchell and Lapata [75] and others, the main question that the task aims to answer is the following: Given an ambiguous verb such as 'file', to what extent can the presence of an appropriate context disambiguate its intended

meaning? The context (a subject/object pair for a transitive verb) is composed with two "landmark" verbs corresponding to different meanings (for example, 'smooth' and 'register') in order to create simple sentences. Then, each one of these sentences is compared with the sentence produced by composition of the main verb ('file') with the same context. Therefore we have two sentence pairs of the following form:

(11) a. Woman files application / Woman registers application
b. Woman files application / Woman smooths application

The assumption behind this methodology is that a good compositional model should be able to reflect the fact that the sentences of pair (11a) are closer to each other than the sentences of pair (11b). In other words, we have a task that aims to disambiguate the head verb of a sentence, disguised as a sentence similarity task. Following a tradition that started in [40] and aims for categorical models to be evaluated on at least one form of this task, I will start my experiments by testing the models of Table 7.1 on the dataset of transitive sentences introduced by Grefenstette and Sadrzadeh in [40] (to which I will refer as G&S 2011 for the rest of the thesis). The size of this dataset is 200 pairs of sentences (10 main verbs $\times$ 2 landmarks $\times$ 10 contexts).

7.1.1 Semantic space

For this task[1] I train my vectors using ukWaC [33], a corpus of English text with 2 billion words (100m sentences). The basis of the vector space consists of the 2,000 content words (nouns, verbs, adjectives, and adverbs) with the highest frequency, excluding a list of stop words. Furthermore, the vector space is lemmatized and unambiguous regarding syntactic information; in other words, each vector is uniquely identified by a ⟨lemma,pos-tag⟩ pair, which means for example that 'book' as a noun and 'book' as a verb are represented by different meaning vectors. The weights of each vector are set to the ratio of the probability of the context word given the target word to the probability of the context word overall, as follows:

$$v_i(t) = \frac{p(c_i|t)}{p(c_i)} = \frac{\text{count}(c_i, t) \cdot \text{count}(total)}{\text{count}(t) \cdot \text{count}(c_i)} \tag{7.1}$$

where $\text{count}(c_i, t)$ refers to how many times c_i appears in the context of t (that is, in a five-word window on either side of t) and $\text{count}(total)$ is the total number of

[1]The experiments presented in this thesis took place over the course of three years, and inevitably the reader will notice some inconsistency between the various settings from task to task.

word tokens in the corpus. The choice of the particular weighting scheme (which is just PMI without the logarithm) was based on the study of Bullinaria and Levy [16], where it has been found very effective in various semantic similarity tasks.

7.1.2 Results

Each entry of the G&S 2011 dataset has the form ⟨subject, verb, object, high-sim landmark, low-sim landmark⟩. The context is combined with the verb and the two landmarks, creating three simple transitive sentences. The main-verb sentence is paired with both the landmark sentences, and these pairs are randomly presented to human evaluators, the duty of which is to evaluate the similarity of the sentences within a pair in a scale from 1 to 7. The scores of the compositional models are the cosine distances between the composite representations of the sentences of each pair. As an overall score for each model, I report its Spearman's ρ correlation with the human judgements.[2] The results are shown in Table 7.2.

Model	**Spearman's ρ**
Only verbs	0.198
Additive	0.127
Multiplicative	0.137
Relational	0.219
Copy-Subject	0.070
Copy-Object	**0.241**
Frobenius additive	0.142
Human agreement	0.600

Table 7.2: Head verb disambiguation results on the G&S 2011 dataset. Difference between Copy-Object and Relational is s.s. with $p < 0.001$.

The most successful model is the Copy-Object Frobenius variation, which suggests two things. First, it provides an indication that the object of an ambiguous verb can be more important for disambiguating that verb than the subject. Intuitively, we can imagine that the crucial factor for disambiguating between the verbs "write" "publish" and "spell" is more the object than the subject: a book or a paper can be both published and written, but a letter or a shopping list can only be written.

[2]Unless stated otherwise, in the correlation-based experiments of this thesis each human judgement is treated as a separate entry in the correlation list. Note that this in general results in much lower numbers than first averaging human judgements, and then getting the correlation with computer scores. The relative performance of a model though with regard to other models is almost identical in both methodologies.

Similarly, a word can be spelled and written, but a book can only be written. The subject in all these cases is not so important. Furthermore, the superiority of the Copy-Object model over the Relational model (in which a sentence is represented as a matrix) means that the reduction of the dimensions imposed by the Frobenius operators has a positive effect, producing a compact vector that reflects better the semantic relationships between the various sentences. As a last comment, most of the Frobenius models perform better than the two vector mixtures, which suffer from the additional disadvantage that they completely ignore the word order.

7.2 Verb phrase similarity

The head verb disambiguation task of §7.1 implicitly correlates the strength of the disambiguation effect that takes place during composition with the quality of composition; essentially it assumes that the stronger the disambiguation, the better the compositional model that produced this side-effect. Unfortunately, the extent to which this assumption is valid or not is not quite clear.

In order to avoid this problem, this second experiment is based on a phrase similarity task introduced by Mitchell and Lapata [76] as part of a larger test-bed. In contrast to the task of §7.1, this one does not involve any assumptions about disambiguation, and thus it seems like a more genuine test of models aiming to provide appropriate phrasal or sentential semantic representations; the only criterion is the degree to which these models correctly evaluate the *similarity* between pairs of sentences or phrases. I work on the verb phrase part of the dataset (which from now on I will denote as M&L 2010), consisting of 72 short verb phrases (verb-object structures). These 72 phrases have been paired in three different ways to form groups exhibiting various degrees of similarity: the first group contains 36 pairs of highly similar phrases (e.g. *produce effect-achieve result*), the pairs of the second group are of medium similarity (e.g. *write book-hear word*), while a last group contains low-similarity pairs (*use knowledge-provide system*)—a total of 108 verb phrase pairs. The task is again to compare the similarity scores given by the various models for each phrase pair with those of human annotators.

7.2.1 Computing the meaning of verb phrases

Before proceeding to the results, I will first work out the closed form formulas giving the meaning of a verb phrase for the various categorical models. I remind the reader

that in the canonical case of a full (order-3) verb tensor, the verb phrase meaning is simply given by:

$$\overline{vp} = \overline{verb} \times \overrightarrow{obj} \tag{7.2}$$

which returns a matrix. For the Frobenius models, let us assume a verb matrix $\overline{verb} = \sum_{ij} v_{ij} \overrightarrow{n_i} \otimes \overrightarrow{n_j}$ and an object vector $\overrightarrow{obj} = \sum_j o_j \overrightarrow{n_j}$. Then, for the Relational case we have the following:

$$\begin{aligned} (1_{N\otimes N\otimes N} \otimes \epsilon_N^l) \circ (\Delta_N \otimes \Delta_N \otimes 1_N)(\overline{verb} \otimes \overrightarrow{obj}) &= \sum_{ij} v_{ij} \langle \overrightarrow{n_j} | \overrightarrow{obj} \rangle \overrightarrow{n_i} \otimes \overrightarrow{n_i} \otimes \overrightarrow{n_j} \\ &= \sum_{ij} v_{ij} o_j \overrightarrow{n_i} \otimes \overrightarrow{n_i} \otimes \overrightarrow{n_j} \end{aligned} \tag{7.3}$$

which is just the vector $\overline{verb} \times \overrightarrow{obj}$ expanded to a cube. When one follows the Copy-Subject model, the verb phrase meaning is again the same vector, but this time expanded to a matrix:

$$\begin{aligned} (1_{N\otimes N} \otimes \epsilon_N^l) \circ (\Delta_N \otimes 1_N \otimes 1_N)(\overline{verb} \otimes \overrightarrow{obj}) &= \sum_{ij} v_{ij} \langle \overrightarrow{n_j} | \overrightarrow{obj} \rangle \overrightarrow{n_i} \otimes \overrightarrow{n_i} \\ &= \sum_{ij} v_{ij} o_j \overrightarrow{n_i} \otimes \overrightarrow{n_i} \end{aligned} \tag{7.4}$$

Finally, the Copy-Object case returns a matrix as follows:

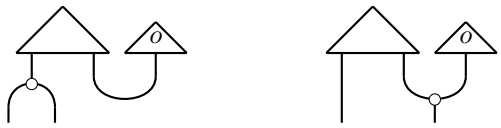
$$\begin{aligned} (1_{N\otimes N} \otimes \epsilon_N^l) \circ (1_N \otimes \Delta_N \otimes 1_N)(\overline{verb} \otimes \overrightarrow{obj}) &= \sum_{ij} v_{ij} \langle \overrightarrow{n_j} | \overrightarrow{obj} \rangle \overrightarrow{n_i} \otimes \overrightarrow{n_j} \\ &= \sum_{ij} v_{ij} o_j \overrightarrow{n_i} \otimes \overrightarrow{n_j} \end{aligned} \tag{7.5}$$

Note that this is essentially the point-wise product (also known as Hadamard product) of the verb matrix with a matrix whose rows correspond to the vector of the object:

$$\begin{pmatrix} v_{11} & \dots & v_{1n} \\ \vdots & \ddots & \vdots \\ v_{n1} & \dots & v_{nn} \end{pmatrix} \odot \begin{pmatrix} o_1 & \dots & o_n \\ \vdots & \ddots & \vdots \\ o_1 & \dots & o_n \end{pmatrix} \tag{7.6}$$

7.2.2 Results

For this experiment I used the same semantic space as in §7.1.1. This time (and in the absence of any ambiguity) the results presented in Table 7.3 show top performance from the Copy-Subject variation. Recall from Eq. 7.4 that in this model a vector is prepared for each verb phrase "categorically", by matrix-multiplying the verb matrix with the object vector; on the other hand, in the Copy-Object model the object vector is merged (by point-wise multiplication) with the object dimension of the verb matrix, as follows:



Copy-Subject Copy-Object (7.7)

We can think of the matrix produced by the Copy-Object case as representing the meaning of an *incomplete sentence*, which can take definite shape only after the introduction of the missing subject. From this perspective, the better performance of the Copy-Subject model is justified: the verb phrase here has the role of a *complete theme*, so the categorically prepared composite vector reflects the meaning of the verb phrase in a more self-contained way.

Model	**Spearman's ρ**
Only verbs	0.310
Additive	0.291
Multiplicative	0.315
Copy-Subject/Relational	**0.340**
Copy-Object	0.290
Frobenius additive	0.270
Human agreement	0.550

Table 7.3: Verb phrase similarity results on the M&L 2010 dataset. Difference between Copy-Subject and Multiplicative is s.s. with $p < 0.001$.

7.3 Classification of definitions to terms

The sentence comparison methodology I adopted in this chapter suffers from an important problem: *subjectivity*. Humans rarely agree on their judgements, whereas in many cases their evaluations differ drastically. In the paper that introduced the

Frobenius models [51] (joint work with Sadrzadeh and Pulman), I and my co-authors propose a way to cope with this difficulty. The suggestion was based on the idea that the meaning of certain phrases is expected to be close to the meaning of certain words, as in the case of a dictionary entry: the meaning vector for a definition, such as "woman who rules a country", must be close to the word vector of the term ('queen'), and far away from vectors of other irrelevant words (e.g. 'window'). A setting like this eliminates the necessity of gold-standard scores, since the evaluation can be approached as a normal classification task: each term is assigned to the closest definition according to the cosine distance of their vectors, and the result is expressed by the usual means of information retrieval: precision/recall and F-score, or accuracy. This idea also appears in the work of Turney [119], where bigrams corresponding to noun compounds (e.g. 'dog house') are compared with single nouns ('kennel').

In this section I describe two relevant experiments which were based on a dataset I created in collaboration with M. Sadrzadeh.[3] The dataset consists of 112 terms (72 nouns and 40 verbs) that have been extracted from Oxford Junior Dictionary [105] together with their main definitions. For each term we added two more definitions, either by using entries of WordNet for the term or by simple paraphrase of the main definition using a thesaurus, getting a total of three definitions per term. In all cases a definition for a noun term is a noun phrase, whereas the definitions for the verb terms consist of verb phrases. A sample of the dataset entries can be found in Table 7.4.

Term	**Main definition**	**Alternative definition 1**	**Alternative definition 2**
blaze	large strong fire	huge potent flame	substantial heat
husband	married man	partner of a woman	male spouse
foal	young horse	adolescent stallion	juvenile mare
horror	great fear	intense fright	disturbing feeling
apologise	say sorry	express regret or sadness	acknowledge shortcoming or failing
embark	get on a ship	enter boat or vessel	commence trip
vandalize	break things	cause damage	produce destruction
regret	be sad or sorry	feel remorse	express dissatisfaction

Table 7.4: A sample of the dataset for the term/definition classification task (noun terms at the top part, verb terms at the lower part).

