UNIVERSITÄT
DES
SAARLANDES

Dep. of Language Science and Technology
MSc. Language Science and Technology

Compositionality in Distributional Formal Semantics

— A thesis submitted for the degree of Master of Science —

Author:
Luuk Suurmeijer
2581219

Supervisors:
Prof. Dr. Matthew W. Crocker
Dr. Noortje J. Venhuizen

Advisor:
Dr. Harm Brouwer

30th of August 2021

Abstract

Compositionality—the construction of complex meaning from simpler constituents—is a fundamental notion in the study of linguistic meaning. Two prominent approaches to modelling meaning are Formal Semantics and Distributional Semantics. While in Formal Semantics, compositionality takes center stage, there is no straightforward means to represent graded probabilistic phenomena. Distributional Semantics, by contrast, is inherently probabilistic, but does not adequately capture truth-conditional constructions such as quantification. A novel framework for meaning representation, Distributional Formal Semantics (DFS), combines the strengths of both approaches, by offering graded probabilistic meaning representations in the form of vectors that are truth-conditionally grounded. This allows DFS to model logical and probabilistic phenomena, and to remain compositional on the propositional level. While it has been shown that meaning representations in DFS can be incrementally constructed using an RNN, we here aim to provide an explicit, set-theoretic mechanism of compositional meaning construction that derives propositional meaning representations from their subpropositional constituents. By arriving at such a mechanism we scale up the coverage of DFS from the confined, small-scale training data of an RNN to the full compositional complexity of natural language.

Contents

Declaration of authorship

I hereby declare that the thesis submitted is my own unaided work. All direct or indirect sources used are acknowledged as references. I am aware that the thesis in digital form can be examined for the use of unauthorized aid and in order to determine whether the thesis as a whole or parts incorporated in it may be deemed as plagiarism. For the comparison of my work with existing sources I agree that it shall be entered in a database where it shall also remain after examination, to enable comparison with future theses submitted. Further rights of reproduction and usage, however, are not granted here. This paper was not previously presented to another examination board and has not been published.

Luuk Suurmeijer, 30th of August 2021, Saarbrücken

Acknowledgements

First I would like to thank Noortje Venhuizen and Harm Brouwer for their excellent guidance and supervision throughout the writing of this thesis. Secondly I thank Martin van Harmelen for his mathematical comments and criticism, which have elevated the quality of the final product. Lastly I would like to express my gratitude to a number of people that helped me grasp the subject matter through having insightful discussions. Namely Pauline Sander, Pia Weißenhorn and Alina Leippert.

Chapter 1

Introduction

Distributional and formal semantics are two strains of modeling linguistic meaning with different strengths and weaknesses. Formal semantic models (inspired by early 19th-century philosophical work such as Frege, 1892 and Russell, 1940) rely on abstractly defined truth-conditional symbolic systems. This allows them to have great expressive power over sentence meaning and its construction (compositionality). However, meaning in formal semantic models is discrete and it typically struggles to accurately account for graded expectation-based phenomena in human meaning construction, such as probabilistic inference. Distributional (or vector-based) semantics (inspired by the distributional hypothesis Firth, 1957) on the other hand has proven immensely successful in modeling graded semantic memory in the lexical domain, but it is not compositional and struggles to model logical phenomena such as quantification (Baroni et al., 2012).

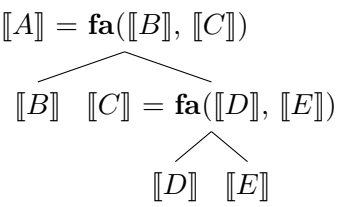
From a cognitive perspective, the qualities of both systems are desirable for modeling human meaning construction and representation. The human computational and representational principles seem to allow for the incremental, compositional construction of sentence meaning and at the same time retain the abilities of expectation-based probabilistic inference. Based on the work by Frank et al. (2003), Venhuizen et al. (2021) propose a unification of the two strains of thought: Distributional Formal Semantics (DFS). Meaning representations in DFS are vectors of truth values of propositions observed from different formal models representing the state of the world at a given moment. A collection of these vector representations (a matrix) then comprises the meaning space. Venhuizen et al. (2021) show that 1. DFS is compositional on the propositional level, 2. can model probabilistic phenomena and 3. DFS representations can be incrementally learned and constructed using a Recurrent Neural Network (RNN). Moreover, DFS representations have been shown to directly give rise to information theoretical principles which have been shown to be related to correlates of human processing difficulty (Venhuizen, Crocker et al., 2019). In order to maximally capitalize on the explanatory power of DFS and to expand its scope to a wider range of linguistic phenomena, full compositionality is desirable for DFS. It would then be possible to see how the notions of surprisal and probabilistic inference as defined in the propositionally instantiated DFS space trickle down into the subpropositional level. This requires the explicit definition of meaning representations on the subpropositional level in DFS (words), as well as a compositional mechanism to combine these representation into more complex (sub)propositional units. The aim of this thesis is exactly to define such a mechanism in the context of DFS. In the following sections, first the notion of compositionality and is implications are discussed. Then, formal and distributional semantics, their attempts at

compositionality and their shortcomings are further explicated. In section 1.2, a more complete introduction to DFS is provided. Lastly, the aim and trajectory of the present thesis is given as well as an introduction to the concept that is key to achieving full compostionality in DFS: the definition of subpropositional units as sets of meaning vectors.

More generally, chapter 1 provides the necessary background for DFS and distributional semantics. Chapter 2 introduces the formal machinery for deriving sentence meaning in DFS. Chapter 3 demonstrates the functionality of the proposed compositional mechanism using a python implementation (https://github.com/LuukSuurmeijer/dfs_semantics_python). Chapter 4 reflects on potential issues and further avenues for research on compositionality in DFS.

1.1 Compositionality in semantics

Compositionality, mainly attributed to Frege (1892) and sometimes referred to as Frege's Principle, is the idea that the meaning of an utterance in a language is a function of the meaning of its constituents and the way in which they are combined. More specifically it dictates that the meaning of a compositional language depends on i) the meaning representation of non-divisible units of the language, ii) the order in which these units are combined (syntax) and iii) the operation by which these units are combined. This allows a languages' interpretable units to be interpreted bottom-up. In order to make the definition of compositionality slightly more tangible, ex. 1 (taken from Potts, 2018) is a visualisation of a compositional derivation of sentence meaning. The terminal nodes of this structure are primitive meaning units. These are composed into the complex meaning units at the non-terminal nodes via the function **fa**(). This operation is key to deriving full meaningful utterance and is referred to as FUNCTION APPLICATION. This example illustrates that theories that strive to be compatible with compositionality must provide explicit answers to the aforementioned points i) and iii), which in ex. 1 are the exact denotations of {A, B, C, D, E} and the instantiation of the operation over those denotations (**fa**()).

(1)


In the context of natural language, the concept of compositionality is nearly uncontroversial (Partee, 2004). It is, given the aim of this thesis, nevertheless useful to discuss the necessity of compositionality for the study of meaning in natural language. First of all, natural language is unbounded and speakers continuously utter and interpret phrases that are totally novel. The assumption of compositionality in any framework of meaning immediately allows accounts for this unboundedness as there is now an (underspecified) mechanism that allows the construction of non-primitive units of meaning of arbitrary complexity. Secondly, natural language is systematic, meaning that the meanings of expressions of different make-up still are predictably related to each other. The sentence "the musician grabbed his instrument" has a related meaning to the sentence "the musician played the tune" and the sentence "the surgeon grabbed his instrument" precisely because of the overlap in meaning of the primitives and structural make-up. In other words, under compositionality the meaning of a sentence is the sum of the meaning of its parts.

It is not exactly clear, however, whether compositionality is a factual claim about language or merely a methodological guideline in the formalization of natural language meaning (Herbelot, 2020). What makes this issue even more complicated is that there is no singular formalization of compositionality for natural language. From the cognitive perspective of language perception, compositionality in the broad sense is simply a prerequisite for meaning construction as the hard constraint on language production and perception is its incrementality. This means that the starting point of how one constructs a representation of an utterance meaning is necessarily a singular primitive of the language to which more and more input is added one by one (continuously). The human language system is hence tasked with incorporating this input into some sort of meaning representation. This incorporation necessarily has a bottom-up component and can be assumed to be, independent of the exact instantiation, composition.

Additionally, we must ask ourselves whether it is really true that the meaning of natural language expressions truly only depends on the meaning of its primitives and the structure of the utterance. It is namely the case compositionality does not immediately have an answer for context-dependent phenomena often categorized as pragmatics, such as intensional predicacy, speaker-orientedness, and reference resolution (Kamp & Partee, 2002). For example, the meaning of a sentence like "John believes that it's raining" seemingly does not depend exclusively on the composition of the meanings of the individual words, but also on a certain belief state that pertains to John (intensionality). This belief state is crucially not encoded in the lexical items (top-down information) and thus compositionality under its strict definition does not allow for this information to contribute to the construction of sentence meaning. Another example is discourse context that is required to interpret the meaning of a sentence. In order to interpret coreference, speakers must have interpreted antecedents in the preceding discourse contexts before being able to interpret their referents in the present input.

Although these phenomena seem to contradict the definition of compositionality, they are not necessarily problematic for many theories of meaning. As mentioned before, compositionality is a theory-dependent concept. The exact instantiation of a compositional mechanism is not uniformly defined and crucially depends on the nature of the representations of the respective model. Hence whether or not a theory of meaning can incorporate top-down information in its derivation of sentence meaning depends on how its primitive representations are defined. Theories that employ such richer semantic representations are often classified under the umbrella term Dynamic Semantics and generalize from traditional formal semantics (section 1.1). An example of a semantic theory that incorporates such context-dependent information is Discourse Representation Theory (DRT) (Kamp & Reyle, 1993), which, as the name implies, incorporates discourse information into its representations. Traditionally, DRT is not explicitely compositional and hence often dubbed a "representational" theory of semantics. However, any theory can be made compositional (and inherit its usefulness and necessity) by defining explicit operations over its representations. Such is the case for DRT with the advent of λ-DRT (Muskens, 1996).

The history of Discourse Representation Theory illustrates that contemporary theories of semantics can have powerful representations that elucidate on context-dependent phenomena, and also may have well-defined compositional rules to abide by compositionality, but that these two considerations are not strictly congruent with each other. The theoretical implication of compositionality is a robust point of departure for theories of semantics, as its requirements force the semanticist to think explicitely about the nature of subpropositional meaning and their

composition. On the other hand, there are meaning-related pragmatic phenomena that seem to defy compositionality. However, richer semantic representations (such as the ones employed in DRT) may provide greater explanatory power even for those phenomena that are traditionally seen as belonging to the field of pragmatics. Given these considerations, in the following section a working definition of compositionality is proposed that will be the guide for the rest of this thesis based on Szabó (2012).

A working definition of compositionality in this thesis

Szabó (2012) argues that the traditional formulation of compositionality is too strict with respect to the arguments for it (i.e. the argument from productivity), and it is not feasible to respect strict compositionality as an objective truth about language given the weaknesses in these arguments. He namely shows that the strict definition is too underspecified to reasonably be supported by its supposed arguments (such as the argument from productivity and systematicity) and to sufficiently deflect counterexamples (such as scope ambiguities and the context-dependence phenomena mentioned earlier). Specifically in the phrase "the meaning of an utterance in a language is determined by the meaning of its constituent parts and the way in which they are combined" Szabó (2012) wonders what "determined by", "meaning of its constituent parts", and "they" could refer to. Szabó (2012) continues to argue that in order to allow semantic theories to scale up their representational currency to include such phenomena, a more specific and weaker form of compositionality must be specified beforehand. However this must come at the expense of working with a weaker statement of the compositionality principle that is costly to assume to be a general fact of language. In order to allow for rich meaning representation that may explain (a subset of) context-dependence phenomena as well as structural phenomena, this thesis takes one of the suggestions for a weaker definition proposed by Szabó (2012) and defines compositionality as follows.

(2) There is a function **fa**() to the meaning of complex utterances from the individual meaning of its constituents and the way in which those meanings are combined.

This thesis explores the notion of compositionality in a novel theory of semantics that seeks to combine the truth-conditional compositional framework of formal semantics (section 1.1) with the representational and empirical power of distributional semantics (section 1.1): Distributional Formal Semantics (Venhuizen et al., 2021). In what follows, this thesis discusses both of these theories. The power of formal semantics, stemming from its definitional rigor and strict compliance with compositionality, is introduced first. Secondly, the power of distributional semantics, stemming from its rich data-driven representations is discussed. We then arrive at Distributional Formal Semantics, the framework that seeks to combine these strengths, and introduce its representations and explanatory power. Lastly, the aim of the present thesis is introduced. Namely to introduce a formal semantic notion of subpropositional meaning and composition to DFS.

Compositionality in formal semantics

Modern day formal semantics is built on concepts introduced by philosophy of language throughout history (for example Frege, 1892 and Plato). One of the well-known formalizations of natural language semantics in a mathematical predictive model, by utilizing properties from logic and

model theory, is Montagovian semantics (Montague, 1973). Montagovian formal semantics tried to explicitly address the mechanisms required for reconstructing the meaning of a sentence given a syntactic structure. Specifically, natural language utterances are formalized as first-order logic sentences. The interpretation of these first-order logical forms comes down to asserting their truth with respect to a certain model $\mathbf{M}$ and an interpretation function $\mathbf{V}$. The model represents a certain state of affairs and the interpretation function relates the linguistic units (individuals, predicates) to the entities and relations represented by the model. I give an example of such a model and interpretation function in fig. 1.1 and eq. (1.1).