As is evident from the samples, besides the elimination of gold-standard scores the specific setting has another important advantage for our purposes: abandoning the restricted syntactic structure of the previous experiments provides us the opportunity

[3]This and all other datasets introduced in the present thesis are available at `http://www.cs.ox.ac.uk/activities/compdistmeaning/`.

to apply in practice a number of suggestions from Chapter 5 regarding the treatment of functional words. Specifically, in order to create composite vectors for the definitions I prepare tensors for prepositions (§5.2), phrasal verbs (§5.3), and conjunctions (§5.4), as described in the referred sections.

Note that the nature of the experiment imposes a treatment of verb phrases different than that of §7.2.1. Since now our goal is to compare the meaning of phrases with that of words, as expressed by their distributional vectors, both noun- and verb phrases must live in our basic vector space N. Recall from the discussion in Chapter 3 that this is the case only for atomic types, which means that now, not only a noun phrase, but also a verb phrase must be treated accordingly. Assuming the atomic type j for a verb phrase, a verb that expects a noun at object position in order to return a verb phrase gets the type $j \cdot n^r$, which means that its tensor is a matrix in $N \otimes N$. Following the argument summing procedure, we can construct these by summing over all distributional vectors of objects with which the verb occurs in the corpus, and then expand this vector to a matrix as usual; that is,

$$\overline{verb} = \Delta\left(\sum_i \overrightarrow{obj_i}\right) \tag{7.8}$$

This is referred to as the Frobenius model, and is compared with the additive and multiplicative models. The semantic space used in this task follows exactly the setting of §7.1.1, with the important difference that the training was based on the BNC corpus, which is magnitudes smaller than ukWaC.

In the first experiment, each term corresponds to a distinct class; in order to classify a definition to a specific term, we compare the composite vector of that definition with the distributional vectors of all terms in the dataset, and we select the term that gives the highest similarity. The results are reported in Table 7.5 as the weighted average of F1-scores. The performance on the noun terms is evaluated separately from the performance on the verb terms, since a mixing of the two sets would be inconsistent.

	Nouns			**Verbs**		
Model	P	R	F1	P	R	F1
Additive	0.21	0.17	0.16	0.28	0.25	0.23
Multiplicative	0.21	0.22	0.19	0.31	0.30	0.26
Frobenius	0.22	0.24	**0.21**	0.32	0.28	**0.27**

Table 7.5: Results of the term/definition comparison task.

The Frobenius model delivers again the best performance, although the difference from the multiplicative model in the case of verbs is small. All models perform better on the verb terms than the noun part of the dataset, yet in general F-scores tend to be low. In order to get a better insight regarding the potential of the model, we perform a second experiment where the direction of comparison is reversed: now we compare each term with every *main* definition, constructing a list of definitions ranked by similarity, then examine the position of the correct definition inside this list. The detailed results are shown in Table 7.6.

		Multiplicative			**Frobenius**		
	Rank	Count	%	%*	Count	%	%*
Nouns	1	26	36.1	36.1	25	34.7	34.7
	2-5	20	27.8	63.9	22	30.6	65.3
	6-10	11	15.3	79.2	5	6.9	72.2
	11-72	15	20.8	100.0	20	27.8	100.0
Verbs	1	15	37.5	37.5	8	20.0	20. 0
	2-5	10	25.0	62.5	13	32.5	52.5
	6-10	6	15.0	77.5	4	10.0	62.5
	11-40	9	22.5	100.0	15	37.5	100.0

Table 7.6: Results of the term/definition comparison task based on the rank of the main definition.

This new perspective shows that for the noun-term set the Frobenius model returns the correct main definition in 25 of the 72 cases (34.7%), whereas in 47 cases (65%) the correct definition is in the top-five list for that term. The multiplicative model performs similarly, and better for the verb-term set. Furthermore, an error analysis revealed that some of the "misclassified" cases can also be considered as somehow "correct". For example, the definition originally assigned to the term 'jacket' was 'short coat'; however, the system "preferred" the definition 'waterproof cover', which is also correct. Some other interesting cases are presented in Table 7.7.

Term	**Original definition**	**Assigned definition**
rod	fishing stick	round handle
jacket	short coat	waterproof cover
mud	wet soil	wet ground
labyrinth	complicated maze	burial chamber

Table 7.7: A sample of ambiguous cases where the model assigned a different definition than the original.

7.4 Discussion

The experimental work presented in this chapter suggests that the Frobenius setting forms a robust implementation basis for the categorical framework of [25]. Compared with the original implementation of [40] (Relational model), it improves the results while at the same time overcomes its shortcomings, which all stemmed from the fact that logical and concrete types did not match. Furthermore, this is despite the fact that in the original model a sentence is assigned a richer representation, with exponentially higher number of parameters.

For consistency reasons, the results presented in this chapter were selected from work that uses the same experimental setting with the research I present in Chapter 8 (specifically, the experiments in §7.1, §7.2 and §8.4 follow the setting of [48]). However, it should be noted that the same general performance pattern between the models has been verified in many other cases. For example, one parameter that can drastically affect the performance of a CDM is the choice of the vector space. In a recent comparative study I jointly conducted with Milajevs, Sadrzadeh and Purver [73], we showed that neural word representations (briefly discussed in §2.2.3) present performance superior to that of co-occurrence vectors for tensor-based settings and tasks similar to those I use in the current chapter. Using a vector space[4] trained with the skip-gram model of Mikolov et al. [72], for example, we report results for the G&S 2011 task as shown in Table 7.8.

Model	**Spearman's ρ**
Only verbs	0.107
Additive	0.149
Multiplicative	0.095
Relational	0.362
Copy-Subject	0.131
Copy-Object	**0.456**
Frobenius additive	0.359
Human agreement	0.600

Table 7.8: Results on the G&S 2011 dataset using neural embeddings. Difference between Copy-Object and Relational is s.s. with $p < 0.05$. Correlation is reported on *averaged* human judgements.

The numbers clearly follow a pattern similar to that of Table 7.2. However, now the score of the Copy-Object model is very close to the current state-of-the-art for this

[4]The vector space is provided as part of the `word2vec` toolkit, which can be found online at `https://code.google.com/p/word2vec/`.

dataset, which is 0.47 and has been reported by Tsubaki et al. [118] who use a deep-learning model for composition. Furthermore, note that this time the performance of the Frobenius Additive and Relational models converges. This is encouraging, since it suggests that the vector produced by the sum of all Frobenius vectors is at least as expressive as the matrix of the Relational model in modelling the meaning of a declarative sentence. Similar observations (with small variations) have been reported by me and Sadrzadeh in [49], where we used yet another form of vector space, based on singular value decomposition.

A second observation stemming from the current work is that in general the categorical models present higher performance than that of vector mixtures. Given the word order-forgetting nature of these models, this is quite encouraging. However, the results are less conclusive regarding this matter. As has been reported in the past [11], models based on simple element-wise operations between vectors can pose an important challenge even for non-linear settings with the highest theoretical power. This fact will be more evident in Chapter 8, where there will be cases in which the best performance in some experiment comes from a vector mixture model.

Chapter 8

Prior Disambiguation in Practice

Chapter Abstract

This chapter proposes an algorithm for the implementation of a prior disambiguation step in tensor-based models of meaning, and it details a specific way to instantiate it. Based on this work, I proceed and experimentally verify the prior disambiguation hypothesis laid out in Chapter 6 in a number of natural language processing tasks. The experimental work is extended to every CDM class with positive results, and the outcome is discussed at a technical level. Material is based on [52, 48, 47, 19]

In Chapter 6, I detailed a theory regarding the proper treatment of lexical ambiguity in CDMs, and specifically in tensor-based models of meaning. The purpose of the current chapter is to make the abstract ideas expressed there more concrete, and to provide practical evidence based on real-world tasks for their validity. Although tensor-based models and, specifically, the categorical framework of [25], still remain our main point of interest, the experiments and the discussion that will follow are not restricted solely to these cases. I am particularly interested to investigate how the proposed prior disambiguation methodology affects every model class of the taxonomy presented in §4.9—that is, vector mixtures, partial tensor-based models, tensor-based models, and deep learning models. The discussion that follows the experimental work attempts a meaningful comparison of each CDM's behaviour, and hopefully draws some useful conclusions based on the special nature of each CDM class.

Recall that the central point of the methodology discussed in Chapter 6 was that each word is assigned to a number of discrete meanings, which ideally correspond to lexicographic definitions of homonymous usages for that particular word. I will start

by detailing a method for achieving that at the level of simple vectors; later, in §8.2, I propose an algorithm to extend this to relational tensors of any arity.

8.1 Generic word sense induction and disambiguation

The notion of word sense induction, that is, the task of detecting the different meanings under which a word appears within a text, is related to that of distributional semantics through the distributional hypothesis—that the meaning of a word is always context-dependent. The assumption is, for example, that the word 'bank' as a financial institution would appear in quite different contexts than as land alongside a river. If we had a way to create a vectorial representation for the contexts in which a specific word occurs, then, a clustering algorithm could be applied in order to create groupings of these contexts that hopefully reveal different usages of the word—different meanings—in the training corpus.

This intuitive idea was first presented by Schütze [107] in 1998, and more or less is the cornerstone of every unsupervised word sense induction and disambiguation method based on semantic word spaces up to today. The approach I use is a direct variation of this standard technique, close to the multi-prototype models of Reisinger and Mooney [99]. For what follows, I assume that each word in the vocabulary has already been assigned to an *ambiguous* semantic vector by following typical distributional procedures, for example similar to the setting described in §7.1.1.

First, each context for a target word w_t is represented by a *context vector* of the form $\frac{1}{n}\sum_{i=1}^{n}\overrightarrow{w_i}$, where $\overrightarrow{w_i}$ is the semantic vector of some other word $w_i \neq w_t$ in the same context. Next, I apply the hierarchical agglomerative clustering method on this set of vectors in order to discover the latent senses of w_t. Ideally, the contexts of w_t will vary according to the specific meaning in which this word has been used. The above procedure returns a number of clusters, each consisting of context vectors in the sense described above; the *centroid* of each cluster, then, can be used as a vectorial representation of the corresponding sense/meaning. Thus, in this model each word w is initially represented by a tuple $\langle\overrightarrow{w}, S_w\rangle$, where $\overrightarrow{w}$ is the ambiguous semantic vector of the word as created by the usual distributional practice, and S_w is a set of *sense vectors* (centroids of context vectors clusters) produced by the above procedure. This completes the *word sense induction* (WSI) step.

Being equipped with an unambiguous semantic space created as above, we can

now accomplish the disambiguation of an arbitrary word w under a context C as follows: we create a context vector $\overrightarrow{c}$ for C as above, and we compare it with every sense vector of w; the word is assigned to the sense corresponding to the closest sense vector. Specifically, if S_w is the set of sense vectors for w, $\overrightarrow{c}$ the context vector for C, and $d(\overrightarrow{v}, \overrightarrow{u})$ our vector distance measure, the preferred sense $\hat{s}$ is given by:

$$\hat{s} = \arg\min_{\overrightarrow{v_s} \in S_w} d(\overrightarrow{v_s}, \overrightarrow{c}) \tag{8.1}$$

Eq. 8.1 above offers to us a practical way for addressing the space complexity problems related to density matrices, as these were briefly discussed in §6.5; instead of having a massive operator expressing a probability distribution over all the different potential meanings of a word, it allows us to select on the spot the single most probable meaning tensor, using the local context as an oracle. For the experiments that follow later in this chapter, this method will be used as the homonymy-related function ϕ in Eq. 6.2.

8.2 Disambiguating tensors

The procedure of §8.1 above takes place at the vector level (as opposed to tensors of higher order), so it provides a natural way to create sets of meaning vectors for "atomic" words of the language, that is, for nouns. In this section I generalize the WSI and WSD methodology to relational words that live in tensor product spaces. Recall that the training of a relational word tensor is based on the set of contexts where this word occurs. Hence, the problem of creating disambiguated versions of tensors can be recast to that of further breaking down the set of contexts in such a way that each subset reflects a different sense of the word in the corpus. If, for example, S is the whole set of sentences for a word w that occurs in the corpus under n different senses, then the goal is to create n subsets $S_1, \dots S_n$ such that S_1 contains the sentences where w appears under the first sense, S_2 the sentences where w occurs under the second sense, and so on. Each one of these subsets can then be used to train a tensor for a specific sense of the target relational word.

It turns out that the required partitioning of the context set is something we have already achieved during the WSI stage. Recall that the clustering step of the procedure detailed in §8.1 provides us with n clusters of context vectors for a target word. Since in our case each context vector corresponds to a distinct sentence, the output of the clustering scheme can also be seen as n subsets of sentences, where

the word appears under different meanings. This one-to-one correspondence of the context clusters with sentence sets can be used for learning unambiguous versions of verb tensors. Let $S_1 \dots S_n$ be the sets of sentences returned by the clustering step for a verb. Then, the verb tensor for the ith sense is:

$$\overline{verb}_i = \sum_{s \in S_i} (\overrightarrow{subj_s} \otimes \overrightarrow{obj_s}) \tag{8.2}$$

where $subj_s$ and obj_s refer to the subject/object pair that occurred with the verb in sentence s. We have now re-stated the "argument summing" procedure of Eq. 4.9 in a way that creates unambiguous versions of relational tensors. Note that this approach can be generalized to any arity n as follows:

$$\overline{word}_i = \sum_{s \in S_i} \bigotimes_{k=1}^{n} \overrightarrow{arg_{k,s}} \tag{8.3}$$

where $arg_{k,s}$ denotes the kth argument of the target word in sentence s. The disambiguation process for the words in a sentence is summarized in pseudo-code form in Algorithm 8.1.

```
 1: Algorithm WSD(M = {⟨w_1, w_1→, C_1, S_1⟩, ..., ⟨w_n, w_n→, C_n, S_n⟩}) returns list
 2: ; w_i: the ith word in the sentence
 3: ; w_i→: ambiguous vector for word w_i
 4: ; C_i = {cl_1, ..., cl_k}: set of clusters for word w_i
 5: ; S_i = {s_1, ..., s_k}: set of contexts for word w_i in 1-to-1 correspondence with C_i
 6: ; list: an ordered list of the disambiguated word vectors/tensors
 7:
 8: list ← [ ]
 9: for each ⟨w_t, w_t→, C_t, S_t⟩ in M do
10:    c→ = (1/n) Σ_{w≠w_t} w→
11:    cl = arg min_{cl∈C_t} d(centroid(cl), c→)
12:    if w_t is noun then
13:       list ← list ⊕ [centroid(cl)]
14:    else
15:       tensor‾ = Σ_{cnt∈s_cl} ⊗_{arg∈cnt} arg→
16:       list ← list ⊕ [tensor‾]
17:    end if
18: end for
19: return list
```

Algorithm 8.1: The disambiguation process. The $\oplus$ symbol in lines 13 and 16 denotes list concatenation.

As is evident from the discussion so far, clustering is an essential part of every unsupervised WSI method—actually, the heart of it. Before I proceed to start detailing the experimental work in §8.4, I would like to discuss how I approached this important topic in some depth. The following exposition builds on my first paper related to the prior disambiguation hypothesis, co-authored with Sadrzadeh and Pulman [52].

8.3 Clustering

The choice of a clustering algorithm is an important decision in the overall process, and involves a selection ranging from approaches such as the well-known k-means algorithm and hierarchical clustering to more advanced probabilistic methods like Bayesian mixture models. Deciding which approach is the most appropriate for a specific case might depend on a number of factors, for example the size and the form of the dataset or the available computational resources. In practice it has been found that although Bayesian approaches tend to give better results, simpler methods like k-means usually follow very closely. Brody and Lapata [14], for example, report an F-score of 87.3 on the SEMEVAL 2007 task for their sense induction approach based on Latent Dirichlet Allocation, not much higher than the 84.5 F-score reported by Pedersen [86] for the same task, who creates meaning vectors using k-means.

Since my intention here is not to construct a cutting-edge word sense induction system, but rather to provide a proof of concept for the benefits of prior disambiguation in CDMs that would not be tied to any specific clustering algorithm, I will use a generic iterative bottom-up approach known as *hierarchical agglomerative clustering* (HAC). Hierarchical clustering has been invariably applied for unsupervised word sense discrimination on a variety of languages, generally showing good performance—see, for example, the comparative study of [13] for English and Polish. Compared to k-means clustering, this approach has the major advantage that it does not require us to define in advance a specific number of clusters. Compared to more advanced probabilistic techniques, such as Bayesian mixture models, it is much more straightforward and simple to implement.[1]

HAC is a bottom-up method of cluster analysis, starting with each data point (context vector in our case) forming its own cluster; then, in each iteration the two closest clusters are merged into a new cluster, until all points are finally merged under the same cluster. This process produces a *dendrogram* (i.e. a tree diagram),

[1] Actually, informal experimentation with robust probabilistic methods, such as Dirichlet process Gaussian mixture models, revealed no significant benefits for my purposes.

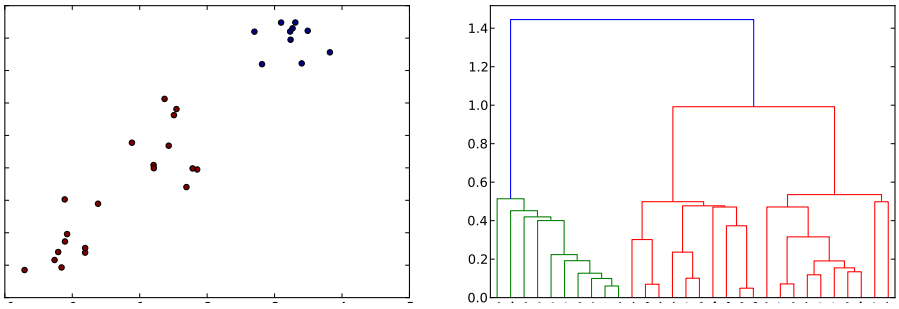


Figure 8.1: Hierarchical agglomerative clustering.

which essentially embeds every possible clustering of the dataset. As an example, Figure 8.1 shows a small dataset produced by three distinct Gaussian distributions, and the dendrogram derived by the above algorithm. Implementation-wise, I use the `fastcluster` library [80], which provides $O(n^2)$ implementations for all the methods relevant to this work, and comes with interfaces for R and Python.[2]

8.3.1 Choosing the number of clusters

Detecting the number of clusters that gives the most natural partitioning of the data is not an easy task, and is the reason that many word sense induction systems, like those of [107] and [14], adopt a fixed number of senses. Of course, such a solution seems artificial and unintuitive—a good WSI system should be able to automatically detect the correct number of meanings from the underlying data. Our choice of clustering method actually makes this task easier, since the dendrogram produced by HAC is a compact representation of all possible clusterings. However, one still needs to decide where exactly to "cut" the tree in order to get the best possible partitioning of the data.

Taking as an example the dendrogram of Figure 8.1, the optimal partitioning of 3 clusters is given by cutting the tree at about 0.8. However, cutting the tree at 1.2 will give us two clusters; even worse, if we cut the tree at 0.2 we will produce 17 clusters, a way off the natural partitioning of the dataset. Although the right answer to this problem might depend on the specific level of granularity we wish to get from our clusters, we can safely assume that the optimal partitioning is the one that provides the most compact and maximally-separated clusters. One way to measure the quality of a clustering based on this criterion is the Caliński/Harabasz index [17], also known as *variance ratio criterion* (VRC). Given a set of N data points and a partitioning

[2] I am grateful to Daniel Müllner for his help and the useful suggestions regarding the proper use of his library.

of k disjoint clusters, VRC is computed as follows:

$$VRC_k = \frac{\mathrm{Tr}(\mathbf{B})}{\mathrm{Tr}(\mathbf{W})} \times \frac{N-k}{k-1} \tag{8.4}$$

Here, $\mathbf{B}$ and $\mathbf{W}$ are the inter-cluster and inner-cluster dispersion matrices, respectively, defined as:

$$\mathbf{W} = \sum_{i=1}^{k} \sum_{l=1}^{N_i} (\overrightarrow{x_i}(l) - \bar{x_i})(\overrightarrow{x_i}(l) - \bar{x_i})^{\mathsf{T}} \tag{8.5}$$

$$\mathbf{B} = \sum_{i=1}^{k} N_i (\bar{x_i} - \bar{x})(\bar{x_i} - \bar{x})^{\mathsf{T}} \tag{8.6}$$

where N_i is the number of data points assigned to cluster i, $\overrightarrow{x_i}(l)$ is the lth point assigned to this cluster, $\bar{x_i}$ is the centroid of ith cluster (the mean), and $\bar{x}$ is the data centroid of the overall dataset. Given the above formulas, the trace of $\mathbf{B}$ is the sum of inter-cluster variances, while the trace of $\mathbf{W}$ is the sum of inner-cluster variances. A good partitioning should have high values for $\mathbf{B}$ (which is an indication for well-separated clusters) and low values for $\mathbf{W}$ (an indication for compact clusters), so the higher the quality of the partitioning the greater the value of the ratio.

Compared to other criteria, VRC has been found to be one of the most effective approaches for clustering validity—see for example the comparative studies of [74] and [120]. Furthermore, it has been previously applied to word sense discrimination successfully, returning the best results among a number of other measures [106]. In the experiments that will follow, I compute the VRC for a number of different partitionings (ranging from 2 to 10 clusters), and I keep the partitioning that results in the highest VRC value as the optimal number of senses for the specific word. Note that since the HAC dendrogram already embeds all possible clusterings, the cutting of the tree in order to get a different partitioning is performed in constant time. This is quite different, for example, from k-means or other methods based on an iterative expectation-maximization approach, where in order to get a different clustering one needs to define the number of clusters beforehand and run the algorithm again.

While this method of automatically detecting the number of clusters is efficient and effective, it has one limitation: VRC needs at least two clusters in order to be computable, so this is the lowest number of senses that a word can get in our setting. Although on the surface this seems unintuitive, note that my previous comments in §6.2 regarding the distinction between cases of homonymy and cases of polysemy are also relevant here: when a word is completely unambiguous (so normally it would

be represented by a single vector instead of two), we can safely assume that the two sense vectors produced from the partitioning imposed by VRC will be very close to each other as well as to the original ambiguous word vector, so the effect for the sense selection process as expressed by Eq. 8.1 would be minimal in any case.

8.3.2 Tuning the parameters

The parameters of the clustering scheme are optimized on the noun set of SEMEVAL 2010 Word Sense Induction & Disambiguation Task [68]. Specifically, when using HAC one has to decide how to measure the distance between the clusters, which is the merging criterion applied in every iteration of the algorithm, as well as the measure between the data points, i.e. the individual vectors. A number of empirical tests I conducted limited the available options to two inter-cluster measures: complete-link and Ward's methods. In the complete-link method the distance between two clusters X and Y is the distance between their two most remote elements:

$$D(X,Y) = \max_{x \in X, y \in Y} d(x,y) \tag{8.7}$$

In Ward's method, two clusters are selected for merging if the new partitioning exhibits the minimum increase in the overall intra-cluster variance. The cluster distance is given by:

$$D(X,Y) = \frac{2|X||Y|}{|X|+|Y|} \|\overrightarrow{c_X} - \overrightarrow{c_Y}\|^2 \tag{8.8}$$

where $\overrightarrow{c_X}$ and $\overrightarrow{c_Y}$ are the centroids of X and Y.

These *linkage* methods have been tested in combination with three vector distance measures: Euclidean, cosine, and Pearson's correlation (6 models in total). The metrics were chosen to represent progressively more relaxed forms of vector comparison, with the strictest form to be the Euclidean distance and correlation as the most relaxed. For sense detection I use Eq. 8.1, considering as context the whole sentence in which a target word appears. The distance metric used for the disambiguation process in each model is identical to the metric used for the clustering process, so in the Ward/Euclidean model the disambiguation is based on the Euclidean distance, in complete-link/cosine model on the cosine distance, and so on. I evaluate the models using V-measure, an entropy-based metric that addresses the so-called *matching problem* of F-score [101]. Table 8.1 shows the results.

Ward's method in combination with correlation distance provided the highest

Model	V-Measure	Avg clusters
Ward/Euclidean	0.05	1.44
Ward/Correlation	**0.14**	**3.14**
Ward/Cosine	0.08	1.94
Complete/Euclidean	0.00	1.00
Complete/Correlation	0.11	2.66
Complete/Cosine	0.06	1.74
Most frequent sense	0.00	1.00
1 cluster/instance	0.36	89.15
Gold standard	1.0	4.46

Table 8.1: Results on the noun set of SEMEVAL 2010 WSI&D task.

V-measure, followed by the combination of complete-link with (again) correlation. Although a direct comparison of my models with the models participating in this task would not be quite sound (since these models were trained on a special corpus provided by the organizers), it is nevertheless enlightening to mention that the 0.14 V-measure places the Ward/Correlation model at the 4th rank among 28 systems for the noun set of the task, while at the same time provides a reasonable average number of clusters per word (3.14), close to that of the human-annotated gold standard (4.46). Compare this, for example, with the best-performing system that achieved a V-measure of 0.21, a score that was largely due to the fact that the model assigned the unrealistic number of 11.54 senses per word on average (since V-measure tends to favour higher numbers of senses, as the baseline *1 cluster/instance* shows in Table 8.1).[3]

Table 8.2 provides an example of the results, showing the senses for a number of words learned by the best model of Ward's method and correlation measure. Each sense is visualized as a list of the most dominant words in the cluster, ranked by their TF-IDF values. Furthermore, Figure 8.2 shows the dendrograms produced by four linkage methods for the word 'keyboard', demonstrating the superiority of Ward's method.