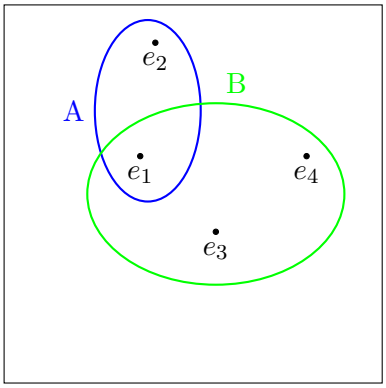


Figure 1.1: An example visualization of a model in formal semantics

$$\begin{aligned} \mathbf{V_M}(Chet) &= e_1 & \mathbf{V_M}(Miles) &= e_2 \\ \mathbf{V_M}(Sarah) &= e_3 & \mathbf{V_M}(Ella) &= e_4 \\ \mathbf{V_M}(sing) &= \{e_1, e_3, e_4\} & \mathbf{V_M}(playtrumpet) &= \{e_1, e_2\} \end{aligned} \tag{1.1}$$

Using a set of symbolic rules, formalized natural language utterances can be interpreted with respect to a model and interpretation function. These symbolic rules are explicated in work like Montague, 1973. For example, a one-place-predicate $[\![R(t_1)]\!]^{M,g}$ is true iff $[\![t_1]\!]^{M,g} \in \mathbf{V_M}(R)$. A conjunction between two utterances $[\![\phi \wedge \psi]\!]^{M,g}$ is true iff $[\![\phi]\!]^{M,g} = 1$ and $[\![\psi]\!]^{M,g} = 1$. In this way, we can see that a sentence like $playtrumpet(Chet) \wedge sing(Chet)$ is true given the model in fig. 1.1, but the sentence $playtrumpet(Miles) \wedge sing(Miles)$ is false. The other logical connectives as well as the first-order quantifiers are covered in this fashion using symbolic interpretation rules. This allows formal semantics to interpret a broad variety of linguistic phenomena and explicitly denote their semantics.

With the interpretation and assertion of complete logical sentences in formal semantics in place, we can now take a look at the exact compositional mechanism employed to construct these complete logical units. Formal semantics employs typed λ-calculus to achieve this. Firstly, we must observe that first-order logic is not expressive enough to capture the full space of natural language utterances. Think for example of second-order predicacy (predicates that take predicates as arguments, "to be ignorant is to be blessed" for example). The aim of type-theory is to assign the elements of our semantic model to a certain amount of mathematical objects thereby restricting the combinatorial space. Concretely, all lexical items are assigned either e

(entities), t (truth-values, sentences) or a complex type consisting of those two elements. Complex types are taken to represent functions, such that an expression of type et denotes a function from entities to truth values. Type theory allows for a systematic, simple but expressive formalism for capturing a whole range of new phenomena (see Winter (2016) for a helpful introduction). More importantly however, the functional approach to complex expressions gives formal semantics an explicit index of compositionality. Individual lexical items may now denote functions of arbitrary order, and the types of complex meaning units may be inferenced from their constituent parts. A helpful mathematical tool for representing these possibly very complex functions is λ-calculus. A full description of λ-calculus is beyond the scope of the present thesis, mainly because λ-calculus inherently is largely a notational device for representing functions (the main type of denotation in formal semantics). Typed λ-calculus essentially allows lexical items and partial functions and their types to be explicitly represented and have an explicit operation for combining linguistic expressions with each other ($\beta - reduction$). For the sake of completeness, an example of the derivation of a full proposition from typed λ-expressions is given in fig. 1.2 (slightly simplified). The expression at the root of the tree in fig. 1.2 is a first-order logic sentence which can be evaluated with respect to a model like fig. 1.1 according to the interpretation rules discussed before (which would result in a truth value of 1 in this case).

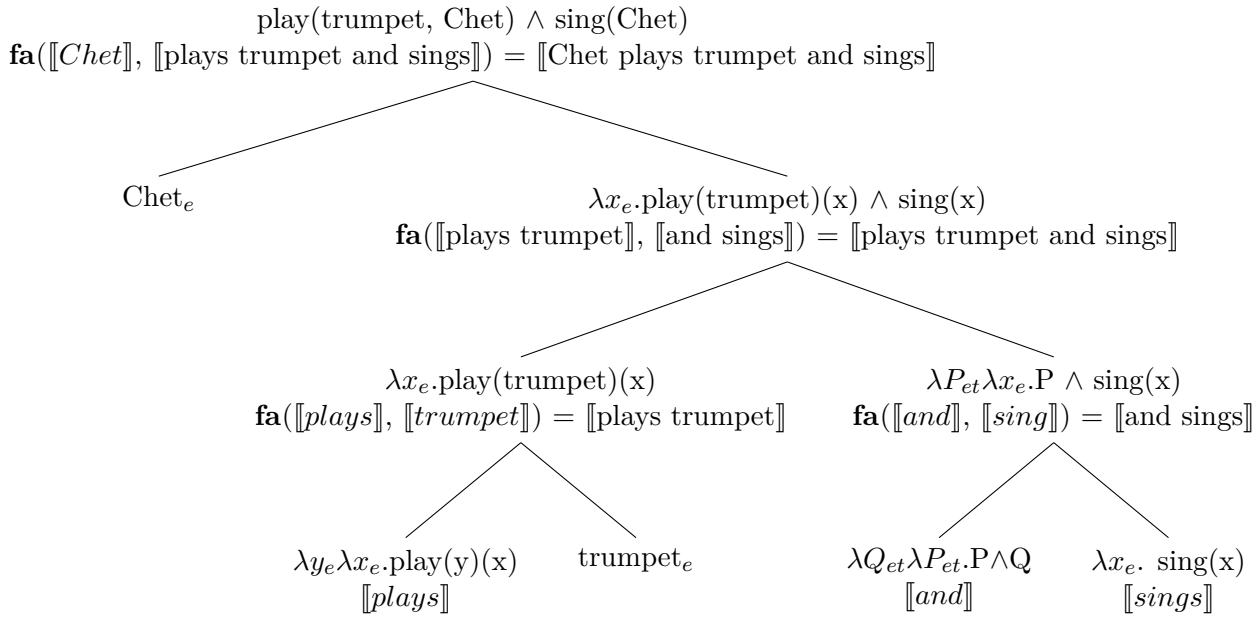


Figure 1.2: An example derivation of a first order logic sentence in formal semantics.

What allows and facilitates compositionality in formal semantics is typed λ-calculus, but the final interpretation of linguistic items is still fully dependent on the formalization of its lexical representations which for formal semantics is first-order logic and model theory. The coverage of the semantic model and its answer to context-dependent phenomena thus still is largely restricted by the nature of its representations. The nature of the representations in first order logic and model theory is binary and deterministic, and hence does not account for graded expectation-based phenomena such as probabilistic inference, fuzzy adverbs (few, many), and tendencies without explicit modification of the representations. Capturing this type of phenomena requires the explicit modification of the mathematical basis of the representations,

which has been attempted in the form of probabilistic model theory (Emerson & Copestake, 2017) and stochastic λ-calculus (Goodman & Lassiter, 2015) among others.

Compositionality in Distributional Semantics

Distributional semantics, sometimes referred to as Vector Space Modelling, is an empirically driven approach to modelling meaning, in which lexical meanings are encoded as vectors. Turney and Pantel (2010) review the literature and technical basis of Vector Space Models. The vector-representations are extracted from corpora based on co-occurrence counts of the corpus' vocabulary items. This results in a description of lexical items that consists solely of information about its distributional use in language. The core assumption of DS is the distributional hypothesis famously attributed to Firth, 1957: "one might know the meaning of a word by the company it keeps". In other words, word meanings that occur in similar contexts tend to have similar meanings. Distributional semantics assumes that the meaning of lexical items, lexical categories may be inferred from the contexts in which the item may appear in and its similarity to other lexical items (see Turney and Pantel, 2010).

Distributional semantics has a number of properties that are appealing for modelling meaning. First of all it is empirically driven, meaning that lexical meanings can be extracted and learned from data automatically. This allows distributional semantics to make large-scale predictions and set-up ambitious experiments regarding the exact nature of its representations. Second of all, because words are represented as vectors, distributional semantics essentially embeds its lexical items into an n-dimensional space. The mathematical nature of these representations allows the use of linear algebraic properties that yield very precise metrics of concepts like semantic similarity (Turney & Pantel, 2010). Baroni et al. (2014) mention that distributional vectors provide a very accurate analogue to human similarity judgements, and that there is an increasing body of evidence that neural activation patterns in language comprehension are akin to distributional representations such as the ones employed in DS (Spivey, 2006).

These properties have made distributional semantics into more than just a model that is useful for the study of natural language meaning. Lexical items represented as distributional word embeddings are now commonplace in other NLP tasks. Using word embeddings as the input to Neural Network architectures for NLP tasks is empirically shown to boost performance by quite a large margin, indicating that distributional representations of lexical items quite accurately capture the necessary linguistic information required for tasks like POS-tagging and question answering (Turney and Pantel, 2010, Bakarov, 2018, Baroni et al., 2014).

On the flip side of the immense success of DS in the lexical domain, it is less clear what exactly the DS framework predicts in functional linguistic items (prepositions, grammatical morphemes) and structural phenomena. Traditional distributional semantics has no explicit operation by which simplex vector representations can be combined into larger complex meaning units. An often proposed function application mechanism consists of certain vector mixture operations (Baroni et al., 2014). The most obvious ways in which to compose vectors is by adding or multiplying them element-wise. Under this view, the composition of the vectors representing the words **jazz** and **musician** would be as in ex. 3 under the additive model and as ex. 4 under the multiplicative model. The appeal of the multiplicative model is that it intersects features of the respective vectors with each other. If both **jazz** and **musician** have a high value for the ith entry, the ith entry of **jazz** $\odot$ **musician** will have a much higher score for the ith entry

(compared to the additive model). Additionally, if one of the features in either component is 0, the resulting entry is by definition also zero irrespective of the other component.

(3) $\mathbf{fa}(\mathbf{jazz}, \mathbf{musician}) = \mathbf{jazz} + \mathbf{musician}$

(4) $\mathbf{fa}(\mathbf{jazz}, \mathbf{musician}) = \mathbf{jazz} \odot \mathbf{musician}$

However, there are number of issues with this compositional mechanism.[1] Firstly, the vector mixture operations are mathematically symmetric. This means that $\mathbf{jazz} \odot \mathbf{musician} = \mathbf{musician} \odot \mathbf{jazz}$. This is contrary to our intuitions about language, where in complex noun phrases the meaning of the noun is modified by the meaning of the adjective and not vice versa. Secondly, words with grammatical function are not easily captured by distributional representations. The features of the vector for the word **musician**, because it is a content word, are tied to specific contexts and topics. To the contrary, consider the vector representing a quantifier like *some* or *all*. Since the function of these words is grammatical, their distribution will not be context-specific and may be associated with any noun thus resulting in a highly generic and non-informative representation. In Baroni et al. (2012), the question of whether DS could capture semantic phenomena like quantification and entailment was investigated using a classification approach. Results indicated that vector representations are able to encapture some lexical entailment relations (for example: *big dog* $\models$ *dog*), but struggled to do so in the case of less explicit lexical entailments involving quantification (*many dogs* $\models$ *some dogs*). Thirdly, distributional representations are not grounded whereas formal semantic representations are grounded in a rudimentary way. It is not obvious that the meaning of a word with an inherent and somewhat fixed spatio-temporal extension (like the english preposition "*on*") can be accurately represented by its distribution in natural language. Distributional representations of objects trained on linguistic form alone cannot capture sensory-motor information about the shape, color, smell etc. of said object beyond some of the sensory terms that are encoded in the language's lexicon.[2]. Lastly, the possibility of recursion in nautral language is problematic for these vector mixture models. Linguistic embeddings of higher depth will tend to either explode the numerical values of each feature, or almost nullify them if there happen to be features represented by floating point numbers < 1 (under the multiplicative model).

Formal semantics sidesteps these problems because of its logical and definitional rigor. Representations are grounded, because their truth is asserted with respect to some model. The meaning of words like *on* is allowed to depend on a high level description of the physical state-of-affairs of the world, which might be different at each point in time or for each individual. Additionally, Formal Semantics treats words like the quantifier *some* as a separate type from content words. Quantifiers are then functions operating on sets of content words. Therefore there is no need to estimate the distribution of the words like *some*, because their function is defined a priori. The logical question to ask is how to reconcile the representational power of Distributional Semantics with the compositional and logical strength of Formal Semantics. The following section discusses such a reconciliation in more detail, namely Distributional Formal Semantics.

[1] See Baroni et al. (2014) for an overview of different accounts of compositionality for distributional semantics.

[2] See also Bender and Koller (2020) for a discussion of whether computational methods trained on linguistic form are able to generalize to non-linguistic meaning.

1.2 Distributional Formal Semantics

The strengths and weaknesses of formal semantics and distributional semantics seem almost complementary. Formal semantic representation and composition is logically sound and grounded, but does not explicitly account for the gradedness of language. Distributional semantics does account for this gradedness and is neurally plausible, but seems to lack the logical mechanisms for dealing with structural and grammatical phenomena. Distributional Formal Semantics combines the strengths of both models while providing an answer to their weaknesses. DFS employs distributional yet truth conditional representations of propositions, which are embedded in a meaning space. I now illustrate the representations defined and employed in DFS in more detail.