8.4 Experiments on Frobenius models

I will now start presenting a series of experiments, the purpose of which is to evaluate the prior disambiguation hypothesis in concrete NLP tasks. In this section I focus on the Frobenius models of Chapter 4, so naturally my evaluation methodology follows

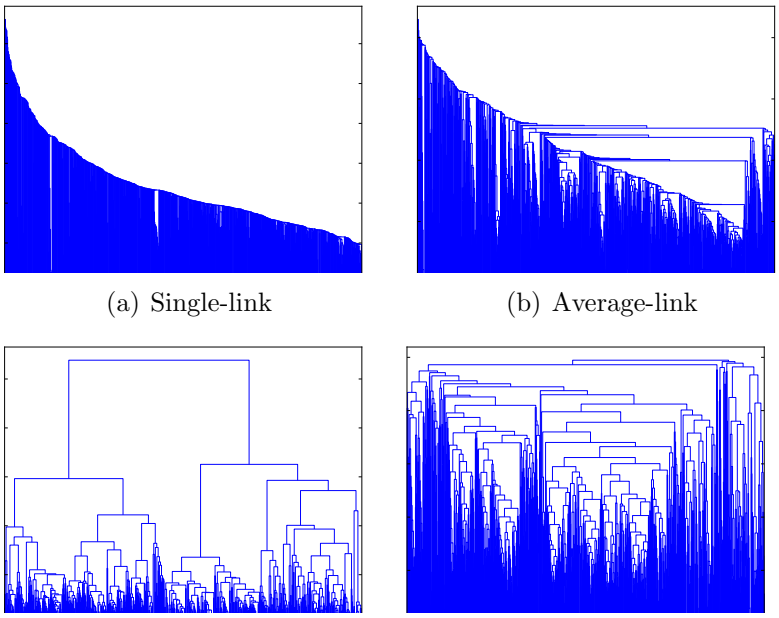
[3]The results of SEMEVAL 2010 can be found online at `http://www.cs.york.ac.uk/semeval2010_WSI/task_14_ranking.html`.

keyboard: 2 senses
COMPUTER: program dollar disk power enter port graphic card option select language drive pen application corp external editor woman price page design sun cli amstrad lock interface lcd slot notebook
MUSIC: drummer instrumental singer german father fantasia english generation wolfgang wayne cello body join ensemble mike chamber gary saxophone sax ricercarus apply form son metal guy clean roll barry orchestra

vessel: 2 senses
NAVAL: fishing state member ship company british united sea kingdom register flag owner registration community crew law boat board merchant cargo operator operate act quota man national naval fleet
MEDICAL: disease coronary grade nerve muscle vascular infarction tumour non-bleeding dilate myocardial probe patency successful angina infarct lymph haemorrhage chronic capillary stain bleeding heater clot mucosal constrict thrombosis

tank: 3 senses
GENERIC: channel upper wheel cable cupboard flight gate standard haul rail pound acid cistern programme rope tender gauge insulation incline certain set round passenger insulate ballast garden safety lead thin frame philip lift joint flotation
FISH TANK: divider swordtail well-planted tanganyikan hagen female fish-only tang gravity-fed infusorium mulm ledsam spawner scavenger devoid pair misbehave malawis cleanest synodontis bear growing-on piranha dirty nett bristleworm
MILITARY: syrian pact supreme ural station sarajevo campaign parliament israeli guerrilla secretary soviet-made emirates launcher jna cordon deployment press export times dramatic moscow mikhail motorize defender rumour uae bosnian

Table 8.2: Derived senses for various words.



(a) Single-link (b) Average-link

(c) Ward's method (d) Complete-link

Figure 8.2: Dendrograms produced for word 'keyboard' according to 4 different linkage methods.

a pattern similar to that of Chapter 7. Specifically, I perform experiments based on (a) two verb disambiguation tasks similar to the one presented in §7.1; (b) the verb phrase similarity task of §7.2; and (c) a transitive version of that task, where the verb-object structures have been extended with an appropriate subject.

In all cases the prior disambiguation methodology follows Algorithm 8.1. First, I apply the word sense induction procedure of §8.1 in order to induce meaning clusters for every word in my datasets. As discussed in §8.2, these clusters correspond to distinct sets of sentences in which the verb occurs under different meanings. Each one of these sets is used for training an "unambiguous" tensor according to Eq. 8.2, which represents a distinct usage of the verb in the corpus. The application of the prior disambiguation step then proceeds as follows: For each word in the sentence, the optimal sense is selected according to Eq. 8.1. If this word is a noun, the vector used in the composition is the centroid of the cluster that corresponds to the optimal sense, while for verbs we pick the corresponding unambiguous tensor created as above. Composition proceeds according to the models and equations of Table 7.1 on the unambiguous word representations. For ambiguous models, a tensor is trained for each verb by using the union of all sentence sets, and composition takes place as usual between those tensors and the original ambiguous word vectors for subjects and objects. All experiments that follow use the semantic space I have introduced

in §7.1.1, and have been presented in an EMNLP 2013 paper I co-authored with Sadrzadeh [48].

8.4.1 Head verb disambiguation

Since this time the hypothesis under test is directly related to disambiguation, I will perform the head verb disambiguation task of Kintsch on two different datasets: the G&S 2011 dataset from §7.1, and an additional dataset (to which I will refer as K&S 2013) that has been created by Mehrnoosh Sadrzadeh in collaboration with Edward Grefenstette, and introduced for the first time in relevant work I conducted jointly with Sadrzadeh and Pulman [52]. The main difference between the two datasets is that in the former the verbs and their alternative meanings have been selected automatically using the JCN metric of semantic similarity [44], while in the latter the selection was based on human judgements from the work of Pickering and Frisson in psycholinguistics [88]. One consequence of the selection process was that while in the first dataset many verbs are clear cases of polysemy (e.g. 'say' with meanings *state* and *allege* or 'write' with meanings *publish* and *spell*), the landmarks in the second dataset correspond to clearly separated meanings (e.g. 'file' with meanings *register* and *smooth* or 'charge' with meanings *accuse* and *bill*), i.e. they express homonymous cases. Furthermore, subjects and objects of this latter dataset are modified by appropriate adjectives. The important point for our purposes is that both datasets together cover a wide range of ambiguity cases, which can help us to draw useful results regarding the performance of the tested models.

The results for the G&S 2011 dataset are shown in Table 8.3, expressed once more as the Spearman's correlation with human judgements.[4] The Frobenius models present much better performance than the vector mixture ones, with the disambiguated version of the Copy-Object model significantly higher than the Relational model. In general, the disambiguation step improves the results of all the Frobenius models; however, the effect is drastically reversed for the vector mixtures.

The effect of disambiguation is clearer for the K&S 2013 dataset (Table 8.4). The longer context in combination with genuinely ambiguous verbs produces two effects: first, disambiguation is now helpful for all models, either vector mixtures or tensor-based; second, the disambiguation of just the verb (Only Verbs model), without any interaction with the context, is sufficient to provide the best score (0.22) with a difference statistically significant from the second model (0.19 for disambiguated

[4]For all tables in this section, $\ll$ and $\gg$ denote highly statistically significant differences with $p < 0.001$.

Model	Ambiguous		Disambiguated
Only verbs	0.198	$\gg$	0.132
Multiplicative	0.137	$\gg$	0.044
Additive	0.127	$\gg$	0.047
Relational	0.219	$<$	0.223
Copy-Subject	0.070	$\ll$	0.122
Copy-Object	**0.241**	$\ll$	**0.262**
Frobenius additive	0.142	$\ll$	0.214
Human agreement	0.599		

Table 8.3: Results for the G&S 2011 dataset. The difference between the Copy-Object and Relational model is s.s. with $p < 0.001$.

Model	Ambiguous		Disambiguated
Only verbs	**0.151**	$\ll$	**0.217**
Multiplicative	0.131	$<$	0.137
Additive	0.085	$\ll$	0.193
Relational	0.036	$\ll$	0.121
Copy-Subject	0.035	$\ll$	0.117
Copy-Object	0.033	$\ll$	0.095
Frobenius additive	0.036	$\ll$	0.114
Human agreement	0.383		

Table 8.4: Results for the K&S 2013 dataset. The difference between Only Verbs and Additive model is s.s. with $p < 0.001$

additive). In fact, further composition of the verb with the context decreases performance, confirming results I reported in previous work [52] for vectors trained using BNC. Given the nature of the specific task, which is designed around the ambiguity of the verb, this result is not surprising: a direct disambiguation of the verb based on the rest of the context should naturally constitute the best method to achieve top performance—no composition is actually necessary for this task to be successful.

8.4.2 Verb phrase/sentence similarity

The second set of experiments is based on the phrase similarity task I introduced in §7.2. Recall from the discussion in §7.2.1 that since now we deal with verb phrases, the Relational model is reduced to Copy-Subject. The results presented in Table 8.5 show that the effects of disambiguation for this task are quite impressive: the differences between the scores of all disambiguated models and those of the ambiguous versions are highly statistically significant (with $p < 0.001$), while 4 of the 6 models present

an improvement greater than 10 units of correlation. The models that benefit the most from disambiguation are the vector mixtures; both of these approaches perform significantly better than the best Frobenius model (which is Copy-Object again).

Model	**Ambiguous**		**Disambiguated**
Only verbs	0.310	$\ll$	0.420
Multiplicative	0.315	$\ll$	**0.448**
Additive	0.291	$\ll$	0.436
Relational/Copy-Subject	**0.340**	$\ll$	0.367
Copy-Object	0.290	$\ll$	0.393
Frobenius additive	0.270	$\ll$	0.336
Human agreement	0.550		

Table 8.5: Results for the M&L 2010 task. Difference between Multiplicative and Additive is not s.s; difference between Multiplicative and Only Verbs is s.s. with $p < 0.001$.

In a variation of the above experiment, my goal is to examine the extent to which the picture of Table 8.5 can change for the case of text structures longer than verb phrases. In order to achieve this, I extend each one of the 72 verb phrases to a full transitive sentence by adding an appropriate subject such that the similarity relationships of the original dataset are retained as much as possible, so the human judgements for the verb phrase pairs could as well be used for the transitive cases. I worked pairwise: for each pair of verb phrases, I first selected one of the 5 most frequent subjects for the first phrase; then, the subject of the other phrase was selected by a list of synonyms of the first subject in a way that the new pair of transitive sentences constitutes the least more specific version of the given verb-phrase pair. So, for example, the pair *produce effect/achieve result* became *drug produce effect/medication achieve result*, while the pair *pose problem/address question* became *study pose problem/paper address question*.

The restrictions of the verb-phrase version do not hold for the transitive case, so here the evaluation takes place on the full set of models (Table 8.6). Once more disambiguation produces better results in all cases, with highly statistically significant differences for all models. The most evident observation is that as the length of the text segments increases, the performance of vector mixtures and tensor-based models seems to converge. Indeed, note how the numbers of the vector mixture models are significantly lower compared to the verb phrase task.

Model	Ambiguous		Disambiguated
Only verbs	0.310	$\ll$	0.341
Multiplicative	0.325	$\ll$	0.404
Additive	0.368	$\ll$	**0.410**
Relational	**0.368**	$\ll$	0.397
Copy-Subject	0.310	$\ll$	0.337
Copy-Object	0.321	$\ll$	0.368
Frobenius additive	0.309	$\ll$	0.390
Human agreement		0.550	

Table 8.6: Results for the transitive version of M&L 2010 task.

8.4.3 Discussion

The experimental work presented here allows us to say with enough confidence that disambiguation as an additional step prior to composition is very beneficial for Frobenius models. Furthermore, it confirms and strengthen previous work [98, 52] that showed better performance of disambiguated vector mixture models compared to their ambiguous versions. Fig. 8.3 provides a summary of the results in the form of diverging stacked bar graphs.

The positive effect of disambiguation is more evident for the vector mixture models (especially for the additive model) than for the Frobenius ones. Perhaps this is not surprising: composite representations created by element-wise operations are

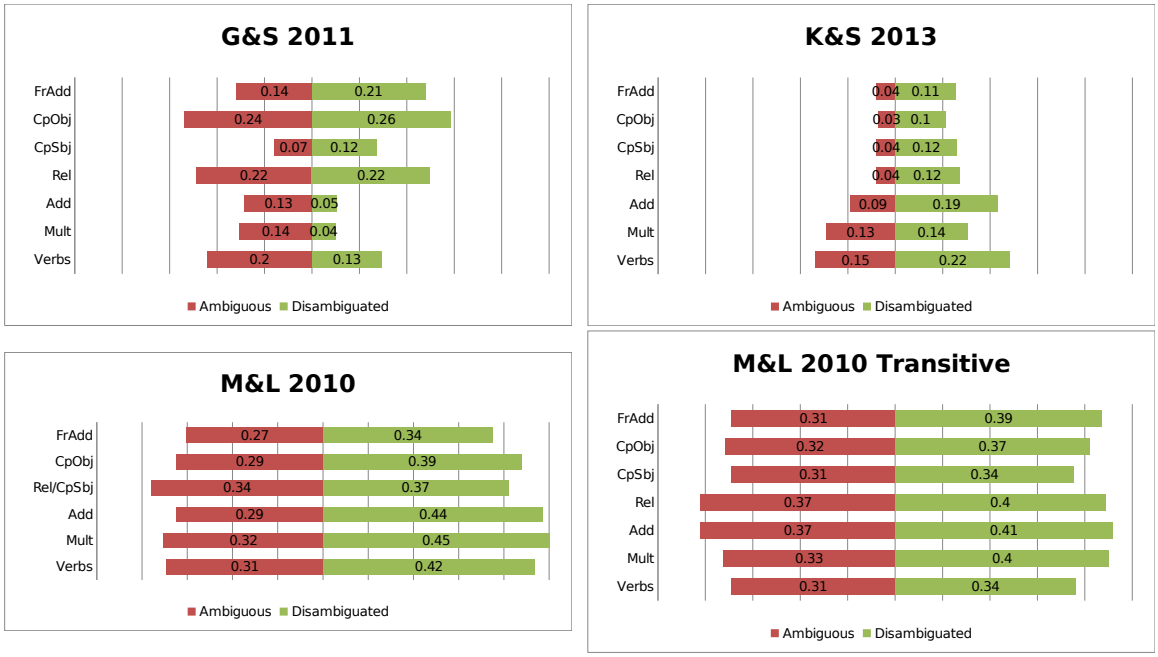


Figure 8.3: Prior disambiguation in diagrams.

averages, and the provision of more accurate starting points by a prior step of disambiguation can make a great difference.

From a task perspective, the effect of disambiguation was much more definite in the phrase/sentence similarity task. This observation is really interesting, since the words of that dataset were *not* selected in order to be ambiguous in any way. The superior performance of the disambiguated models, therefore, implies that the proposed methodology can improve tasks based on phrase or sentence similarity *regardless* of the level of ambiguity in the vocabulary. For these cases, the proposed disambiguation algorithm acts as a fine-tuning process, the outcome of which seems to be always positive; it can only produce better composite representations, not worse.

For disambiguation-based tasks similar to those of §8.4.1, the effect of prior disambiguation seems more positive for homonymous cases than for polysemous ones; hence the inferior performance of disambiguated models in the G&S 2011 dataset, compared to the K&S 2013 dataset. In fact, the G&S 2011 dataset was the only one where disambiguation was not helpful for some cases (specifically, for vector mixtures). Although the better performance under the presence of genuinely ambiguous words is rather expected, a comparison of the results between the two datasets should also take into account that the automatic selection of landmark verbs using the JCN metric (as done with the G&S 2011 dataset) was not very balanced for certain cases. Note, for example, that the bare baseline of comparing just *ambiguous* versions of verbs (without any composition) in that dataset already achieves a very high correlation of 0.20 with human judgements (Table 8.3).[5] This number is only 0.15 for the K&S 2013 dataset, due to the more effective verb selection procedure. The Only Verbs baseline can be seen as a "sanity test" for the specific task, given its design; putting composition aside, one would expect that comparing *disambiguated* versions of the verbs alone will always produce better results than those produced by comparing ambiguous ones. This sanity test is passed only by the K&S 2013 dataset.

Finally, the results presented in §8.4.2 imply that the effectiveness of vector mixture and Frobenius models depends to some extent on the length of the text segment: when more words are involved, vector mixture models tend to be less effective; on the contrary, the performance of tensor-based models seems to be proportional to the length of the phrase or sentence—the more, the better. These observations comply with the nature of the approaches: "averaging" larger numbers of points results in more general (hence less accurate) representations; on the other hand, a larger number of arguments makes a function (such as a verb) more accurate.

[5]The reported number for this baseline by [40] was 0.16 using vectors trained from BNC.

8.5 Experiments on full tensors

The experiments in §8.4 provide evidence that a prior disambiguation step on word vectors before the composition is beneficial for vector mixture models and partial tensor-based models that follow the Frobenius framework of Chapter 4. The purpose of the current section is to generalize the results for the case of full tensor-based models, where the tensor of a verb follows faithfully the grammar type without any application of Frobenius operators. In order to do that, I apply the linear regression method described in §4.8 for constructing ambiguous and disambiguated tensors of order 2 (that is, matrices) for verbs taking one argument. In principle, the methods discussed here are directly applicable to tensors of higher order, assuming one has some way to construct them efficiently.[6]

We will concentrate on elementary verb phrases of the form *verb-object* (e.g. 'play football', 'admit student'), since in general objects comprise stronger contexts for disambiguating the usage of a verb. In order to create a matrix for, say, the verb 'play', I first collect all instances of the verb occurring with some object in the training corpus, and then I create non-compositional vectors for these elementary sentences (to which I will refer as *holistic* vectors) following exactly the same methodology as if they were words. We now have a dataset with instances of the form $\langle \overrightarrow{obj_i}, \overrightarrow{play\ obj_i} \rangle$ (e.g. the vector of 'flute' paired with the holistic vector of 'play flute', and so on), that can be used to train a linear regression model in order to produce an appropriate matrix for verb 'play'. As discussed in §4.8, the premise of a model like this is that the multiplication of the verb matrix with the vector of a new object will produce a result that approximates the distributional behaviour of all these elementary two-word exemplars used in training. The work of this section has been presented in the context of ACL 2014 [47] (joint with Kalchbrenner and Sadrzadeh).

8.5.1 Experimental setting

Linear regression in general can impose long training times, so the word vectors that are going to be used as input to the algorithm must be relatively small. For this reason, this time I use a 300-dimensional semantic space that has been produced by application of SVD. In order to compute the weights of the vectors, I used local mutual information (LMI), that is, the original counts multiplied by PMI. From almost every other aspect the creation of the vector space follows the methodology of §7.1.1. In

[6] For example, Grefenstette et al. [39] create order-3 tensors for transitive verbs using a multi-step procedure based on linear regression.

order to create the vector space of the holistic verb phrase vectors, I first collected all instances where a verb participating in the experiments appeared at least 100 times in a verb-object relationship with some noun in the corpus. As context of a verb phrase I considered any content word that falls into a 5-word window from either side of the verb *or* the object. For the 68 verbs participating in the experiments, this procedure resulted in 22,000 verb phrases, a vector space that again was projected into 300 dimensions using SVD.

For each verb I use linear regression with gradient descent directly applied on matrices $\mathbf{X}$ and $\mathbf{Y}$, where the rows of $\mathbf{X}$ correspond to vectors of the nouns that appear as objects for the given verb and the rows of $\mathbf{Y}$ to the holistic vectors of the corresponding verb phrases. Our objective function then becomes:

$$\hat{\mathbf{W}} = \arg\min_{\mathbf{W}} \frac{1}{2m} \left(\|\mathbf{W}\mathbf{X}^\intercal - \mathbf{Y}^\intercal\|^2 + \lambda \|\mathbf{W}\|^2 \right) \tag{8.9}$$

where m is the number of training examples and λ a regularization parameter. The matrix $\mathbf{W}$ is used as the tensor for the specific verb.

8.5.2 Supervised disambiguation

The first experiment tests the effectiveness of a prior disambiguation step for a tensor-based model using supervised learning. The goal is to create composite vectors for a number of elementary verb phrases of the form *verb-object* with and without an explicit disambiguation step, and evaluate which model approximates better the holistic vectors of these verb phrases.

The verb phrases of our dataset are based on the 5 ambiguous verbs of Table 8.7. Each verb has been combined with two different sets of nouns that appear in a verb-object relationship with that verb in the corpus (a total of 343 verb phrases). The nouns of each set have been manually selected in order to explicitly represent a different meaning of the verb. As an example, in the verb 'play' I impose the two

Verb	**Meaning 1**	**Meaning 2**
break	violate (56)	break (22)
catch	capture (28)	be on time (21)
play	musical instrument (47)	sports (29)
admit	permit to enter (12)	acknowledge (25)
draw	attract (64)	sketch (39)

Table 8.7: Ambiguous verbs for the supervised task. The numbers in parentheses refer to the collected training examples for each case.

distinct meanings of using a musical instrument and participating in a sport; so the first set of objects contains nouns such as 'oboe', 'piano', 'guitar', and so on, while in the second set we see nouns such as 'football', 'baseball" etc.

In more detail, the creation of the dataset was done in the following way: First, all verb entries with more than one definition in the Oxford Junior Dictionary [105] were collected into a list. Next, a hired linguist (who was a native speaker of English) annotated the semantic difference between the definitions of each verb in a scale from 1 (similar) to 5 (distinct). Only verbs with definitions exhibiting completely distinct meanings (marked with 5) were kept for the next step. For each one of these verbs, a list was constructed with all the nouns that appear at least 50 times under a verb-object relationship in the corpus with the specific verb. Then, each object in the list was manually annotated as *exclusively* belonging to one of the two senses; so, an object could be selected only if it was related to a single sense, but not both. For example, 'attention' was a valid object for the *attract* sense of verb 'draw', since it is unrelated to the *sketch* sense of that verb. On the other hand, 'car' is not an appropriate object for either sense of 'draw', since it could appear under both of them in different contexts. The verbs of Table 8.7 were the ones with the highest numbers of exemplars per sense, creating a dataset of significant size for the intended task (each holistic vector is compared with 343 composite vectors).

The experiment proceeds as follows: I apply linear regression in order to train verb matrices using jointly the object sets for both meanings of each verb, as well as separately—so in this latter case we get two matrices for each verb, one for each sense. For each verb phrase, I create a composite vector by matrix-multiplying the verb matrix with the vector of the specific object. Then I use 4-fold cross validation to evaluate which version of composite vectors (the one created by the ambiguous tensors or the one created by the unambiguous ones) approximates better the holistic vectors of the verb phrases in our test set. This is done by comparing each holistic vector with all the composite ones, and then evaluating the rank of the correct composite vector within the list of results.

In order to get a proper mixing of objects from both senses of a verb in training and testing sets, I set the cross-validation process as follows: I first split both sets of objects in 4 parts. For each fold then, our training set is comprised of $\frac{3}{4}$ of set #1 plus $\frac{3}{4}$ of set #2, while the test set consists of the remaining $\frac{1}{4}$ of set #1 plus $\frac{1}{4}$ of set #2. The data points of the training set are presented in the learning algorithm in random order.

I measure approximation in three different metrics. The first one, accuracy, is the

strictest, and evaluates in how many cases the composite vector of a verb phrase is the closest one (the top one in the result list) to the corresponding holistic vector. A more relaxed and perhaps more representative method is to calculate the *mean reciprocal rank* (MRR), which is given by:

$$\text{MRR} = \frac{1}{m}\sum_{i=1}^{m}\frac{1}{rank_i} \tag{8.10}$$

where m is the number of objects and $rank_i$ refers to the rank of the correct composite vector for the ith object. Finally, a third way to evaluate the performance of each model is to simply calculate the average cosine similarity between every holistic vector and its corresponding composite vector:

$$\text{SIM}_{avg} = \frac{1}{m}\sum_{i=1}^{m}\cos(\overrightarrow{\overline{verb}\ \overline{obj}_i}, \overline{verb} \times \overrightarrow{obj}_i) \tag{8.11}$$

The results are presented in Table 8.8, reflecting a clear superiority ($p < 0.001$ for average cosine similarity) of the prior disambiguation method for every verb and every metric.

	Accuracy		**MRR**		**Avg Sim**	
	Ambiguous	Disamb.	Ambiguous	Disamb.	Ambiguous	Disamb.
break	0.19	0.28	0.41	0.50	0.41	0.43
catch	0.35	0.37	0.58	0.61	0.51	0.57
play	0.20	0.28	0.41	0.49	0.60	0.68
admit	0.33	0.43	0.57	0.64	0.41	0.46
draw	0.24	0.29	0.45	0.51	0.40	0.44

Table 8.8: Results for the supervised task.

8.5.3 Unsupervised disambiguation

In §8.5.2 I used a controlled procedure to collect genuinely ambiguous verbs and I trained my models from manually annotated data. In this section I outline how the process of creating tensors for distinct senses of a verb can be automated, and I test this idea on the generic verb phrase similarity task introduced in §7.2.