A given meaning space $\mathcal{S}$ in DFS is defined in terms of only a handful of ingredients. A set of models $\mathcal{M}$ (where every model $M_i \in \mathcal{M} = \langle U_{M_i}, V_{M_i} \rangle$) a set of propositions $\mathcal{P}$ and some assignment of truth values to each $p_i \in \mathcal{P}$ for every $M_i \in \mathcal{M}$. The meaning space S can then be represented as a matrix A of dimensionality $|\mathcal{M}| \times |\mathcal{P}|$ where each entry A_{ij} is the truth value that $\mathcal{M}_i$ assigns to $\mathcal{P}_j$. I give an example visualisation of a DFS meaning space in fig. 1.3. A DFS meaning space can have a number of theoretical interpretations depending on one's goal. It can be seen as a set of possible worlds (in the tradition of possible-world semantics), a representation of model-theoretical semantics through time, or the belief state (or world-knowledge system) of an individual.

$$
\begin{array}{c} \\ M_1 \\ M_2 \\ M_3 \\ \cdots \\ M_m \end{array}
\begin{array}{c}
\begin{array}{ccccc} p_1 & p_2 & p_3 & \cdots & p_n \end{array} \\
\begin{pmatrix}
1 & 0 & 0 & \cdots & 0 \\
1 & 1 & 1 & \cdots & 1 \\
0 & 1 & 1 & \cdots & 1 \\
\cdots & \cdots & \cdots & \cdots & \cdots \\
0 & 1 & 0 & \cdot & 0
\end{pmatrix}
\end{array}
$$

Figure 1.3: An example meaning space in DFS in matrix form.

It follows from this definition that the representation of unique models and propositions is given by the row and column vectors of the meaning space respectively. This allows DFS to inherit powerful mathematical properties from linear algebra and, with some additional definitions, probability theory. More importantly, these vector representations are key in defining propositional compositionality in DFS. The formal definition of a propositional meaning vector as defined in Venhuizen et al. (2021) is given in eq. (1.2).

$$[\![p]\!]^{\mathcal{S}} = \vec{v}(p) \text{ st: for all } i \in |\mathcal{M}_{\mathcal{P}}| \ \vec{v}_i(p) = 1 \text{ iff } \mathcal{M}_i \models p \tag{1.2}$$

Meaning spaces inherently carry the structure of the world they represent in terms of the co-occurence of propositions in certain models. Say, for example, one would like to model a world in which $\mathcal{P}_1$ = *possess(MasterSword, Link)* and $\mathcal{P}_2$ = *worthy(Link)*, then anyone with knowledge of the Zelda franchise will recognize that Link can only possess the Master Sword if Link is worthy. The meaning space associated with this knowledge must reflect this condition. Formally speaking, it must hold that $\mathcal{P}_1 \models \mathcal{P}_2$ in all models $\in \mathcal{M}_{\mathcal{P}}$ in the meaning space. Venhuizen et al. (2021) propose a sampling-algorithm that can generate a consistent DFS meaning space from a high-level specification of a world and the desired number of models.[3] This high-level specification

[3]For a full description and discussion of the sampling algorithm I refer the reader to Venhuizen et al. (2021).

contains probabilistic and hard constraints (like *possess(MasterSword, Link)* $\models$ *worthy(Link)* in the previous example) that the output space must preserve in terms of co-occurrences of propositions. Hence the meaning of a proposition is defined as its truth-conditional co-occurence with other propositions across models in the meaning space (akin to distributional representation), but the truth of the proposition in each model is asserted based on traditional model theory (akin to Formal Semantics).

I have illustrated how DFS can model worlds in terms of a (thus far) finite set of propositions and models, but DFS can represent the meaning of propositions of arbitrary complexity as well. Venhuizen et al. (2021) ascertain this by defining the notions of negation and conjunction over meaning vectors, thus providing functional completeness and the ability to express any proposition that is some composition of the propositions in $\mathcal{P}$. Negation of a meaning vector is defined as the complement of that vector. Conjunction between two meaning vectors is defined as the element-wise conjunction of two vectors. The definitions are given in eq. (1.3) and eq. (1.4) respectively.[4] Meaning spaces in DFS can thus represent complex world knowledge structures, dependencies and constraints in the form of meaning vectors.

$$[\![\neg p]\!]^{\mathcal{S}} = \vec{v}(\neg p) \text{ st: for all } i \in |\mathcal{M}_{\mathcal{P}}|\ \vec{v}_i(\neg p) = 1 \text{ iff } M_i \not\models p \tag{1.3}$$

$$[\![p \wedge q]\!]^{\mathcal{S}} = \vec{v}(p \wedge q) \text{ st: for all } i \in |\mathcal{M}_{\mathcal{P}}|\ \vec{v}_i(p \wedge q) = 1 \text{ iff } M_i \models p \text{ and } M_i \models q \tag{1.4}$$

Venhuizen et al. (2021) furthermore show that probabilistic inference can also be formalized from meaning vectors and meaning spaces. The probability of a proposition $p \in \mathcal{P}$ can be formalized as the number of models that entail p over the total number of models in the meaning space (eq. (1.5)).

$$P(p) = \frac{|\{\mathcal{M}_i \in \mathcal{M}_{\mathcal{P}} \mid \mathcal{M}_i \models p\}|}{|\mathcal{M}_{\mathcal{P}}|} \tag{1.5}$$

The joint probability of two propositions $p, q \in \mathcal{P}$ can be calculated by calculating the probability of their conjunction as in eq. (1.6) (and similarly for any arbitrarily complex proposition). In this case meaning the fraction of models that entail both p and q.

$$P(p \wedge q) = \frac{|\{\mathcal{M}_i \in \mathcal{M} \mid \mathcal{M}_i \models p \text{ and } \mathcal{M}_i \models q\}|}{|\mathcal{M}_{\mathcal{P}}|} \tag{1.6}$$

From these definitions we can infer that the conditional probability of p given q is computed by the probability of their conjunction over the probability of q.

$$P(p|q) = \frac{P(p \wedge q)}{P(q)} \tag{1.7}$$

Meaning spaces are structured in terms of probabilistic constraints and entailment constraints. For example, playing the piano means the player is probably inside (where piano's typically reside, but not always), but it also means the player is definitely producing a sound. Playing the piano has a probabilistic relationship to being inside and an entailing relationship to producing a sound. Both types of co-occurence can be modelled with probabilities, as the entailing dependency ($p \models q$) will have a conditional probability of 1 (and a negative entailing dependency a probability

[4] Note that p and q are taken to represent some $\mathcal{P}_i$ and $\mathcal{P}_j \in \mathcal{P}$. To adhere to the traditional conventions of propositional logic these notations are henceforth considered equivalent.

of 0). Venhuizen et al. (2021) formalize both types of dependencies in a single metric dubbed the *inference score* eq. (1.8). The inference score between p and q yields a number in the range $[-1, 1]$ that indexes to what extent p can be inferred from q, where $inf(p, q) = -1$ means $q \models \neg p$ and $inf(p, q) = 1$ means $q \models p$.

$$inf(p,q) = \begin{cases} \frac{P(p|q)-P(p)}{1-P(p)} & P(p|q) > P(p) \\ \frac{P(p|q)-P(p)}{P(p)} & otherwise \end{cases} \tag{1.8}$$

Venhuizen et al. (2021) show that these building blocks allow DFS to model a wide range of semantic phenomena, such as presupposition, probabilistic inference, anaphoricity and quantification.

1.3 Subpropositional meaning in DFS

So far, only probabilism and compositionality on the propositional level have been introduced in DFS. In order to analyze and make predictions about the meaning of individual words, similar to how Formal Semantics do so in fig. 1.2 for example, an instantiation of subpropositional meaning with respect to DFS meaning spaces is desirable.

Meaning vectors of propositions in a DFS meaning space consist only of the binary integers 0 and 1, reflecting their truth value in a particular model. However the meaning space itself is by definition continuous, implying that any real valued vector of size $|\mathcal{M}|$ represents some sort of meaning in the space. Because these real valued vectors lie in between the propositional meaning vectors, we can take these intermediately located vectors as representing meaning on the subpropositional level.

Venhuizen et al. (2021) show that these meanings can be indexed using a Recurrent Neural Network (Elman, 1990). A multilayered RNN is trained to map between localist word representations of the lexical items in the propositions of the space and DFS meaning vectors from a sampled meaning space of 10000 models (reduced to 150 for modelling purposes). A schematic of the network architecture is given in fig. 1.4. The network is fed one of these localist word representations at each time step. The RNN outputs an estimate of the intended word meaning after every word. The estimate after the model has seen only one word of the intended utterance will not directly map onto a complete meaning vector, as the model cannot reasonably infer whether for instance the utterance "John orders beer" or the utterance "John orders cola" is meant after only processing the word "John" (bar the probabilistic constraints on John's drink preferences contained in the meaning space). This means that as the model is exposed to more input contained in the utterance, the estimate for the intended meaning will incrementally move closer to the final (binary) utterance meaning. Figure 1.5 gives a visual representation of this trajectory. Formally, a sequence of words $w_1 \dots w_n$ define a trajectory through the meaning space $\vec{v}_1 \dots \vec{v}_n$ where each $\vec{v}_i$ represents a subpropositional meaning of the words w_1 up to w_i.

The non-final estimations of utterance meaning by the model yield real valued vectors in the meaning space. The estimate for a given word lies somewhat in the middle of its possible continuation. This is because the network estimate for a particular word in isolation are partial towards propositions involving this word, but relatively impartial between these relevant propositions. However, the estimate is not completely impartial as the network output is sensitive to the frequency of certain constructions the network has seen at training time. Venhuizen et al.

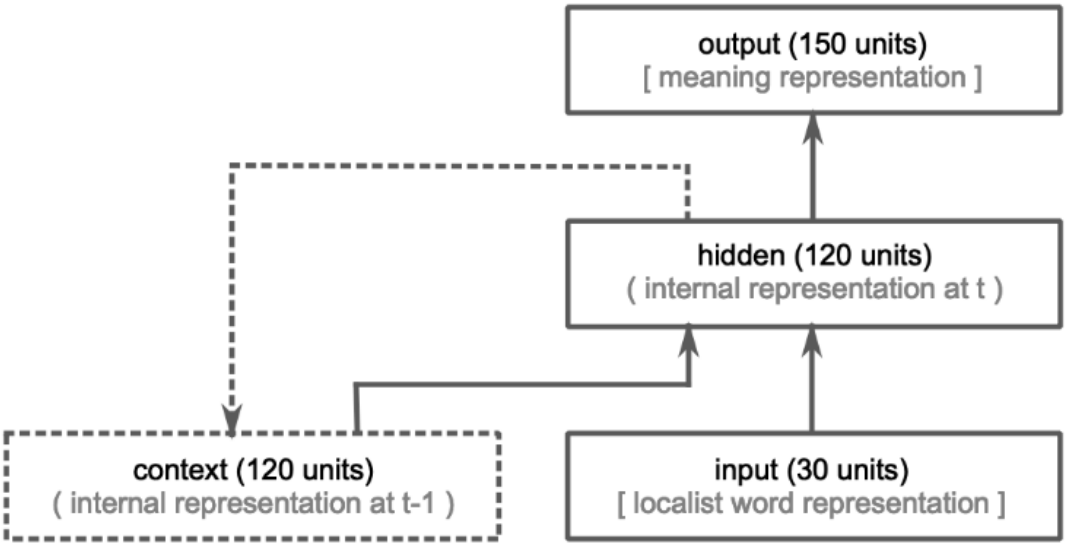


Figure 1.4: Schematic of the neural network architecture taken from Venhuizen et al. (2021). Full arrows imply matrix of learnable parameters. The dotted arrow indicates a copy operation.

(2018) show that the structure of DFS space and frequency modulation in the RNN training data can effectively model world knowledge versus linguistic experience. Ultimately, these neural network estimates are taken to be the subpropositional meaning of the input utterance w up to time t in a particular DFS space.

Venhuizen et al. (2021) furthermore show that the distance between each network estimate is proportional to the point-wise entropy or surprisal of the next incoming word given the history of the utterance so far. Here (conditional) surprisal (eq. (1.9)) of a word/utterance is defined as the negative logarithm of its probability in the distribution of possible words/utterances (given the context). The less expected a word is, the higher its surprisal (or information content) is. In DFS surprisal can be interpreted as the expectedness of a transition from one point in space to another. Cognitively, information theoretical surprisal has proven to be an excellent index of processing difficulty for empirical data like reading times (Brouwer et al., 2021).

$$suprisal(x) = -logP(x|h) \tag{1.9}$$

In conclusion, the probabilistic, compositional and incremental nature of the DFS components (including the neural network generating the subpropositional meaning vectors) are able to capture import aspects of natural language meaning. For example, Venhuizen et al. (2021) show that notions of presupposition, quantification, and anaphoricity emerge from the model behavior. However, there are practical drawbacks to employing a neural network to find the subpropositional representations for further analysis. In the following section these drawbacks are explicated and the main aim of this thesis is introduced, namely to introduce a set-theoretic and transparent notion of subpropositional meaning and composition.