The first thing we have to do is to use the unsupervised learning method detailed in §8.3 in order to detect the latent senses of each verb in the corpus. The next step is to classify every noun that has been used as an object of that verb to the most probable verb sense, and then use these sets of nouns as before for training tensors

for the various verb senses. Being equipped with a number of sense clusters created as above for every verb, the classification of each object to a relevant sense is based on the cosine distance of the object vector from the centroids of the clusters (again, a direct application of Eq. 8.1). Every sense with less than 3 training exemplars is merged to the dominant sense of the verb. The union of all object sets is used for training a single unambiguous tensor for the verb. As usual, data points are presented to the learning algorithm in random order, and no objects in the test set are used for training.

The experiment has the following form: For every pair of verb phrases, I construct composite vectors for the phrases and then I evaluate their cosine similarity. For the ambiguous regression model, the composition is done by matrix-multiplying the ambiguous verb matrix (learned by the union of all object sets) with the vector of the noun. For the disambiguated version, I first detect the most probable sense of the verb given the noun, again by comparing the vector of the noun with the centroids of the verb clusters; then, I matrix-multiply the corresponding unambiguous tensor created exclusively from objects that have been classified as closer to this specific sense of the verb with the vector of the noun. I also test the usual baselines (Only Verbs, Additive, Multiplicative).

The results are presented in Table 8.9, where again the version with the prior disambiguation step shows performance superior to that of the ambiguous version. There are two interesting observations that can be made on the basis of Table 8.9. First and foremost, the regression model is based on the assumption that the holistic vectors of the exemplar verb phrases follow an ideal distributional behaviour that the model aims to approximate as close as possible. The results of Table 8.9 confirm this: using just the holistic vectors of the corresponding verb phrases (no composition is involved here) returns the best correlation with human annotations (0.403), providing

Model	**Spearman's ρ**
Only Verbs	0.331
Additive	0.379
Multiplicative	0.301
Linear regression (ambiguous)	0.349
Linear regression (disambiguated)	**0.399**
Holistic verb phrase vectors	0.403
Human agreement	0.550

Table 8.9: Results for the phrase similarity task. The difference between the ambiguous and the disambiguated version is s.s. with $p < 0.001$.

a proof that the holistic vectors of the verb phrases are indeed reliable representations of each verb phrase's meaning. Next, observe that the prior disambiguation model approximates this behaviour very closely (0.399) on unseen data, with a difference *not* statistically significant. This is very important, since a regression model can only perform as well as its training dataset allows it—and in our case this is achieved to a very satisfactory level.

8.6 Prior disambiguation and deep learning

Throughout this chapter I have experimentally investigated (and verified) the prior disambiguation methodology in almost every compositional model class in the hierarchy of Fig. 4.2. This section deals with the last (and theoretically most powerful) class, that of deep learning models. In joint work with Cheng and Grefenstette [19], we show that one can gain certain benefits even in the case where composition is achieved by the means of a recursive neural network, as this was described in §2.3.4. The effectiveness of prior disambiguation on deep learning compositional models is tested on the same three datasets of §8.4, used for evaluating the disambiguated Frobenius models. In terms of neural compositional models, we implement a RecNN and a RAE (more details about these models can be found in §2.3.4). Furthermore, we use additive and multiplicative models as baselines.

For each dataset and each model, the evaluation is conducted in two ways. First, we measure the Spearman's correlation between the computed cosine similarities of the composite sentence vectors and the corresponding human scores, as in the previous sections of this chapter. Second, we apply a more relaxed evaluation, based on a binary classification task. Specifically, we use the human score that corresponds to each pair of sentences in order to decide a label for that pair (1 if the two sentences are highly similar and 0 otherwise), and we use the training set that results from this procedure as input to a logistic regression classifier. As a measure of the matching rate, the 4-fold cross validation accuracy is reported. The results for each dataset and experiment are listed in Tables 8.10–8.12.

Once more the results are quite promising, since they suggest that disambiguation as an extra step prior to composition can bring at least marginal benefits to deep learning compositional models. Note that the performance of the models follows a pattern very similar to that observed in §8.4, where we evaluated the prior disambiguation hypothesis on the Frobenius models. Specifically, the effect of disambiguation is clearest for the M&L 2010 dataset, which does not contain any words

Spearman's correlation			
	Ambiguous	Disambig. every word	Disamb. only verbs
Additive	**0.221**	0.071	0.105
Multiplicative	0.085	0.012	0.043
RecNN	0.127	0.119	0.128
RAE	0.124	0.098	0.126
Cross validation accuracy			
Additive	63.07%	63.08%	62.48%
Multiplicative	61.89%	59.20%	60.11%
RecNN	62.66%	63.53%	**66.19%**
RAE	63.04%	60.51%	65.17%

Table 8.10: Results for G&S 2011 dataset.

Spearman's correlation			
	Ambiguous	Disambig. every word	Disamb. only verbs
Additive	0.132	**0.152**	0.147
Multiplicative	0.049	0.129	0.104
RecNN	0.085	0.098	0.101
RAE	0.106	0.112	0.123
Cross validation accuracy			
Additive	49.28%	51.51%	51.04%
Multiplicative	49.76%	52.37%	53.06%
RecNN	51.37%	52.64%	**59.26%**
RAE	50.92%	53.35%	59.17%

Table 8.11: Results for K&S 2013 dataset.

Spearman's correlation			
	Ambiguous	Disamb. every word	Disamb. only verbs
Additive	0.379	**0.407**	0.382
Multiplicative	0.301	0.305	0.285
RecNN	0.297	0.309	0.311
RAE	0.282	0.301	0.303
Cross validation accuracy			
Additive	56.88%	59.31%	58.26%
Multiplicative	59.53%	59.24%	57.94%
RecNN	60.17%	61.11%	**61.20%**
RAE	59.28%	59.16%	60.95%

Table 8.12: Results for M&L 2010 dataset.

chosen to be ambiguous on purpose; in both evaluations we carried out, disambiguation has a positive effect for the subsequent composition. For one more time, prior disambiguation acts as a useful pre-processing step, which might improve the final outcome (if the sentence has ambiguous words) or not (if all words are unambiguous), but never decrease the performance. The effect of disambiguation seems also to be

quite clear for the K&S 2013 dataset, whereas the result for the G&S 2011 dataset, although positive, is less definite due to the reasons discussed in §8.4.3.

8.7 Assessing the results

The work of this chapter clearly suggests that explicitly dealing with lexical ambiguity in the context of a CDM is a safe way to increase the performance of the model. Especially encouraging is the fact that the prior disambiguation methodology seems to work very well in general cases, such as the M&L 2010 task, where the majority of words are unambiguous or express small deviations in their meaning. This confirms the hypothesis I posed in §6.2, regarding the potential effect of the prior disambiguation step on polysemous or unambiguous words: since the sense vectors of such a word are very similar to each other (and to the original ambiguous vector), the extent to which the final composite vector is affected remains low. As we move from polysemous to homonymous cases and deviations in meaning get higher, the prior disambiguation step starts to contribute positively to the composite representation of the phrase or sentence.

Interestingly, this observation seems to hold for every class of CDM in the hierarcy of §4.9, although up to a different degree. Table 8.13 provides a summary of the results across models for the M&L 2010 task, which shows that the improvement we get in each case is inversely proportional to the theoretical power of the model. This is more clear in Fig. 8.4, where this improvement is expressed as percentage over the performance of the ambiguous version.

Model	**Ambiguous**		**Disambiguated**
Deep learning	0.282	$\ll$	0.301
Tensor-based	0.349	$\ll$	0.399
Partial-tensor based	0.290	$\ll$	0.393
Vector mixture	0.315	$\ll$	0.448

Table 8.13: Summary of disambiguation results for M&L 2010 dataset across CDM classes. The numbers of each class refer to Spearman's correlation for models maximally benefited from the prior disambiguation step.

While the effect of prior disambiguation is minimal for the deep learning model, it is doubled for the tensor-based approach of §8.5. Recall from the discussion in §2.3.5 that non-linear models are in a position to drastically transform the phrase or sentence space in a way that reflects better any underlying relationships. In principle, this means that a non-linear compositional model approximates the concept of continuous

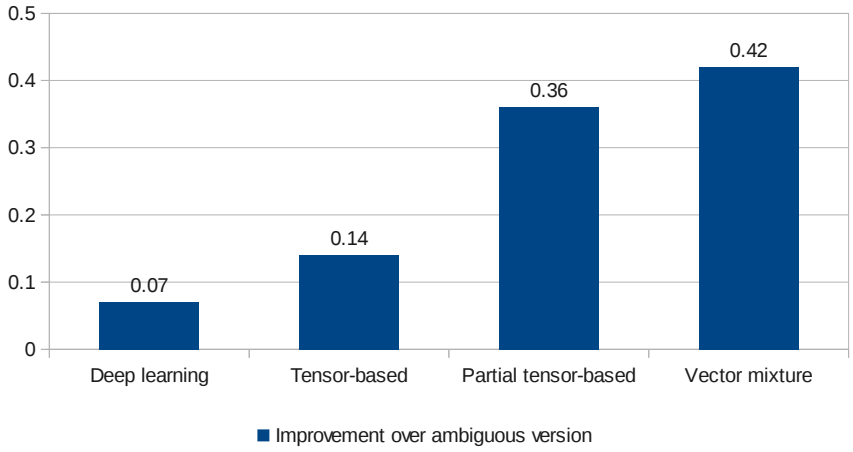


Figure 8.4: General improvement due to prior disambiguation per CDM class for the M&L 2010 dataset.

meaning (as expressed by a function that can directly act on ambiguous vectors efficiently without any prior discrimination of the senses) to a much higher degree than a linear one.

This provides us some intuition for the success of prior disambiguation on tensor-based models: replacing a single linear model with two or more, offers a means of non-linearity that improves the performance of the model. This idea is demonstrated in Fig. 8.5. Imagine that the green curve describes verb-object structures for the verb 'play', such as "play guitar", "play football", and so on; naturally, we expect phrases of the former kind to be closer to base 'music', and of the latter form to base 'sports'. The dashed line in the left-hand diagram represents a single ambiguous tensor for verb 'play' that aims to approximate this non-linear behaviour of the verb phrases. However, this can be done efficiently only by creating a separate linear model for every meaning, as in the right-hand diagram. Of course, the ability of a deep learning model to fit on such a curve would be greater in any case.

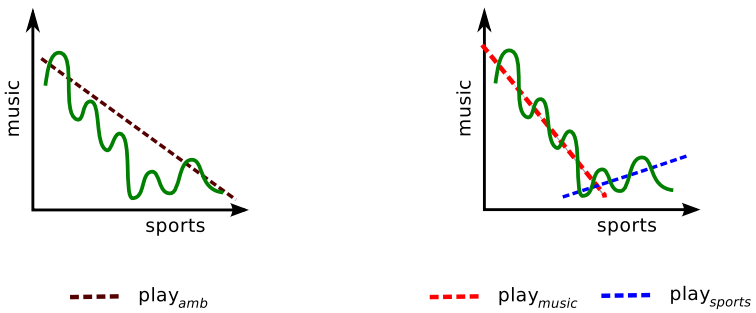


Figure 8.5: Prior disambiguation as a means of providing non-linearity to linear models.

Prior disambiguation offers even greater benefits for partial tensor-based models and vector mixtures, as it is shown in Fig. 8.4. The composition in both of these classes is based (only partially for the former class) on some form of averaging of the vectors of the words of the sentence or phrase. As discussed before, the contribution of the prior disambiguation step in this case is that it prevents the model from using very generic inputs, which will only produce even more generic and non-informative points at the sentence space as results.

Chapter 9

Future Work and Conclusions

In this dissertation I presented various ways to improve and extend the categorical model of Coecke, Sadrzadeh and Clark [25]. The implementation framework based on the Frobenius algebras constitutes a scalable and linguistically motivated base that provides extensive coverage of the language and a test-bed for creative experimentation. Furthermore, although in my exposition I mostly used as a starting point the simple argument summing procedure of [40], the framework in principle is independent of any specific technique of creating the concrete verb tensors, assuming of course that the adopted approach will respect the multi-linear nature of the model. In §4.8, for example, I briefly discussed how one can produce the required multi-linear maps using regression.

The work contributed by this project, in conjunction with relevant research coming from other sources, allow us for the first time to abandon the fixed simple structures on which the categorical model has been evaluated until now and move on towards a large-scale evaluation conducted on sentences of arbitrary complexity. This is one of our first priorities for the future. An appropriate generic task towards this purpose is paraphrase detection, which has been used in the past for evaluating the performance of other classes of CDMs [112], thus it allows the first cross-model direct comparison.

Admittedly, there are a lot of issues that still need to be addressed in the context of tensor-based models. One of the biggest problems for the moment is the proper treatment of logical aspects of language, such as negation, for which there still exists no satisfactory solution. Furthermore, concepts such as quantification seem to be incompatible with vector space semantics: a quantifier operates on a number of individual entities by counting or enumerating them: all men, some women, three cats, at least four days. This fits nicely in the logical view of formal semantics, but it is not clear how it can be applied in a distributional setting. For the next few years, it is

expected that these issues will receive a lot of attention from researchers working on the theoretical aspects of CDMs. As an example, Rypacek and Sadrzadeh [102] made a first step towards modelling quantifiers in the context of the categorical framework.

The quantum formulation of §6.3 provides many opportunities for further work, both at the theory and the practice side. For the latter case, the plan is to attempt a first evaluation of the setting using real-valued state vectors. In order to avoid the inevitable explosion of dimensionality that comes with density matrices, an experiment like that would have to be restricted to compositions between two words, such as the verb-object constructs used in §8.5. The Frobenius framework will be once more a valuable ally towards this purpose; recall from §6.5 that compositions of this kind reduce to the point-wise multiplication between the density matrices of the two words, significantly mitigating space complexity problems.

The new quantum perspective opens many new doors for exciting new research at the theory side as well. One interesting direction for us is to further investigate the role of entanglement in linguistics, a topic that has been briefly discussed in §4.7, but when seen in the context of quantum mechanics takes a new richer meaning (§6.6). Furthermore, as discussed in §6.5, a very interesting topic with both theoretical and practical aspects would be the further investigation of alternative structures that conform to the definition of a Frobenius algebra in **CPM(FHilb)**, and the analysis and evaluation of the results from a linguistic perspective. Last, but not least, note that under certain conditions the density matrix formalism imposes a form of logic that potentially can provide solutions to the aforementioned problems related to logical aspects of language. Further work is needed for making these insights concrete and for evaluating their true potential.

An important general outcome of this thesis which is of great interest for any class of CDMs, is that explicitly dealing with lexical ambiguity improves the compositional process. Especially for multi-linear models, a prior disambiguation step can bring substantial improvements, due to the fact that it provides some means for them to approximate possibly non-linear relationships in the sentence space. Although the effect is smaller, the experimental work suggests that there exist certain benefits even in the case of deep learning models, despite their higher theoretical power. A reasonable future step is to investigate the hypothesis that incorporating the disambiguation step as an additional layer *within* a deep learning compositional architecture can be an even more effective way to improve the quality of the produced vectors. In the simplest case, this can have the form of a *competitive layer*, an idea that has been applied before for creating unambiguous neural vector spaces by Neelakantan et al.

[83], although not in a compositional setting. A more interesting approach is to model the unsupervised part of the architecture as a *self-organized map* (SOM) [57], which in general can provide more fine-grained clusters.

A closing remark

In the recent years, compositional distributional models of meaning have been evolved to a very active area of research, attracting a lot of attention from the NLP community. The chances are that this trend will continue in the future. Admittedly, there is a lot of ground yet to be covered—to the extent that, for many people, the ultimate goal of capturing the meaning of a sentence in a computer's memory might currently seem rather utopian, something of only theoretical interest for the researchers to play with. That would be wrong; this is a technology with great potential that can drastically influence and improve the practice of natural language processing, and my hope for the work presented in this dissertation is that it positively contributes towards this purpose.

Bibliography

[1] Samson Abramsky and Bob Coecke. A categorical semantics of quantum protocols. In *19th Annual IEEE Symposium on Logic in Computer Science*, pages 415–425, 2004.

[2] Kazimierz Ajdukiewicz. Die Syntaktische Konnexität. Studia Philosophica 1: 1-27. In Storrs McCall, editor, *Polish Logic 1920-1939*, pages 207–231. 1935.

[3] Emmon Bach. An extension of classical transformational grammar. In *Problems in Linguistic Metatheory: Proceedings of the 1976 Conference at Michigan State University*, pages 183–224, Lansing, 1976. Michigan State University.

[4] Esma Balkır. Using density matrices in a compositional distributional model of meaning. Master's thesis, University of Oxford, 2014.

[5] Yehoshua Bar-Hillel. A quasi-arithmetical notation for syntactic description. *Language*, pages 47–58, 1953.

[6] M. Baroni and R. Zamparelli. Nouns are Vectors, Adjectives are Matrices. In *Proceedings of Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2010.

[7] Marco Baroni, Raffaela Bernardi, and Roberto Zamparelli. Frege in space: A program of compositional distributional semantics. *Linguistic Issues in Language Technology*, 9, 2014.

[8] Marco Baroni, Georgiana Dinu, and Germán Kruszewski. Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, volume 1, 2014.

[9] Yoshua Bengio, Holger Schwenk, Jean-Sébastien Senécal, Fréderic Morin, and Jean-Luc Gauvain. Neural probabilistic language models. In *Innovations in Machine Learning*, pages 137–186. Springer, 2006.

[10] William Blacoe, Elham Kashefi, and Mirella Lapata. A quantum-theoretic approach to distributional semantics. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 847–857, Atlanta, Georgia, June 2013. Association for Computational Linguistics.

[11] William Blacoe and Mirella Lapata. A comparison of vector-based representations for semantic composition. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 546–556, Jeju Island, Korea, July 2012. Association for Computational Linguistics.

[12] Ron Borowsky and Michael EJ Masson. Semantic ambiguity effects in word identification. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 22(1):63, 1996.

[13] B. Broda and W. Mazur. Evaluation of clustering algorithms for word sense disambiguation. *International Journal of Data Analysis Techniques and Strategies*, 4(3):219–236, 2012.

[14] S. Brody and M. Lapata. Bayesian word sense induction. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 103–111. Association for Computational Linguistics, 2009.

[15] Peter Bruza, Kirsty Kitto, Douglas Nelson, and Cathy McEvoy. Is there something quantum-like about the human mental lexicon? *Journal of Mathematical Psychology*, 53(5):362–377, 2009.

[16] John A Bullinaria and Joseph P Levy. Extracting semantic representations from word co-occurrence statistics: A computational study. *Behavior research methods*, 39(3):510–526, 2007.

[17] T. Caliński and J. Harabasz. A Dendrite Method for Cluster Analysis. *Communications in Statistics-Theory and Methods*, 3(1):1–27, 1974.

[18] A. Carboni and R.F.C. Walters. Cartesian Bicategories I. *Journal of Pure and Applied Algebra*, 49, 1987.

[19] Jianpeng Cheng, Dimitri Kartsaklis, and Edward Grefenstette. Investigating the role of prior disambiguation in deep-learning compositional models of meaning. In *2nd Workshop of Learning Semantics, NIPS 2014*, Montreal, Canada, December 2014.

[20] Stephen Clark. Type-driven syntax and semantics for composing meaning vectors. In Chris Heyden, Mehrnoosh Sadrzadeh, and Edward Grefenstette, editors, *Quantum Physics and Linguistics: A Compositional, Diagrammatic Discourse*. Oxford University Press, 2013.

[21] Stephen Clark and Stephen Pulman. Combining Symbolic and Distributional Models of Meaning. *Proceedings of the AAAI Spring Symposium on Quantum Interaction*, pages 52–55, 2007.

[22] B. Coecke, E. Grefenstette, and M. Sadrzadeh. Lambek vs. Lambek: Functorial vector space semantics and string diagrams for Lambek calculus. *Annals of Pure and Applied Logic*, 2013.

[23] B. Coecke and E. Paquette. Categories for the Practicing Physicist. In B. Coecke, editor, *New Structures for Physics*, pages 167–271. Springer, 2011.

[24] B. Coecke, D. Pavlovic, and J. Vicary. A New Description of Orthogonal Bases. *Mathematical Structures in Computer Science*, 1, 2008.

[25] B. Coecke, M. Sadrzadeh, and S. Clark. Mathematical Foundations for a Compositional Distributional Model of Meaning. Lambek Festschrift. *Linguistic Analysis*, 36:345–384, 2010.

[26] Bob Coecke and Dusko Pavlovic. Quantum measurements without sums. *arXiv preprint quant-ph/0608035*, 2006.

[27] Bob Coecke and Robert W Spekkens. Picturing classical and quantum Bayesian inference. *Synthese*, 186(3):651–696, 2012.

[28] Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. ACM, 2008.

[29] J. Curran. *From Distributional to Semantic Similarity*. PhD thesis, School of Informatics, University of Edinburgh, 2004.

[30] Donald Davidson. The logical form of action sentences. In Nicholas Rescher, editor, *The Logic of Decision and Action*, pages 81–120. Univ. of Pittsburgh Press, 1967.

[31] David R Dowty, Robert Wall, and Stanley Peters. *Introduction to Montague Semantics*, volume 11. Springer, 1981.

[32] K. Erk and S. Padó. A Structured Vector-Space Model for Word Meaning in Context. In *Proceedings of Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 897–906, 2008.

[33] Adriano Ferraresi, Eros Zanchetta, Marco Baroni, and Silvia Bernardini. Introducing and evaluating ukWaC, a very large web-derived corpus of English. In *Proceedings of the 4th Web as Corpus Workshop (WAC-4) Can we beat Google*, pages 47–54, 2008.

[34] J.R. Firth. A synopsis of linguistic theory 1930–1955. In *Studies in Linguistic Analysis*. 1957.

[35] G. Frege. *The Foundations of Arithmetic: A logico-mathematical enquiry into the concept of number*. Northwestern Univ Pr, 1980. (Translation: Austin, J.L.).

[36] G. Frege. Letter to Jourdain. In G. Gabriel, editor, *Philoshopical and Mathematical Correspondence*, pages 78–80. Chicago University Press, 1980.

[37] D. Graff, J. Kong, K. Chen, and K. Maeda. English Gigaword. *Linguistic Data Consortium, Philadelphia*, 2003.

[38] Edward Grefenstette. *Category-Theoretic Quantitative Compositional Distributional Models of Natural Language Semantics*. PhD thesis, June 2013.

[39] Edward Grefenstette, Georgiana Dinu, Yao-Zhong Zhang, Mehrnoosh Sadrzadeh, and Marco Baroni. Multi-step regression learning for compositional distributional semantics. In *Proceedings of the 10th International Conference on Computational Semantics (IWCS 2013)*, 2013.

[40] Edward Grefenstette and Mehrnoosh Sadrzadeh. Experimental support for a categorical compositional distributional model of meaning. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1394–1404, Edinburgh, Scotland, UK., July 2011. Association for Computational Linguistics.

[41] G. Grefenstette. *Explorations in Automatic Thesaurus Discovery*. Springer, 1994.

[42] Z. Harris. *Mathematical Structures of Language*. Wiley, 1968.

[43] Paul Hoffman, Matthew A Lambon Ralph, and Timothy T Rogers. Semantic diversity: a measure of semantic ambiguity based on variability in the contextual usage of words. *Behavior research methods*, 45(3):718–730, 2013.

[44] J. Jiang and D. Conrath. Semantic similarity based on corpus statistics and lexical taxonomy. In *Proceedings on International Conference on Research in Computational Linguistics*, pages 19–33, Taiwan, 1997.

[45] Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. A convolutional neural network for modelling sentences. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 655–665, Baltimore, Maryland, June 2014. Association for Computational Linguistics.

[46] Dimitri Kartsaklis. Compositional operators in distributional semantics. *Springer Science Reviews*, 2(1-2):161–177, 2014.

[47] Dimitri Kartsaklis, Nal Kalchbrenner, and Mehrnoosh Sadrzadeh. Resolving lexical ambiguity in tensor regression models of meaning. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 212–217, Baltimore, Maryland, June 2014. Association for Computational Linguistics.

[48] Dimitri Kartsaklis and Mehrnoosh Sadrzadeh. Prior disambiguation of word tensors for constructing sentence vectors. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1590–1601,

Seattle, Washington, USA, October 2013. Association for Computational Linguistics.

[49] Dimitri Kartsaklis and Mehrnoosh Sadrzadeh. A study of entanglement in a categorical framework of natural language. In Bob Coecke, Ichiro Hasuo, and Prakash Panangaden, editors, Proceedings 11th workshop on *Quantum Physics and Logic,* Kyoto, Japan, 4-6th June 2014, volume 172 of *Electronic Proceedings in Theoretical Computer Science*, pages 249–261. Open Publishing Association, 2014.

[50] Dimitri Kartsaklis and Mehrnoosh Sadrzadeh. A Frobenius model of information structure in categorical compositional distributional semantics. In *Proceedings of the 14th Meeting on the Mathematics of Language*, Chicago, USA, 2015. To appear.

[51] Dimitri Kartsaklis, Mehrnoosh Sadrzadeh, and Stephen Pulman. A unified sentence space for categorical distributional-compositional semantics: Theory and experiments. In *Proceedings of 24th International Conference on Computational Linguistics (COLING 2012): Posters*, pages 549–558, Mumbai, India, December 2012. The COLING 2012 Organizing Committee.

[52] Dimitri Kartsaklis, Mehrnoosh Sadrzadeh, and Stephen Pulman. Separating disambiguation from composition in distributional semantics. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 114–123, Sofia, Bulgaria, August 2013. Association for Computational Linguistics.