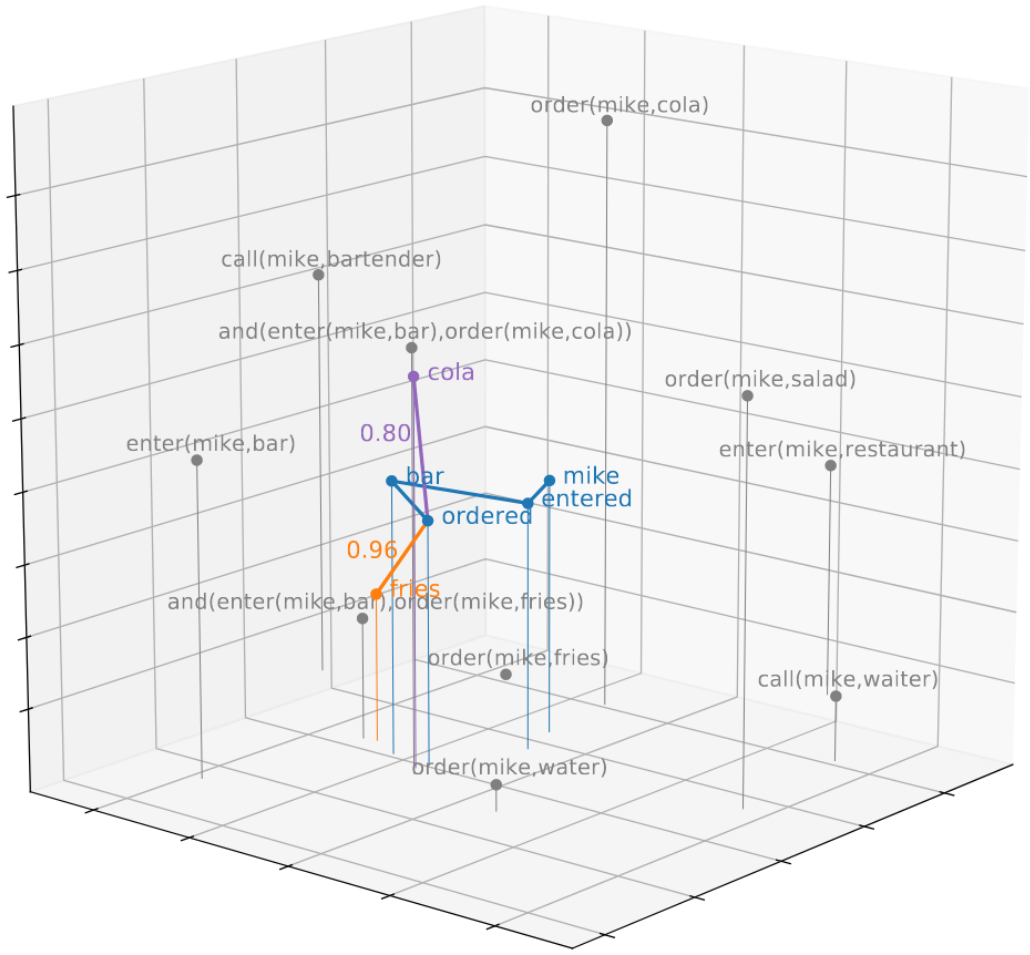


Figure 1.5: Trajectory of RNN outputs through the dimensionality-reduced meaning space taken from Venhuizen et al. (2021). The coloured nodes are the network outputs after seeing a particular word. The coloured lines (and numbers) indicate the distance between respective points in the meaning space.

1.4 Towards explicit compositionality in DFS

In principle, there are no huge theoretical reservations for using neural networks as compositional mechanisms. A neural network adheres largely to the definition of compositionality in ex. 2. Given some sort of distributed vector representation of lexical items, the composition of multiple items can be given by their multiplication with a matrix of learned parameters (Potts, 2018). Because this weight matrix is estimated holistically on training data (i.e. every input previously seen and used during training has an effect on the weight matrix which percolates to the output estimate of the current input), one might say that this violates the spirit of compositionality. However, the definition in ex. 2 poses no intrinsic restriction on the richness of the mechanism nor the representation. In fact, we have already seen some context-dependent "counterexamples" that seem to require such holistically informed mechanisms to be captured.

Furthermore, the benefits of a deep learning approach are considerable for the field of semantics on the whole. The performance of a neural network is directly quantifiable on preset evaluation criteria, specifically its ability to generalize to unseen data (something that should be of particular interest to researchers interested in meaning construction). Furthermore, deep

learning approaches to semantic modelling open the door to larger-scale empiricism in the field of semantics (see Potts (2018) for additional information of the role of neural networks in semantics).

However, as already mentioned, using an RNN for the task of composing complex meaning vectors from simplex parts has a particular drawback. To train a neural model of sufficient accuracy on a semantic task, a certain degree of complexity is required. The RNN employed in Venhuizen et al. (2021) alone, which is a comparatively simple model, already has roughly 36.000 learnable parameters (the cardinality of the weight matrices summed). Complexity may also be increased by adding non-linearities to the model, altering the training procedure or introducing more hyperparameters. To a certain extent this complexity is warranted, sometimes because the mapping task of the network is not trivial and furthermore because the neural machinery involved in meaning construction in humans is highly intricate as well. The drawback of this complexity is that the behavior and solution of the network become increasingly hard to trace and explain as complexity increases. For a compositional theory of semantics however, it is exactly this trace-ability and explain-ability of the mechanism that is vital to understanding how meaning is constructed.

This thesis aims to introduce a transparent semantically-informed mechanism and representational format for composing meanings in DFS, that ideally mimics the outputs of the RNN. Although subpropositional word/phrase representations can be implicitly constructed by the RNN, the approach is not unequivocally desirable. Namely because the mapping that the RNN learns is not transparent and it is not trivial to inspect the network and reverse engineer the mechanism that the RNN employs to produce the correct outputs (although DFS representations are structured in a such a way that introspection becomes relatively clear). This means that it is not always clear what information the RNN uses to construct meaning vectors and whether they are semantically informed. Secondly, the lack of explicit subpropositional composition limits the scope of the phenomena that can be studied using DFS. Such a system would allow DFS to increase the scale of its predictions by a large margin. It would also make DFS more directly comparable with existing/competing formal semantic theories. Lastly, being able to show that the RNN mechanism of constructing meaning has an analogue formulated in more traditional semantic vocabulary would be an important argument for solidifying the validity of using deep learning in pushing forward the field of semantics.

The central goal of this thesis is to define and implement such a mechanism for DFS. We already know the formalization of propositions in DFS: they are vectors of truth values (meaning vectors). Subpropositional units can then be defined as sets of meaning vectors relevant to a given subpropositional unit. We can then define operations, such as function application, over these sets that will allow the incremental and compositional construction of complex DFS representations, thus providing the full compositionality that would allow DFS to increase its explanatory power. In essence, the thesis is aimed at re-deriving first-order-logic using operations over sets of vectors. It is then crucial that the mechanisms and representations accurately cover n-ary predicates, logical operators and quantification as a baseline. When this mechanism is in place, we can then inspect how the existing propositional definitions in DFS, for example for surprisal and probabilistic inference, percolate to the subpropositional level. Secondly, we are able to analyse other natural language phenomena such as different adjective classes (“former” versus “Dutch” for example). Lastly, we can compare the set-theoretic predictions to the Neural Network estimates and meaningfully comment on the differences and symbiosis between intensional and

deep learning approaches to modelling compositional semantic construction.

Chapter 2

Set-theoretic Compositonality

As the neural network estimates show, meanings of individual words tend to lie in between their potential continuations. It then makes sense that every proposition in the meaning space involving a word w plays a role in the denotation of w itself. To capture this intuition, a starting point of formalizing subpropositional meaning in DFS is defining the denotation of a word w as the set of propositions that pertain to w (i.e. w occurs in in a proposition in any logical capacity). The definition of the word 'Chet' as in fig. 1.5 is then preferably the set of propositions pertaining to 'Chet'.

(5) $[\![Chet]\!] = \{has(microphone, Chet), play(trumpet, Chet), sing(Chet) \ldots\}$

The definition is already hinted at in Venhuizen et al. (2021) and here given in eq. (2.1). The meaning of a word $\mathbf{w}$ in the meaning space is a set of all propositions p in the set of atomic propositions (A), such that $\mathbf{w}$ pertains to p. Pertainment here is represented as a function w that returns true if a word occurs in a proposition (in any logical capacity).

$$[\![\mathbf{w}]\!]^{\mathcal{S}} = \{\vec{v}(p) \mid w(\mathbf{w}, p), \text{ for all } p \in A\} \tag{2.1}$$

Taking this definition as a starting point, a number of concrete questions arise as central to the thesis. Namely

(i) what are the logical implications of defining word-level meanings as sets of propositions? Does this apply to *all* words in DFS?

(ii) what does it mean to combine sets of propositions into larger meaningful constituents? i.e. What exactly is **fa**() in DFS and what does it do?

(iii) how do the fundamental principles and concepts in DFS, now defined at the propositional level, percolate to the subpropositional level?

(iv) how can we map sets of propositions back into real-valued vectors (like the RNN outputs)?

In what follows a preliminary answer to each is formulated.

2.1 Logical implications of defining words as sets

Using the definition in eq. (2.1), we arrive at a very clear objective for a compositionality in DFS. Since words are sets of propositions, the combinatorial mechanism is tasked with finding the correct

(singleton) subset of propositions from the underlying set-representations at each compositional step. In other words, function application over these sets now involves subsecting and combining the correct vectors in the relevant sets in order to narrow down the possible continuations. We can do this iteratively given some sort of syntactic structure until the composition yields a singleton set containing a meaning vector representing the propositional meaning. An example of the most simple case would be to derive the vector describing **sing(Chet)** as some operation **fa**() over the set of vectors pertaining to **sing** and the set of vectors pertaining to **Chet**.

$$[\![\text{sing(Chet)}]\!] = \mathbf{fa}([\![\text{sing}]\!], [\![\text{Chet}]\!]) \tag{2.2}$$

One of the traditional set-theoretic operations (like intersection, union and complement) might seem intuitively like a good fit for **fa**(). However, operations like intersection and union preserve set immutability, meaning that relevancy depends on the (non) overlap between two sets (i.e. how many and which elements of set 1 and set 2 are identical). This is contrary to the intuition behind DFS meanings, which is that propositions that are true in many of the same models $\in \mathcal{M}$ are relevant to each other (i.e. not just identical vectors, also similar vectors). Furthermore, these operations are not directional. Section 1.1 already illustrated the desirability of directionality in function application.

A more appropriate way to formalize function application is in terms of entailment. A vector $\vec{v}$ entails another vector $\vec{r}$ if and only if the elementwise implication from $\vec{v}$ to $\vec{r}$ yields the tautological vector (the vector that is true in all models $\in \mathcal{M}$) (Venhuizen et al., 2021). In other words, $\vec{v} \models \vec{r}$ means that in every model where $\vec{v} = 1$, $\vec{r} = 1$ too. Entailment is dependent on the co-occurrence of vectors across the meaning space and is also directional and thus seems to be a good candidate for function application in DFS.

$$\vec{v} \models \vec{r} \text{ iff } \vec{v}_i \rightarrow \vec{r}_i = 1 \text{ for all } i \in |\mathcal{M}| \tag{2.3}$$

Entailment as in eq. (2.3) only operates over single vectors, but our subpropositional denotations are sets of vectors. Hence the function application of a word a to a word b is defined as the set of vectors in a that are entailed by at least one vector in b. The formalization is given in eq. (2.4). The definition in eq. (2.4) ensures two things. Firstly, since a vector always entails itself, if a vector is contained in both functor and argument, it is guaranteed to be in the denotation of the reduced expression. Secondly, this definition allows the structure of the meaning space to percolate to the propositional level compostionally. Namely, if a certain entailment relation exists in a meaning space between, say, 'playing trumpet' and 'having a trumpet', then this definition allows for both of these predicates to play a role in the denotation of 'playing trumpet' and further expressions involving someone playing trumpet. In other words, if playing a trumpet entails having one, then play(trumpet, Chet) $\models$ have(trumpet, Chet) as well.

$$\mathbf{fa}(\mathbf{a}, \mathbf{b}) = [\![\mathbf{a}]\!]([\![\mathbf{b}]\!])_{\models} = \{\vec{v} \mid \exists \vec{x}.\, \vec{v} \models \vec{x} \in \mathbf{a}\} \text{ for all } \vec{v} \in \mathbf{b} \tag{2.4}$$

For example, we can derive the meaning of **sings(Chet)** by performing $[\![\mathbf{sings}]\!]([\![\mathbf{chet}]\!])_{\models}$.[1] Since the vector denoting **sings(Chet)** is contained in both the denotation of **sing** and the

[1]Some notational clarification here is in order. The sentence "Chet plays trumpet" is formalized in predicate logic as *plays(trumpet, chet)*, and has the *logical form* **plays(trumpet)(chet)**. The meaning of the predicate logic notation and the logical form notation is equivalent, but the logical form emphasizes the compostional structure of the sentence.

denotation of **Chet**, this vector is ensured to be in the output set. In a meaning space with no further propositional structure, the output of this operation would thus be {**sings(Chet)**}. However, in a meaning space where there are additional constraints on propositional truth the denotation of the final expression may include more vectors. For example, if the meaning space is constructed with the constraint that one may only sing if and only if one has a microphone, then the proposition **have(microphone, Chet)** will be entailed by **sings(Chet)**. Thus the meaning is as follows.

(6) $[\![\textbf{sings(Chet)}]\!]^{\mathcal{S}} = \{\text{sings(Chet)}, \text{have(microphone, Chet)}\}$

However, there is a shortage of representational power in mere sets. We can illuminate on this using an example derivation of an expression involving a two-place predicate and the thus far provided definitions. The logical form of the sentence "Tina teases mike" is given in eq. (2.5).

$$\textbf{(tease(mike))(tina)} \tag{2.5}$$

For the sake of argument, we ignore any structural propositional correspondence that has influence the meaning of this expression that may exist in an arbitrary meaning space $\mathcal{S}$. Since the denotation of *tease* is defined as the set of all propositions which pertain to the word **tease**, it is a set which contains both **tease(mike, tina)** and **tease(tina, mike)**. The denotation of the word **mike** is the set of all propositions pertaining to the entity Mike. This set thus also contains the propositions in which **mike** functions as the object of **tease** and the proposition in which **mike** is the object. As mentioned before, the operation in eq. (2.4) ensures that vectors present in both sets percolate to the output set as identical vectors necessarily entail themselves. Similarly, when applying **tease(mike)** to **tina**, the vectors denoting **tease(mike, tina)** and **tease(tina, mike)** percolate to the output. Thus the result of eq. (2.5) contains both **tease(mike, tina)** and **tease(tina, mike))**. Using the same reasoning, we can show that the output of reverse argument ordering **(tease(tina))(mike)** also contains both of these propositions.

From this example we may draw the conclusion that the mechanism in its current form cannot properly capture the order of operations in natural language semantics. From here we may conclude that the definition in eq. (2.4) is not refined enough, or that the underlying representations of sets are not rich enough to ensure the desired outputs. Both of these hypotheses are explored in what follows below, in which the mechanism is enriched by traditional λ-calculus.

2.2 Lambda-calculus in DFS

The representational currency of sets of meaning vectors is not enough to capture the crucial interaction of argument ordering with the resulting meanings. One way to enrich the representation of sub-propositional phrases is to draw upon traditional λ-calculus. This enrichment can allow us to explicitly specify the order in which a function is to be applied to incoming arguments. The definition given in eq. (2.1) makes intuitive sense given the described trajectory of neural network outputs through the meaning space. However, it can be refined using lexical denotations from typed λ-calculus. Classical type theory assigns types to logical objects, namely entities (individuals) are of type e and sentences (truth-values) are of type t. These basic types may be combined into more complex types, which represent functions with an input and output. All which λ-calculus adds to type theory is notational clarity, allowing us to neatly represent

partial and curried functions. Now, one-place predicates are functions from entities to truth values (et, takes an entity as its argument and returns a truth-value), two-place predicates are functions of type eet (takes an entity and returns a function form entities to truth values) and so on. The machinery I introduce in this section will be referred to as 'λ-DFS'.

Type t expressions in DFS can be easily defined as sets of meaning vectors. Type e expressions can be derived from the underlying structure of a meaning space. A meaning space $\mathcal{S}$ is composed of $|\mathcal{M}|$ formal models, which all consist of a tuple $\langle U, V \rangle$ (a universe and interpretation function) which defines the legal entities in that model. The combined universe of all $M_i \in \mathcal{M} \in \mathcal{S}$ contains the set which describes the domain of all legal type e expressions in the space $\mathcal{S}$ ($D_e^{\mathcal{S}}$). The underlying model theoretic properties of these entities are already exploited in the definition of quantification in vanilla DFS. For example, $[\![\exists x.\phi(x)]\!]$ is defined as the vector disjunction over an assignment function that replaces x with $e_1 \dots e_n \in D_e^{\mathcal{S}}$ in ϕ (see Venhuizen et al. (2021) for a formal definition of the existential quantifier in DFS).

$$D_e^{\mathcal{S}} = \bigcup_{i \in |\mathcal{M}|} U_{\mathcal{M}_i} \tag{2.6}$$

We can then properly define lambda functions whose internal logical currency is expressed in a set of vectors. For example, the word 'sing' (et) or the word 'play' (eet) can be defined as follows.

$$[\![sing]\!]^{\mathcal{S}} = \{\vec{v} \in A \mid \vec{v} \models \lambda x.sing(x)\} \tag{2.7}$$

$$[\![play]\!]^{\mathcal{S}} = \{\vec{v} \in A \mid \vec{v} \models \lambda x \lambda y.play(x)(y)\} \tag{2.8}$$

The denotation $[\![sing]\!]^{\mathcal{S}}$, for example, may be read as 'the set of all atomic propositions (A) that entail *sing(x)* (where x is a λ-bound variable).[2] These functions cannot be interpreted with respect to a space, because there are free/λ-bound variables in their denotations. However, the denotations can gain a correspondence between λ-expressions and interpretable sets of meaning vectors if mediated with an existential closure operation. Existential closure was first introduced by Heim (1982) in the context of indefinite noun phrases and discourse referents and has since been used elsewhere in semantics, for example when dealing with Neo-Davidsonian event semantics (Champollion, 2011). Here, it serves as a mapping between functions ranging over sets and sets themselves.

(7) The unselective binder $\exists$ binds all free or λ-bound variables within its scope.

Applying existential closure to a denotation that still involves unresolved lambda terms results in proposition with one or more quantifiers. Crucially, the meaning vector associated this proposition given a DFS space can be derived, because DFS is functionally complete on the propositional level. For example, applying existential closure to $\lambda x.sing(x)$ yields $\exists x.sing(x)$ which in natural language may yield "Someone sings". Existentially binding a lambda expression thus results in an interpretable proposition which we can use to derive the set-theoretic denotation of words in DFS. Namely, the meaning of a lambda expression in DFS can be defined as the

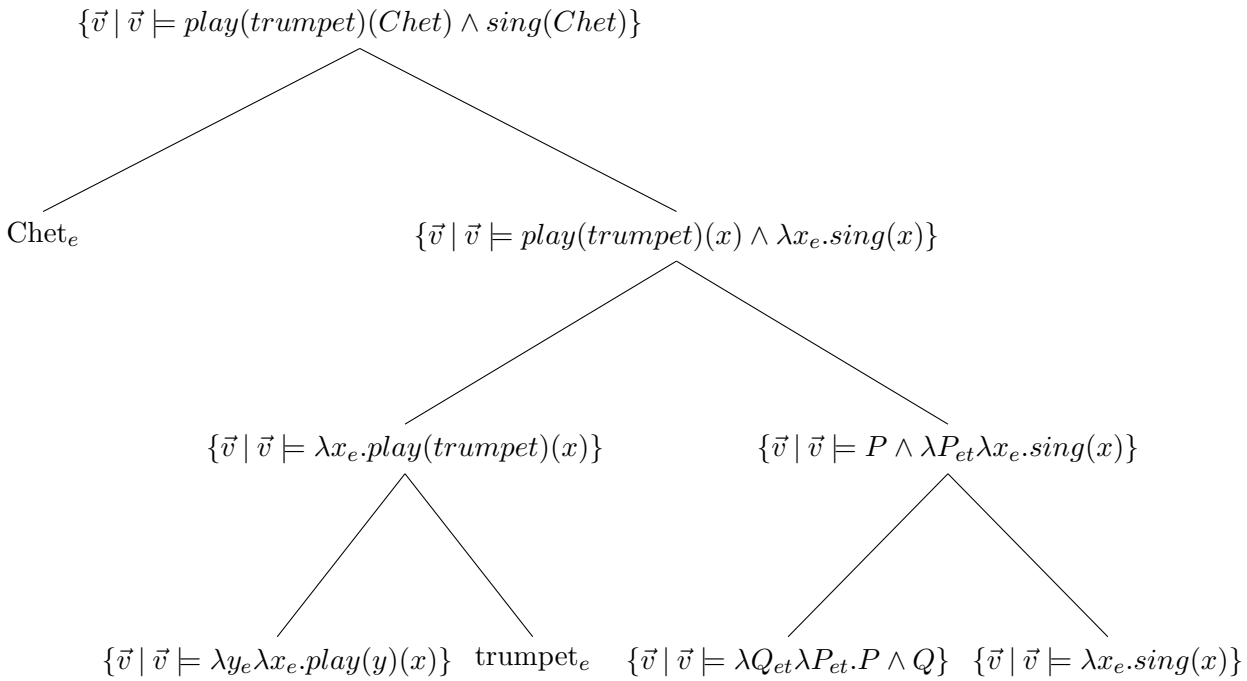
[2] Henceforth, $\vec{v}$ will be shorthand for $\vec{v} \in A$

set of atomic vectors in the meaning space that entail (eq. (2.3)) the existential closure of the expression.

$$[\![\lambda x, \ldots, \lambda n.\phi(x, \ldots, n)]\!]^{\mathcal{S}} = \exists(\lambda x, \ldots, \lambda n.\phi(x, \ldots, n)) = \exists x, \ldots, \exists n.\phi(x, \ldots, n) \tag{2.9}$$

Ordinary β-reduction can already yield interpretable function application in terms of DFS spaces. Applying these functions to arguments of type e using β-reduction results in a reduced expression, which can be directly interpreted in terms of the meaning space by applying existential closure. In this fashion all vocabulary items that accept e as input can be used to reconstruct meanings and at the same time be mapped back onto truth-value-vectors in intermediate β-reduced constituents.

This method of existentially closing lambda terms to map back to the meaning space already allows DFS to inherit the strengths of λ-calculus, as there is now an equivalence between typed lambda functions and sets of meaning vectors, and by extension real-valued points in meaning space. Figure 2.1 exemplifies how the mapping between sets of meaning vectors and lambda terms may work in practice.



$[\![play]\!]^{\mathcal{S}} = \{\vec{v} \mid \vec{v} \models \exists y \exists x.play(y)(x)\}$
$[\![sing]\!]^{\mathcal{S}} = \{\vec{v} \mid \vec{v} \models \exists x.sing(x)\}$
$[\![and]\!]^{\mathcal{S}} = \{\vec{v} \mid \vec{v} \models \exists Q \exists P.P \wedge Q\}*$
$[\![and(sings)]\!]^{\mathcal{S}} = \{\vec{v} \mid \vec{v} \models \exists P \exists x.P \wedge sing(x)\}$
$[\![play(trumpet)]\!]^{\mathcal{S}} = \{\vec{v} \mid \vec{v} \models \exists x.play(trumpet)(x)\}$

Figure 2.1: An example derivation of a first order logic sentence in formal semantics with equivalent set denotations. Additionally, an example interpretation function is provided to illustrate the use of $\exists$-closure. *Note that the denotation of the operator 'and' here is purely illustrative, the more precise denotation of logical operators is discussed in section 2.3.

One of the effects of the conspiracy between $\exists$ closure, entailment and β-reduction, the type t set of meaning vectors is not necessarily singleton anymore. If after maximal β-reduction there

are still vectors $\in A$ that entail the vector representing the first-order-logic formula other than itself, then these vectors will be contained in the final denotation (and will thus be a set with a cardinality of > 1). This trait allows λ-DFS to represent DFS world knowledge, represented in the form of propositional co-occurrence in the meaning space, to percolate to sentence-level denotations. A simple example of this is the two propostions represented by the same vector. These vectors then have the same truth-conditions in DFS-space and thus, for all intents and purposes, denote exactly the same meaning. Hence compositionally constructing either of these propositions results in a set containing both elements.

Existentially closing lambda terms in fig. 2.1 already gives rise to interesting implications for the meaning of sub-propositional elements in DFS. For example, the meaning of the logical operator *and*. Existentially closing this denotation would result in $\exists P.\exists Q.\{\vec{v} \mid \vec{v} \models P \wedge Q\}$ (the set of vectors that entail any conjunction between two atomic propositions in the meaning space). However, this would force us to define $\exists$-closure over higher order logic and also to draw upon higher order elements of the underlying model structure. Additionally, we would like vocabulary items that take complex types (like the logical operator 'and') to operate over the same logical currency. That is, logical operators must necessarily allow for sets to be in their domain. Hence it is a worthwhile exercise to consider denotations of logical operators in λ-DFS that address these issues. Therefore we must devise a boolean algebra for sets of meaning vectors (similarly to how Venhuizen et al. (2021) define a boolean algebra for single meaning vectors) in order to interpret words of this type. This is what is presented in the following section.

2.3 Logical operators in lambda-DFS

Similarly to how Venhuizen et al. (2021) present a boolean algebra over propositional meaning vectors, I attempt to formalize a boolean algebra over subpropositional sets of meaning vectors. There are a number of criteria that these operations must adhere to. Namely, they must operate over sets of meaning vectors, be an extension of the already defined logical operations over single meaning vectors and lastly must be able to be formalized as λ-expressions.

Perhaps the most straightforward operator to define is negation. The negation of a set of n meaning vectors $\{\vec{v}(p_1), \ldots, \vec{v}(p_n)\}$ is the set of the vector negations of those n meaning vectors $\{\vec{v}(\neg p_1), \ldots, \vec{v}(\neg p_n)\}$ (using the definition of vector negation in eq. (1.3)). This definition preserves some of the properties of vector negation. Namely, the set average (which returns a vector) of a word added to the set average of its negation returns the tautological vector (see section 2.5 for the specification of set averages). Here the negation over sets of propositions is denoted by $\neg_\times$.

$$\neg_\times \mathbf{a} = \{\vec{v}(\neg p) \mid \vec{v}(p) \in \mathbf{a}\} \tag{2.10}$$

A λ-formalization of the word '*not*' is as follows.

$$[\![not]\!]^{\mathcal{S}} = \neg_\times \{\vec{v} \in A \mid \lambda P \lambda x. \models P(x)\} \tag{2.11}$$

Slightly less straightforward is the definition of binary logical operators over sets, like conjunction. One issue with the denotation provided in fig. 2.1 is that, since a meaning space contains only atomic propositions, it is possible that the atomic propositions of a meaning space

$\mathcal{S}$ contain no conjunctions and therefore often returns the empty set. On a formal level, it would be desirable to reason over all possible conjunctions in $\mathcal{S}$. This is achievable with a new operator that overrides the traditional conjunction for sets of meaning vectors. Namely, it is the operator $\wedge_\times$ such that $\mathbf{a} \wedge_\times \mathbf{b}$ is equal to the vector conjunction of each tuple in the Cartesian product $a \times b$.

$$[\![\mathbf{a} \wedge \mathbf{b}]\!]^{\mathcal{S}} = \{\vec{v} \in A \mid \vec{v} \models \mathbf{a}\} \wedge_\times \{\vec{v} \in A \mid \vec{v} \models \mathbf{b}\} \tag{2.12}$$

$$\mathbf{a} \wedge_\times \mathbf{b} = \{\mathbf{a}_i \wedge \mathbf{b}_j \mid (\mathbf{a}_i, \mathbf{b}_j) \in \mathbf{a} \times \mathbf{b}\} \tag{2.13}$$

Since in eq. (2.12) and eq. (2.13) both $\mathbf{a} \subseteq A$ and $\mathbf{b} \subseteq A$, the result of $\mathbf{a} \times \mathbf{b} \subseteq A \times A$. Hence the definition of set conjunction satisfies the property eq. (2.14). This machinery allows us to reason over the full range of possible conjunctions in $\mathcal{S}$.

$$\{\vec{v} \in A \mid \vec{v} \models \mathbf{a}\} \wedge_\times \{\vec{v} \in A \mid \vec{v} \models \mathbf{b}\} = \{\vec{v} \in A \wedge_\times A \mid \vec{v} \models \mathbf{a} \wedge \mathbf{b}\} \tag{2.14}$$

As mentioned before, one remaining issue is that, depending on the state of the compositional process, $\exists$-closure may cause us to quantify over properties (in the type-theoretic sense) rather than just entities. Such can be seen in the denotation of *and* in the interpretation function given in fig. 2.1. In order to preserve consistency of being able to infer a set of meaning vectors at any point in the compositional process, we can back off to second order logic to help us infer the set of meaning vectors of a denotation that contains unresolved properties. Similarly to how we derive the overarching domain of entities in the meaning space in eq. (1.1), we can derive the domain of properties *et* over the whole space $\mathcal{S}$. This allows us to define quantification over properties analogously to how Venhuizen et al. (2021) define quantification over entities, namely by taking the conjunction or disjunction over an assignment function over properties for the universal and existential quantifiers respectively. Equation (2.15) provides the definition of this quantification for the existential quantifier.

$$[\![\exists P.\phi]\!]^{\mathcal{S}} = \vec{v}(\exists P.\phi) \text{ st: for all } i \in |\mathcal{M}|\ \vec{v}_i(\exists P.\phi) = 1 \text{ iff } M_i \models \phi^{g[P/Q_1]} \vee \ldots \vee \phi^{g[P/Q_n]} \tag{2.15}$$

Algorithm 1 demonstrates the procedure for generating the conjoined set of two denotations. One loops over the Cartesian product of the sets of both denotations (using $\exists$-closure) and iteratively updates the a new set with the conjunctions of the tuples in the Cartesian product.