[53] Dimitri Kartsaklis, Mehrnoosh Sadrzadeh, Stephen Pulman, and Bob Coecke. Reasoning about meaning in natural language with compact closed categories and Frobenius algebras. In J. Chubb, A. Eskandarian, and V. Harizanov, editors, *Logic and Algebraic Structures in Quantum Computing and Information*, Association for Symbolic Logic Lecture Notes in Logic. Cambridge University Press, 2015. To appear.

[54] G Maxwell Kelly. Many-variable functorial calculus (I). In G.M. Kelly, M. Laplaza, G. Lewis, and S. MacLane, editors, *Coherence in categories*, pages 66–105. Springer, 1972.

[55] W. Kintsch. Predication. *Cognitive Science*, 25(2):173–202, 2001.

[56] J. Kock. Frobenius Algebras and 2D Topological Quantum Field Theories. In *London Mathematical Society Student Texts*. Cambridge University Press, 2003.

[57] Teuvo Kohonen. Self-organized formation of topologically correct feature maps. *Biological cybernetics*, 43(1):59–69, 1982.

[58] S. Lack. Composing PROPs. *Theory and Applications of Categories*, 13:147–163, 2004.

[59] J. Lambek. *From Word to Sentence*. Polimetrica, Milan, 2008.

[60] Joachim Lambek. The mathematics of sentence structure. *American mathematical monthly*, pages 154–170, 1958.

[61] T. Landauer and S. Dumais. A Solution to Plato's Problem: The Latent Semantic Analysis Theory of Acquision, Induction, and Representation of Knowledge. *Psychological Review*, 1997.

[62] T.K. Landauer. Single representation of multiple meanings in latent semantic analysis. In Gorfein D.S., editor, *On the consequences of meaning selection: Perspectives on resolving lexical ambiguity*, pages 217–232. American Psychological Association, Washington, DC, 2001.

[63] Omer Levy, Yoav Goldberg, and Israel Ramat-Gan. Linguistic regularities in sparse and explicit word representations. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning, Baltimore, Maryland, USA, June. Association for Computational Linguistics*, 2014.

[64] D. Lin. Automatic Retrieval and Clustering of Similar Words. In *Proceedings of the 17th international conference on Computational linguistics-Volume 2*, pages 768–774. Association for Computational Linguistics, 1998.

[65] W. Lowe. Towards a Theory of Semantic Space. In *CogSci*, pages 576–581. Citeseer, 2001.

[66] W. Lowe and S. McDonald. The Direct Route: Mediated Priming in Semantic Space. In *Proceedings of the 22nd Annual Conference of the Cognitive Science Society*, pages 675–680, Philadelphia, PA, 2000.

[67] Saunders Mac Lane. *Categories for the Working Mathematician*. Number 5 in Graduate Texts in Mathematics. Springer-Verlag, 1971.

[68] S. Manandhar, I.P. Klapaftis, D. Dligach, and S.S. Pradhan. Semeval-2010 task 14: Word sense induction & disambiguation. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 63–68. Association for Computational Linguistics, 2010.

[69] C.D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[70] James L McClelland, Mark St. John, and Roman Taraban. Sentence comprehension: A parallel distributed processing approach. *Language and cognitive processes*, 4(3-4):SI287–SI335, 1989.

[71] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In *Proceedings of the Workshop at the International Conference on Learning Representations (ICLR)*, 2013.

[72] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119, 2013.

[73] Dmitrijs Milajevs, Dimitri Kartsaklis, Mehrnoosh Sadrzadeh, and Matthew Purver. Evaluating neural word representations in tensor-based compositional settings. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 708–719, Doha, Qatar, October 2014. Association for Computational Linguistics.

[74] G.W. Milligan and M.C. Cooper. An Examination of Procedures for Determining the Number of Clusters in a Data Set. *Psychometrika*, 50(2):159–179, 1985.

[75] J. Mitchell and M. Lapata. Vector-based Models of Semantic Composition. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics*, pages 236–244, 2008.

[76] Jeff Mitchell and Mirella Lapata. Composition in distributional models of semantics. *Cognitive Science*, 34(8):1388–1439, 2010.

[77] Richard Montague. English as a formal language. In *Linguaggi nella Società e nella Tecnica*, pages 189–224. Edizioni di Comunità, Milan, 1970.

[78] Richard Montague. Universal grammar. *Theoria*, 36:373–398, 1970.

[79] Richard Montague. The proper treatment of quantification in ordinary English. In J. Hintikka e.a., editor, *Approaches to Natural Language*, pages 221–242. Reidel, 1973.

[80] Daniel Müllner. fastcluster: Fast Hierarchical Clustering Routines for R and Python. `http://math.stanford.edu/~muellner`, 2012. Version 1.1.6.

[81] Tadasi Nakayama. On Frobeniusean algebras. I. *Annals of Mathematics. Second Series*, 40(3):611–633, 1939.

[82] Tadasi Nakayama. On Frobeniusean algebras. II. *Annals of Mathematics. Second Series*, 42(1):1–21, 1941.

[83] Arvind Neelakantan, Jeevan Shankar, Alexandre Passos, and Andrew McCallum. Efficient non-parametric estimation of multiple embeddings per word in vector space. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, Doha,Quatar, October 2014. Association for Computational Linguistics.

[84] S. Padó and M. Lapata. Dependency-based Construction of Semantic Space Models. *Computational Linguistics*, 33(2):161–199, 2007.

[85] Denis Paperno, Nghia The Pham, and Marco Baroni. A practical and linguistically-motivated approach to compositional distributional semantics. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 90–99, Baltimore, Maryland, June 2014. Association for Computational Linguistics.

[86] T. Pedersen. UMND2: SenseClusters applied to the sense induction task of Senseval-4. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, pages 394–397. Association for Computational Linguistics, 2007.

[87] Francis Jeffry Pelletier. Did Frege believe Frege's principle? *Journal of Logic, Language and information*, 10(1):87–114, 2001.

[88] M.J. Pickering and S. Frisson. Processing ambiguous verbs: Evidence from eye movements. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 27(2):556, 2001.

[89] Robin Piedeleu. Ambiguity in categorical models of meaning. Master's thesis, University of Oxford, 2014.

[90] Robin Piedeleu, Dimitri Kartsaklis, Bob Coecke, and Mehrnoosh Sadrzadeh. Open system categorical quantum semantics in natural language processing. *arXiv preprint arXiv:1502.00831*, 2015.

[91] T.A. Plate. Holographic Reduced Representations: Convolution Algebra for Compositional Distributed Representations. In *Proceedings of the 12th International Joint Conference on Artificial Intelligence*, pages 30–35. Citeseer, 1991.

[92] Martin B Plenio and Shashank Virmani. An introduction to entanglement measures. *Quantum Information & Computation*, 7(1):1–51, 2007.

[93] Tamara Polajnar, Luana Fagarasan, and Stephen Clark. Reducing dimensions of tensors in type-driven distributional semantics. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar, October 2014. Association for Computational Linguistics.

[94] Jordan B Pollack. Recursive distributed representations. *Artificial Intelligence*, 46(1):77–105, 1990.

[95] A. Preller and M. Sadrzadeh. Bell states and negative sentences in the distributed model of meaning. In P. Selinger B. Coecke, P. Panangaden, editor, *Electronic Notes in Theoretical Computer Science, Proceedings of the 6th QPL Workshop on Quantum Physics and Logic*. University of Oxford, 2010.

[96] Anne Preller and Joachim Lambek. Free compact 2-categories. *Mathematical Structures in Computer Science*, 17(02):309–340, 2007.

[97] Stephen Pulman. Distributional Semantic Models. In Chris Heyden, Mehrnoosh Sadrzadeh, and Edward Grefenstette, editors, *Quantum Physics and Linguistics: A Compositional, Diagrammatic Discourse*. Oxford University Press, 2013.

[98] Siva Reddy, Ioannis Klapaftis, Diana McCarthy, and Suresh Manandhar. Dynamic and static prototype vectors for semantic composition. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 705–713, 2011.

[99] Joseph Reisinger and Raymond J Mooney. Multi-prototype vector-space models of word meaning. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 109–117. Association for Computational Linguistics, 2010.

[100] Jennifer M Rodd, M Gareth Gaskell, and William D Marslen-Wilson. Modelling the effects of semantic ambiguity in word recognition. *Cognitive Science*, 28(1):89–104, 2004.

[101] A. Rosenberg and J. Hirschberg. V-Measure: A conditional entropy-based external cluster evaluation measure. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 410–420, 2007.

[102] Ondrej Rypacek and Mehrnoosh Sadrzadeh. A low-level treatment of generalized quantifiers in categorical compositional distributional semantics. In *Second Workshop on Natural Language and Computer Science, A workshop affiliated with the joint meeting of Computer Science Logic (CSL) and Logic in Computer Science (LiCS)*, Vienna, July 2014.

[103] M. Sadrzadeh, S. Clark, and B. Coecke. The Frobenius anatomy of word meanings I: subject and object relative pronouns. *Journal of Logic and Computation*, Advance Access, October 2013.

[104] M. Sadrzadeh, S. Clark, and B. Coecke. The Frobenius anatomy of word meanings II: Possessive relative pronouns. *Journal of Logic and Computation*, June 2014.

[105] R. Sansome, D. Reid, and A. Spooner. *The Oxford Junior Dictionary*. Oxford University Press, 2000.

[106] G. Savova, T. Therneau, and C. Chute. Cluster Stopping Rules for Word Sense Discrimination. In *Proceedings of the workshop on Making Sense of Sense: Bringing Psycholinguistics and Computational Linguistics Together*, pages 9–16, 2006.

[107] H. Schütze. Automatic Word Sense Discrimination. *Computational Linguistics*, 24:97–123, 1998.

[108] Peter Selinger. Dagger compact closed categories and completely positive maps. *Electronic Notes in Theoretical Computer Science*, 170:139–163, 2007.

[109] Peter Selinger. A survey of graphical languages for monoidal categories. In Bob Coecke, editor, *New structures for physics*, pages 289–355. Springer, 2011.

[110] Paul Smolensky. Tensor Product Variable Binding and the Representation of Symbolic Structures in Connectionist Systems. *Artificial Intelligence*, 46:159–216, 1990.

[111] R. Socher, E.H. Huang, J. Pennington, A.Y. Ng, and C.D. Manning. Dynamic Pooling and Unfolding Recursive Autoencoders for Paraphrase Detection. *Advances in Neural Information Processing Systems*, 24, 2011.

[112] R. Socher, B. Huval, C. Manning, and Ng. A. Semantic compositionality through recursive matrix-vector spaces. In *Conference on Empirical Methods in Natural Language Processing 2012*, 2012.

[113] R. Socher, C.D. Manning, and A.Y. Ng. Learning Continuous Phrase Representations and Syntactic Parsing with recursive neural networks. In *Proceedings of the NIPS-2010 Deep Learning and Unsupervised Feature Learning Workshop*, 2010.

[114] M. Steedman. *The Syntactic Process*. MIT Press, 2001.

[115] Mark Steedman. Information structure and the syntax-phonology interface. *Linguistic inquiry*, 31(4):649–689, 2000.

[116] S. Thater, H. Fürstenau, and M. Pinkal. Contextualizing semantic representations using syntactically enriched vector models. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 948–957, Uppsala, Sweden, July 2010. Association for Computational Linguistics.

[117] S. Thater, H. Fürstenau, and M. Pinkal. Word meaning in context: A simple and effective vector model. In *Proceedings of the 5th International Joint Conference of Natural Language Processing*, pages 1134–1143, Chiang Mai, Thailand, November 2011. Asian Federation of Natural Language Processing.

[118] Masashi Tsubaki, Kevin Duh, Masashi Shimbo, and Yuji Matsumoto. Modeling and learning semantic co-compositionality through prototype projections and neural networks. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 130–140, Seattle, Washington, USA, October 2013. Association for Computational Linguistics.

[119] Peter D Turney. Domain and function: A dual-space model of semantic relations and compositions. *Journal of Artificial Intelligence Research*, 44:533–585, 2012.

[120] L. Vendramin, R. Campello, and E.R. Hruschka. On the Comparison of Relative Clustering Validity Criteria. In *Proceedings of the SIAM International Conference on Data Mining, SIAM*, pages 733–744, 2009.

[121] D. Widdows. Orthogonal negation in vector spaces for modelling word-meanings and document retrieval. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pages 136–143. Association for Computational Linguistics, 2003.

[122] L. Wittgenstein. Philosophy. In J. Klagge and A. Nordmann, editors, *Philosophical Occasions 1912-1951*. 1993.

[123] Ludwig Wittgenstein. *Philosophical Investigations*. Blackwell, 1963.

Index