Algorithm 1 Set conjunction

procedure CONJOIN(a, b) ▷ The conjunction of denotation a and denotation b
  $a' \leftarrow \{\vec{v} \mid \vec{v} \models \exists(a)\}$ ▷ Infer the set of meaning vectors for denotation a and b
  $b' \leftarrow \{\vec{v} \mid \vec{v} \models \exists(b)\}$
  $S \leftarrow \emptyset$ ▷ Initialize S as the empty set
  for $i \leq |a'|$ **do**
    for $j \leq |b'|$ **do**
      $S \leftarrow S \bigcup \{(a'_i \wedge b'_j)\}$
    end for
  end for
  return S ▷ S contains all conjunctions in the cartesian product of a' and b'
end procedure

The formalization of the natural language operator 'and' is given in eq. (2.16).

$$[\![and]\!]^{\mathcal{S}} = \lambda P \lambda Q \lambda x.P(x) \wedge_{\times} Q(x) \tag{2.16}$$

With the denotations of negation and conjunction in place, we now have functional completeness over the propositional operators with sets in their domain. This means that all basic logical operators can be rewritten as a combination of negation and conjunction given the above definitions. However, the denotations of other operators simplify to eq. (2.12) with the operator in question replacing $\wedge$.

Operator	Denotation	Domain
$\vee_{\times}$	$\neg_{\times}(\neg_{\times} a \wedge_{\times} \neg_{\times} b)$	$A \vee_{\times} A$
$\oplus_{\times}$	$(a \vee_{\times} b) \vee_{\times} (a \wedge_{\times} b)$	$A \oplus_{\times} A$
$\rightarrow_{\times}$	$\neg_{\times} a \vee_{\times} b$	$A \rightarrow_{\times} A$
$\leftrightarrow_{\times}$	$(a \rightarrow_{\times} b) \wedge_{\times} (b \rightarrow_{\times} a)$	$A \leftrightarrow_{\times} A$

Table 2.1: Subpropositional functional completeness in λ-DFS.

2.4 Quantification in λ-DFS

Intuitively, quantification could be formalized using the definition of a set of meaning vectors (eq. (2.1)), $\exists$-closure and the vanilla DFS definition of quantified propositions given in Venhuizen et al. (2021). However, this has similar issues to the ones mentioned in section 2.3 with respect to conjunction. Namely, if a meaning space S only has atomic propositions, then it is possible that none of them entail a quantified sentence and the denotation in λ-DFS would be the empty set. This is especially true for universal quantification, which is quite restrictive in its truth conditions. It is unlikely that an atomic proposition, say for example *pay(john)* would ever entail a quantified proposition like $\forall x.pay(x)$. The requirements for quantification in λ-DFS are thus that the set representing a quantified proposition contains at least that proposition itself, and additionally any strongly correlated meanings (in the form of entailments).

With the definitions presented in this section, it is possible to define quantification that satisfies these properties. Additionally, it can be done so in a fashion in line with the definition given in Venhuizen et al. (2021). The interpretation of, for example, the universal quantifier is simply the conjunctive closure over some assignment function, except rather than ordinary conjunction it is defined with the set conjunction as given in eq. (2.13). Equation (2.17) gives the definition of the sentence '*Everyone plays trumpet*'.[3]

$$\forall^{\times} y.[\![\text{play(trumpet,y)}]\!]^{\mathcal{S}} = \bigwedge_{i \leq n} \{\vec{v} \mid \vec{v} \models \text{play(trumpet,y}\}^{[y/e_i]} \tag{2.17}$$

The result of eq. (2.17) contains the all conjunctions in the Cartesian product of the sets obtained via the assignment function. If there is no strong propositonal co-occurence imposed on playing trumpet in this hypothetical meaning space, then the result is a singleton set containing the vector 'play(trumpet, chet) $\wedge$ play(trumpet, miles) ...' which is exactly the propositional

[3] I use $\forall^{\times}$ to denote this type of quantification over sets rather than $\forall_{\times}$ in order to avoid confusion between $\forall_{\times}$ and $\forall x$. Additionally, $\bigwedge$ is equivalent to a sum of set conjunctions $(\phi_1 \wedge_{\times} \ldots \wedge_{\times} \phi_n)$

meaning of '$\forall y$.play(trumpet, y)' derived using the operations provided by Venhuizen et al. (2021). If there are any world knowledge constraints in the meaning space, then the resulting set is potentially larger. For example, a reasonably imaginable constraint on this hypothetical meaning space is that *playing* a trumpet entails *having* a trumpet. If this is the case, the result of eq. (2.17) will also contain the conjunction 'has(trumpet, chet) $\wedge$ has(trumpet, miles) . . . '. Thus the two criteria mentioned before are satisfied in this definition.

The question that remains is then how we can formalize single determiners using the traditional λ-calculus and the operations over set introduced here. An example formalization of the determiner '*everyone*' is given in eq. (2.18) and the interpretation is worked out in eq. (2.19).

$$[\![everyone]\!]^{\mathcal{S}} = \lambda P.\forall^{\times}\mathrm{y}.\mathrm{P(y)} \tag{2.18}$$

$$\exists(\lambda P.\forall^{\times}\mathrm{y}.\mathrm{P(y)}) = \exists P. \bigwedge_{i \leq n} \{\vec{v} \mid \vec{v} \models \mathrm{P(y)}\}^{[y/e_i]} \tag{2.19}$$

The definition in eq. (2.19) may seem complicated at first glance, but it follows directly from the definition of quantification in vanilla DFS and the definitions given in this section. Recall that second-order-existential-closure can be reduced to a disjunction over an assignment function over *et* predicates. Thus each element of the sum of conjunction is itself a set representing a disjunction over all *et* predicates. In the hypothetical meaning space, the first element of the sum of conjunctions is for instance $\{\vec{v} \mid \vec{v} \models play(trumpet)(chet) \vee \ldots \vee sing(chet)\}$, in which case the assignment function replaces 'y' with the entity 'chet'. Equation (2.19) shows an interesting interaction between $\exists$-closure and set quantification which crucially does not clash. This is because $\exists$-closure is a second-order-logic operation quantifying over meaning vectors, whilst set quantification ranges over sets. The intuition behind this interpretation is that the word 'everyone' denotes the set of propositions that entail *everyone doing something*.

Of course the existential determiner 'someone' can be defined equivalently with set disjunction rather than conjunction.

$$[\![someone]\!]^{\mathcal{S}} = \lambda P.\exists^{\times}\mathrm{y}.\mathrm{P(y)} \tag{2.20}$$

$$\exists(\lambda P.\exists^{\times}\mathrm{y}.\mathrm{P(y)}) = \exists P. \bigvee_{i \leq n} \{\vec{v} \mid \vec{v} \models \mathrm{P(y)}\}^{[y/e_i]} \tag{2.21}$$

2.5 Mapping sets to $\mathbb{R}^n$

We now have a decent compositional mechanism for deriving sentence meaning in DFS. However, the representational currency no longer consists of vectors as in vanilla DFS, but rather consists of sets of such vectors. In order to preserve some of the attractive linear algebraic properties of vanilla DFS, as well as to mimic the neural network outputs, we need some sort of mapping between the representation of meaning as sets and as vectors.

One intuitive way to define such a mapping is to calculate the average vector (also referred to as the *centroid*) given a set of logical vectors. This function then takes any set of logical vectors and returns a vector $\in \mathbb{R}^{|\mathcal{M}_{\mathcal{P}}|}$.

$$centroid(\mathbf{a}) = \frac{\sum_i^{|\mathbf{a}|} \mathbf{a}_i}{|\mathbf{a}|} \tag{2.22}$$

The centroid of a singleton set is then equal to said sets only element. This follows the intuition that in λ-DFS, singleton sets represent the meaning of a proposition (that has no strong world-knowledge co-occurrence with other propositions). In the case of a set with a larger cardinality (> 1), each value of the centroid represents the ratio of models that are satisfied in the set. Each element is thus essentially the fuzzy logic truth value, or the degree of truth, of each $M_i \in \mathcal{M}$.

Another property of the centroid of a set $\mathbf{a}$, is that it necessarily entails the existential closure $\exists(\mathbf{a})$. This follows from the definition of entailment (eq. (2.3)) and of the definition of sets of meaning vectors. By definition, each element of $\mathbf{a}$ must entail $\exists(\mathbf{a})$. The condition on each element $\mathbf{a}_i$ is thus that where $\vec{v}(\exists(\mathbf{a}))$ is equal to 0, $\vec{v}(\mathbf{a}_i)$ may not be equal to 1 and thus must be equal to 0. The degree of truth for these models in the centroid hence also equals 0 and the condition for entailment is satisfied.

Chapter 3

Compositionally Derived DFS-Semantics

In order to lend the concepts introduced in chapter 2 some tangibility, this section contains worked examples using a real meaning space sampled using the algorithm provided by Venhuizen et al. (2021). In what follows the specification of the meaning space is provided, after which I work through an example of logical operators and quantification using λ-DFS. Additionally, I give some extra intuition on the logical consequences of negation and conjunction. Lastly, I show that λ-DFS, much like the RNN, defines a trajectory through the meaning space with every compositional step. The implementation of λ-DFS is built in Python (https://github.com/LuukSuurmeijer/dfs_semantics_python).

3.1 World specification

In order to showcase the properties and capabilities of λ-DFS, we must take a look at an example meaning space with a somewhat interesting knowledge structure. Venhuizen, Hendriks et al. (2019) provide a prolog implementation of vanilla DFS (DFS-tools)which here is used to sample a meaning space consisting of 1000 models and 26 atomic propositions. The underlying models consist of a number of entities that either enter a bar or restaurant, eat and/or drink something, and pay and leave the venue. The simplicity of this world gives rise to a number of hard constraints. An entity may not enter two locations at the same time, and someone may not order something and leave without paying. Similarly a number of intuitive probabilistic constraints arise from the model specification, namely that someone is more likely to eat pizza at a restaurant than at a bar and that someone is more likely to enter an establishments if there is already another person there. Individuals are also less likely to order something else if they have already ordered something. In all situations that are not explicitly governed by these probabilistic constraints, the prior distribution used for sampling is uniform (a coin flip). Furthermore each entity has their own eating and drinking preferences. The hard constraints on the meaning space are given in table 3.1.

The meaning space in λ-DFS is instantiated with a type signature, which assigns types to all predicates and variables that occur in any of the atomic propositions in the meaning space. Partial types can be inferred based on the type signature. Table 3.2 gives an overview of all type domains in the sampled world, including partial types.

No.	Constraint	Description
1.	$\forall x \forall y \forall z.enter(y,x) \wedge z \neq y \rightarrow \neg enter(z,x)$	Someone may not enter two places at the same time.
2.	$\forall x \forall y.(eat(y,x) \vee drink(y,x) \rightarrow order(y,x)$	If someone eats or drinks something, then they must have ordered it.
3.	$\forall x \forall y.enter(y,x) \wedge pay(x) \rightarrow \exists z.order(z,x)$	If someone enters somewhere and pays, then they must have ordered something
4.	$\forall x.askmenu(x) \wedge pay(x) \rightarrow \exists z.order(z,x)$	If someone has seen the menu and pays, then they must have ordered something
5.	$\forall x \forall y.leaves(x) \wedge order(y,x) \rightarrow pay(x)$	If someone leaves and orders something, then they must have paid

Table 3.1: Hard constraints on the world in first-order-logic and natural language.

Domain	Components
D_e	john, ellen, pizza, fries, wine, beer, restaurant, bar
D_{et}	askmenu(john), askmenu(ellen), pay(john), pay(ellen), leave(john), leave(ellen), enter(restaurant), enter(bar), order(wine), order(beer), eat(fries), eat(pizza)
D_{eet}	enter(restaurant)(john), enter(restaurant)(ellen), enter(bar)(john), enter(bar)(ellen), order(beer)(john), order(beer)(ellen), order(wine)(john), order(wine)(ellen), eat(fries)(john), eat(fries)(ellen), eat(pizza)(john), eat(pizza)(ellen)

Table 3.2: The type domains and their components (in logical form) of the world.

Figure 3.1 represents the structure of the meaning space in terms of inference scores (eq. (1.8)). The inference score $inf(p,q)$, in essence, represents the degree to which the (un)certainty about a proposition p changes if one knows q. This figure thus visualises the type of constraints and inferences present in this meaning space. This allows it capture logical entailments to some extent.

In addition to the atomic propositions, fig. 3.1 also demonstrates the nature of existential closure over word-level denotations as defined in eq. (2.9). Because the existential quantifier distributes over disjunction, the existentially closed denotations are easy to satisfy in a given model. This has the effect that the vectors representing these FOL formulae contain many true values and, depending on the meaning space specification, may even be tautological. This has the consequence that many entailment relations exist between these propositions and the other atomic propositions in the space, many of which are represented by a perfect inference score. We must be careful when interpreting inference scores of (nearly) tautological vectors. One might suspect the inference scores of these vectors to often return 1 as (almost) tautological vectors represent some sort of logical entailment, but this is not the case. For example, even though it holds that $\exists xy.order(x,y) \models$ order(beer, john) the inference score is 0 (assume $\exists xy.order(x,y)$ is tautological). This follows from the definition of the inference score. Since the conjunction of a tautological vector q with any other vector p just returns p, the conditional probability of $P(q|p)$ then simplifies to $\frac{P(p)}{P(p)} = 1$. This leaves us with $inf(p,q) = \frac{1-1}{1} = 0$. In words, one cannot increase their certainty about something they already know. This fact shows that the inference score does not represent all perfect entailments with a value of 1 and that we must interpret it

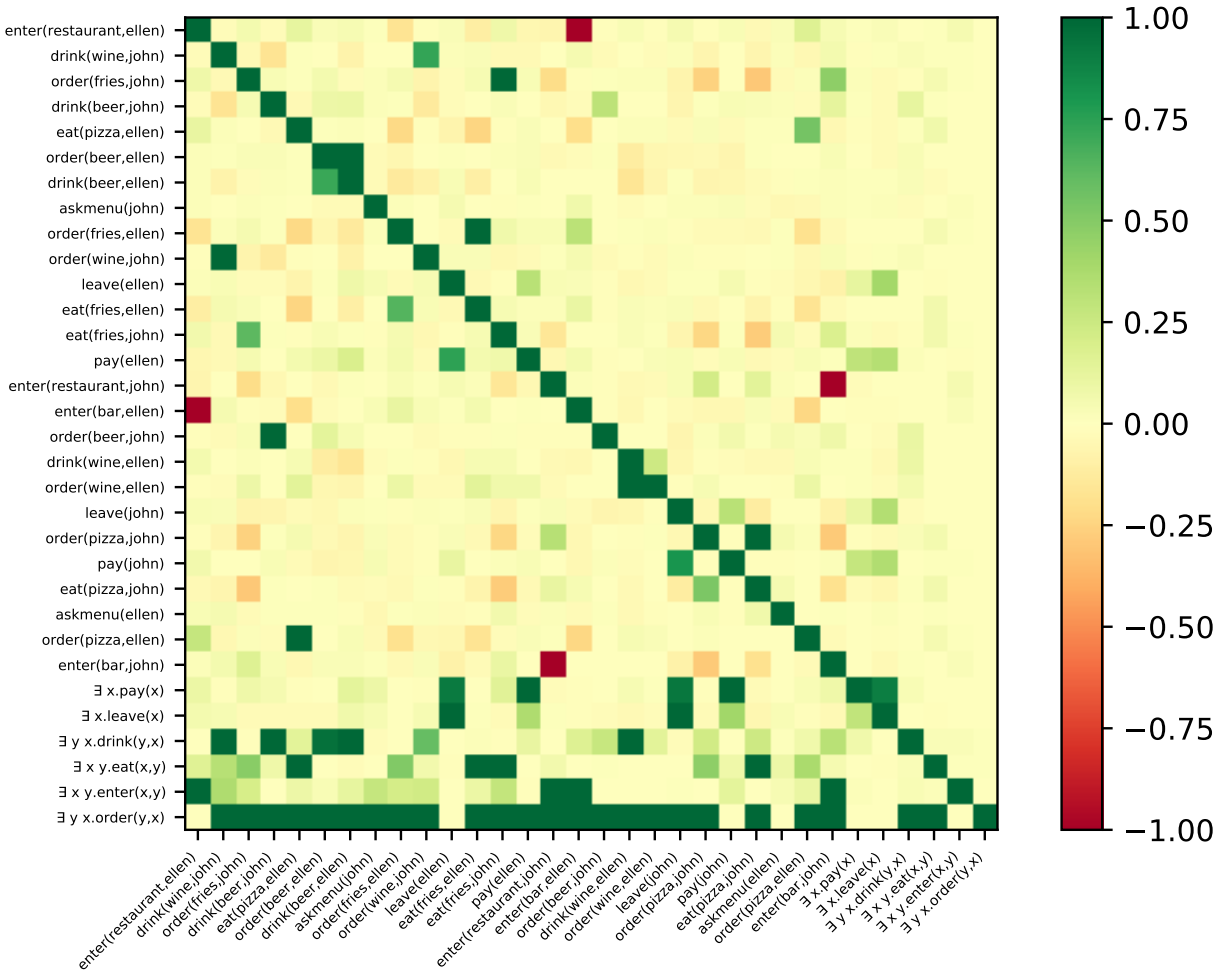


Figure 3.1: Inference score matrix of the example meaning space. The color of each index i, j represents the inference score $inf(i, j)$.

purely as the degree to which certainty is increased or decreased.

3.2 Working examples

Deriving sentence meaning using λ-DFS With the meaning space and definitions in place, we can now observe the behavior of the computational mechanism in λ-DFS. Figure 3.2 presents inference scores at every node in the compositional structure for the sentence "*John entered the restaurant and ordered wine*". Overall, the inferences deeper in the tree are much weaker than the inferences at nodes where some of the λ-terms have already been resolved. This is consistent with the scores observed in fig. 3.1, which are centered around 0 for existentially closed denotations. The operator *and* also exhibits no particular inference with respect to any of the propositions displayed. This is again because the existential closure of this denotation is a very weak statement, indicating the existence of any conjunction in the meaning space.

Resolving λ-terms solidifies existing inferences and creates new ones. Composing *enter* with the entity *restaurant* increases the certainty in proposition 2 and 4, whilst decreasing the certainty of proposition 1. Despite not knowing yet who the agent of the *enter* predicate is, merely knowing that *someone* entered a restaurant is enough to negatively infer that john entered a *bar* to a certain extent, given the world knowledge constraint that individuals enter only one establishment

at a time. Similarly for the predicate *order*, the certainty in proposition 3 and 5 increases by knowing that someone ordered wine. The inferences here are weaker than the *enter* predicate, because there are less restrictions on the *order* predicate (one may order both a food item and/or a beverage, whereas one may only enter a restaurant or a bar). The positive inference with respect to proposition 5 highlights the strong probabilistic constraint that ordering something implies drinking/eating it.

The penultimate compositional step (before β-reducing *John*) displays that the relative inference scores are already determined, but not quite as strong as they are in the sentence-final meaning (some of the inferences are entailments, after all). This is because the meaning lacks an agent for both predicates in the main conjunct. The existentially closed denotation of this node is, in words, "*Someone enters a restaurant and orders wine*". This obviously increases the certainty in either *Ellen* performing these actions or *John*, or both. Similarly for proposition 1, *someone entering a restaurant* decreases the certainty in *John* entering a *bar*, but it could still be the case that it is in fact *Ellen* who enters a *restaurant* while *John* enters a *bar*. Since the agent is still unknown the mechanism cannot commit fully to either interpretation. When *John* is β-reduced into the unfolding denotation, the entailments are resolved. Note that *John* is an entity rather than a function in this space, and hence the inference scores over just *John* are not defined. The β-reduced meaning results in a negative entailment for proposition 1 and positive entailment for propositions 2 and 3. Propositions 4 and 5 display a weaker positive inference, and proposition 6 has no significant inference with the sentence-final meaning (because this proposition is essentially irrelevant to the meaning of this sentence).

Negation Negation as defined in chapter 2 carries some interesting logical consequences. Propositional negation in vanilla DFS has two interesting properties. Firstly two meaning vectors $\vec{v}(a)$ $\vec{v}(\neg a)$ are mutually orthogonal (perpendicular) meaning that the dot product $\vec{v}(a) \cdot \vec{v}(\neg a) = \vec{0}$. Conversely, the addition $\vec{v}(a) + \vec{v}(\neg a) = \vec{1}$.[1] The property of orthogonality percolates to the level of sets to some extent, because by definition the set $\mathbf{a}$ and $\neg\mathbf{a}$ are pairwise orthogonal. Meaning that $\vec{v}(a_i) \cdot \vec{v}(\neg a_j) = \vec{0}$ for all $i = j$ in some set of vectors a. Similarly, these sets satisfy the additive condition pairwise.

In order to satisfy these properties to the full extent, we must reduce both sets to the domain of vectors using the centroid (eq. (2.22)). One can verify the property of orthogonality over the centroids in this specific meaning space empirically. Table 3.3 represents the cosine similarity of the centroid of each predicate with its negation. Note that a cosine similarity of 0 indicates orthogonality and a cosine similarity of 1 indicates an angle of 0 between the vectors (i.e. they point in the same direction). First and foremost, none of the vectors are orthogonal, with some in fact being relatively similar. An interesting observation is that the cosine similarity seems to be significantly higher for two place predicates than for one place predicates. And between two place predicates, the more probable one (ordering usually implies eating or drinking in the this world) has even higher cosine similarity.

The lack of orthogonality can be explained by looking more closely at the interaction between negation and $\exists$-closure. Namely, the elements of a set representing a sub-propositional meaning depends on the $\exists$-closure of the relevant λ-denotation. Strictly speaking, the external negation of a formula containing existential quantifiers, such as "someone orders beer", is "it is not the

[1] $\vec{1}$ meaning the tautological vector or the vector containing only 1.

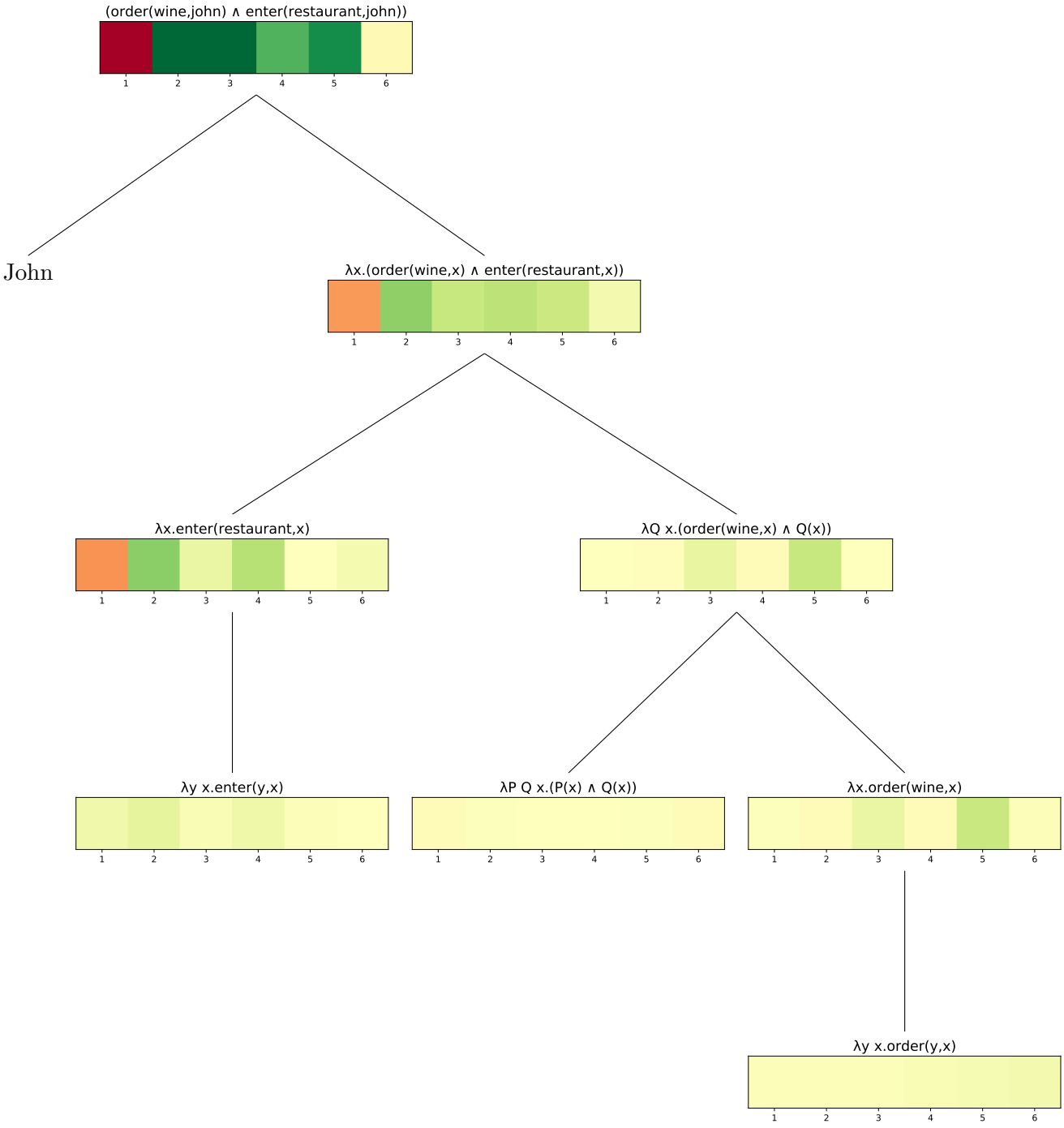


Figure 3.2: A visualization of the compositional process in λ-DFS of the sentence *order(wine,john)* $\wedge$ *enter(restaurant,john)*. Each non-entity node contains a graph representing the inference score between the meaning up until that point (the centroid of that set) and a number of related full propositions (such that each box is $inf(meaning, proposition)$). From $1-6$ the relevant 6 propositions are 1 : *enter(bar, john)*, 2 : *enter(restaurant, john)*, 3 : *enter(restaurant, john)* $\wedge$ *order(wine, john)*, 4 : *enter(restaurant, john)* $\wedge$ *order(beer, john)*, 5 : *drink(wine,john)* and 6 : *eat(fries, ellen)*.

Predicate	Cosine
$\lambda x.\{\vec{v} \mid \vec{v} \models \text{pay(x)}\}$	0.31
$\lambda x.\{\vec{v} \mid \vec{v} \models \text{leave(x)}\}$	0.32
$\lambda x \lambda y.\{\vec{v} \mid \vec{v} \models \text{eat(x,y)}\}$	0.66
$\lambda x \lambda y.\{\vec{v} \mid \vec{v} \models \text{drink(x,y)}\}$	0.67
$\lambda x \lambda y.\{\vec{v} \mid \vec{v} \models \text{enter(x,y)}\}$	0.81
$\lambda x \lambda y.\{\vec{v} \mid \vec{v} \models \text{order(x,y)}\}$	0.90

Table 3.3: Cosine similarity of each centroid predicate in the meaning space with their negation. Calculated as $cosine(centroid(a), centroid(not(a)))$.

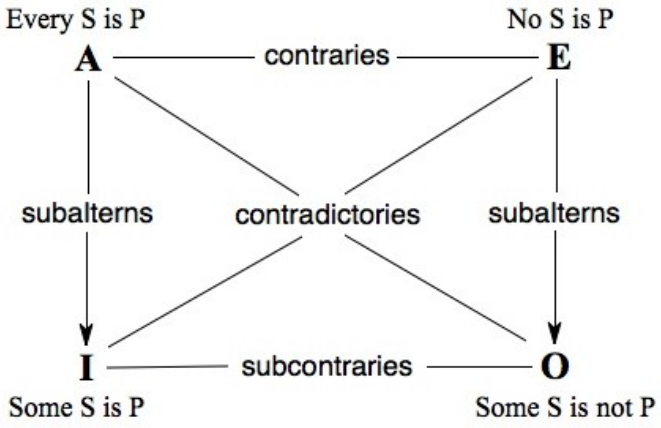


Figure 3.3: The square of opposition

case that someone orders beer" or rather "nobody orders beer" as can be observed in the square of opposition (eg. Parsons, 2017). The square of opposition is a visualization of the interaction between negation and quantifiers, here given in fig. 3.3.

However, negating a predicate like "to order" ideally results in "to not order" rather than the stronger "to never order". Additionally, forcing the definition of negation to adhere to strict external negation has some uncommon notational consequences, namely that negation would scope over λ-terms. The interaction of quantifiers with the definition of negation in this thesis results in the so-called subcontrary in the square of opposition (the subcontrary of "someone orders beer" is "someone does not order beer"). It is then clear why the latter pair is not linear algebraically orthogonal in λ-DFS, as both statements can be simultaneously true.

The second property, namely that summing a meaning vector and its negation results in the tautological vector, percolates to the centroid of sets of meaning vectors. Table 3.4 shows the probability of each centroid and its negation, which is equal to the average value of the centroid. All of these values sum to one, satisfying a number of intuitions. First of all the conjunction between a vector and its negation is the tautological vector, i.e. always true. Similarly the probability of an event (in this case a proposition) or the negation of that event occurring is equal to 1, meaning it is a perfect certainty. Lastly this table illustrates that the centroid represents the fuzzy logic truth values to a certain degree, since the negation of the centroid is equal to $1 - x$ where x is the fuzzy logic value.

The properties of λ-DFS negation illustrate a trade-off. If we were to define negation as strictly external negation in the square of opposition (and thus let the negation operator take

Predicate	$P(a)$	$P(\neg a)$
$\lambda x.\{\vec{v} \mid \vec{v} \models \text{pay(x)}\}$	0.63	0.37
$\lambda x.\{\vec{v} \mid \vec{v} \models \text{leave(x)}\}$	0.41	0.59
$\lambda x \lambda y.\{\vec{v} \mid \vec{v} \models \text{eat(x,y)}\}$	0.42	0.58
$\lambda x \lambda y.\{\vec{v} \mid \vec{v} \models \text{drink(x,y)}\}$	0.43	0.57
$\lambda x \lambda y.\{\vec{v} \mid \vec{v} \models \text{enter(x,y)}\}$	0.38	0.62
$\lambda x \lambda y.\{\vec{v} \mid \vec{v} \models \text{order(x,y)}\}$	0.48	0.52

Table 3.4: All centroids of n-ary predicates in the world, their probability and the probability of their negation.

scope over all quantifiers), the property of orthogonality would be preserved. However, then the property of tautology would be lost as the negation of a vector would now be either impossible or extremely unlikely. This implementation favors the latter property over the former, as otherwise probability theory is no longer properly defined over all meaning vectors. Furthermore to avoid scope-issues as much as possible, it is wise to keep the scope of operators as locally as possible.

Quantification Figure 3.4 displays an inference score syntactic tree where the effects of quantification can be observed, namely that of the sentence 'everyone ordered fries'. The β-reduction incorporating fries into the 'order' predicate has the effect of increasing the certainty in the fact that Ellen ate fries and decreasing certainty in the fact that John did *not* eat fries. Furthermore the certainty of John and Ellen eating pizza is also decreased marginally. These inferences are consistent with the world knowledge constraints and structure, and behave similarly to those in fig. 3.2.

The crucial aspect of this tree lies in the inference of the denotation of 'everyone'. Most of these inferences are direct reflections of the world knowledge constraints in the world, rather than interactions between the meaning of 'everyone' and the individual propositions. Notably, the inferences score of proposition 5 represents the unlikeliness of leaving somewhere without paying. Of special interest is the inference score of proposition 6. Intuitively, one would say that the meaning of 'everyone' would create a positive inference for both John and Ellen eating pizza. However, we can observe a neutral inference in this case. The behaviors observed follow from the definition of 'everyone' (eq. (2.19)). Firstly, the meaning of the set conjunction cannot be computed without some sort of binding of the λ-terms. Otherwise each set would contain free variables, which have no meaning. Hence the existential-closure operations must be performed first. The only way to do this is to distribute the existential quantifier over the different conjuncts. This results in $\exists P.\{\vec{v} \mid \vec{v} \models \text{P(john)}\} \wedge_{\times} \exists P.\{\vec{v} \mid \vec{v} \models \text{P(ellen)}\}$. Crucially, because the scope of the existential quantifier is no longer global, the λ-term (here P) no longer necessarily must have the same assignment over all conjuncts. The resulting meaning of 'everyone' is then, in natural language, 'everyone does something (but not necessarily the same thing)'. Which essentially is the effect of the set quantification taking scope over the $\exists$-closure.

With this in mind, the inferences make sense. The meaning of the word 'everyone' is trivial, as there are virtually no constraints on any model for it to entail the conditions of 'everyone'. Hence, the inferences represent world knowledge constraints that already exist *a priori* of the definition of quantification employed in $[\![everyone]\!]^{\mathcal{S}}$. Since there are no world knowledge constraints on John and Ellen both eating pizza, the inference score is centered around 0.

The last β-reduction that results in the sentence final meaning strengthening the inferences from $1-4$, whilst the inferences for propositions $5-6$ are flipped to a certain extent. This is because binding the λ-variable resolves the scope problem discussed above. This results in the slightly negative inference for proposition 6, meaning that it is unlikely that both John and Ellen eat a pizza given that they have both ordered fries (but not impossible). Secondly, the negative inference for proposition 5 is now neutral. This is because the meaning of 'everyone ordered fries' is not as trivial as the the bare meaning of $[\![everyone]\!]^{\mathcal{S}}$. The neutral inference score indicates that knowing that 'everyone ordered fries' has little bearing on whether john left without paying. While it is unlikely that someone orders something and leaves without paying it could be the case that Ellen paid for Johns order, after all they both ordered something. These two hypotheses cancel each other out and result in an inference score centered around 0. The remaining inferences in $1-4$ can already be observed in the preceding nodes. Just as in fig. 3.2, these are solidified upon receiving information about the agent.

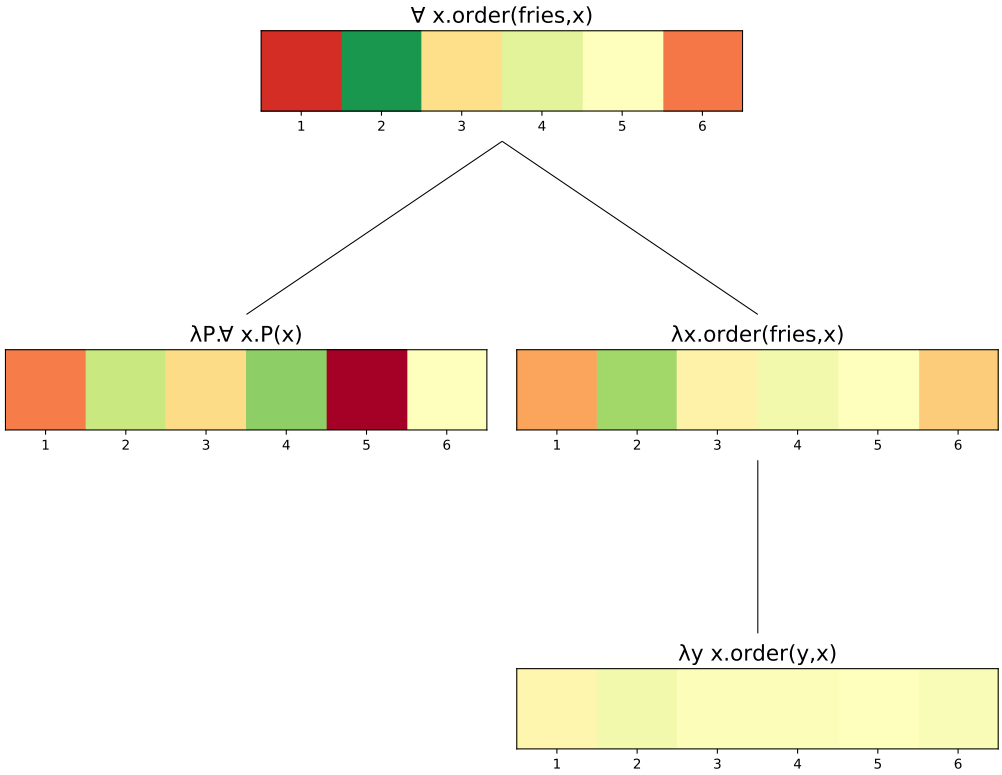


Figure 3.4: A visualization of the compositional process in λ-DFS of the sentence 'everyone ordered fries'. Each non-entity node contains a graph representing the inference score between the meaning up until that point (the centroid of that set) and a number of related full propositions (such that each box is $inf(meaning, proposition)$). From $1-6$ the relevant 6 propositions are $1:$ *¬eat(fries, john)*, $2:$ *eat(fries, ellen)*, $3:$ *enter(restaurant, john)*, $4:$ *enter(bar, john)*, $5:$ *leave(john)* $\wedge$ *¬pay(john)* and $6:$ *eat(pizza, ellen)* $\wedge$ *eat(pizza, john)*.

3.3 Comparison to Neural Network outputs

The mechanism that I have introduced allows us to map to real space at any point in the derivation. This is consistent with the proposal in Venhuizen et al. (2021) that subpropostional meanings in DFS are real valued vectors, whilst full propositional meanings are binary vectors.

It is also analogous to the RNN outputs, which outputs a real-valued estimate of the sentence meaning up until the current timestep and approach a binary vector as the sentence inputted to the network unfolds linearly. Figure 1.5 depicts these estimates in the form of a trajectory through 3D space.

Because the mechanism in λ-DFS defines a mapping to real space from sets of meaning vectors, we can plot a trajectory of the compositional process similar to fig. 1.5. Figure 3.5 depicts this trajectory for the sentence 'John entered the restaurant and ordered wine'. A crucial difference to the RNN outputs is that the compositional in λ-DFS process is not necessarily linear. Hence the order of operations of this trajectory is depicted depth-first, but does not represent the linear order in which an utterance would be produced or perceived.

The centroid of the denotation of the word 'and' lies somewhat in the center of the space. This is consistent with the proposed definition of conjunction in section 2.3, which is essentially any conjunction in the meaning space. Because this denotation is heavily underspecified, it is impartial to any sentence-final meaning in the dimensionality reduced space. This results in a vector roughly in the middle of the space. Incorporating the *et* predicate 'order wine' results in the composed meaning vector moving towards the full meaning of 'John ordered wine' and crucially away from 'John ordered beer' which is in line with the probabilistic constraint that ordering beer and wine are relatively probabilistically exclusive (in general it is good advice not to drink both wine and beer on the same night). Incorporating the second conjunct moves the centroid of the sentence meaning so far in the direction of the sentence-final meaning of 'John entered the restaurant'. Incorporating the entity John moves the meaning average close towards the sentence-final meaning of 'John entered the restaurant and ordered wine'. The result of the compositional process and the sentence-final meaning as computed by the vector algebra proposed in Venhuizen et al. (2021) are not exactly equal, because of the additional entailment relationships contained in the set representing 'John entered the restaurant and ordered wine'. For example, 'John entered the restaurant and drank wine' is also contained in this set.[2] The hard constraint that a person may not enter two places at the same time is also represented by the fact that the final position of the compositional mechanism is furthest away from the proposition 'John entered the bar'.

The mechanism of λ-DFS thus not only defines a trajectory through real space, but the intermediate steps are also sensible continuations of the preceding steps. Thus, although the mechanism employed by the RNN is of a vastly different nature, the two solutions to building sentence meanings incrementally share important properties.

[2]Although ordering does not imply drinking logically as per the hard constarints on the world (table 3.1), it can still be the case that in this meaning space this entailment holds due to the stochastic nature of the sampling algorithm.

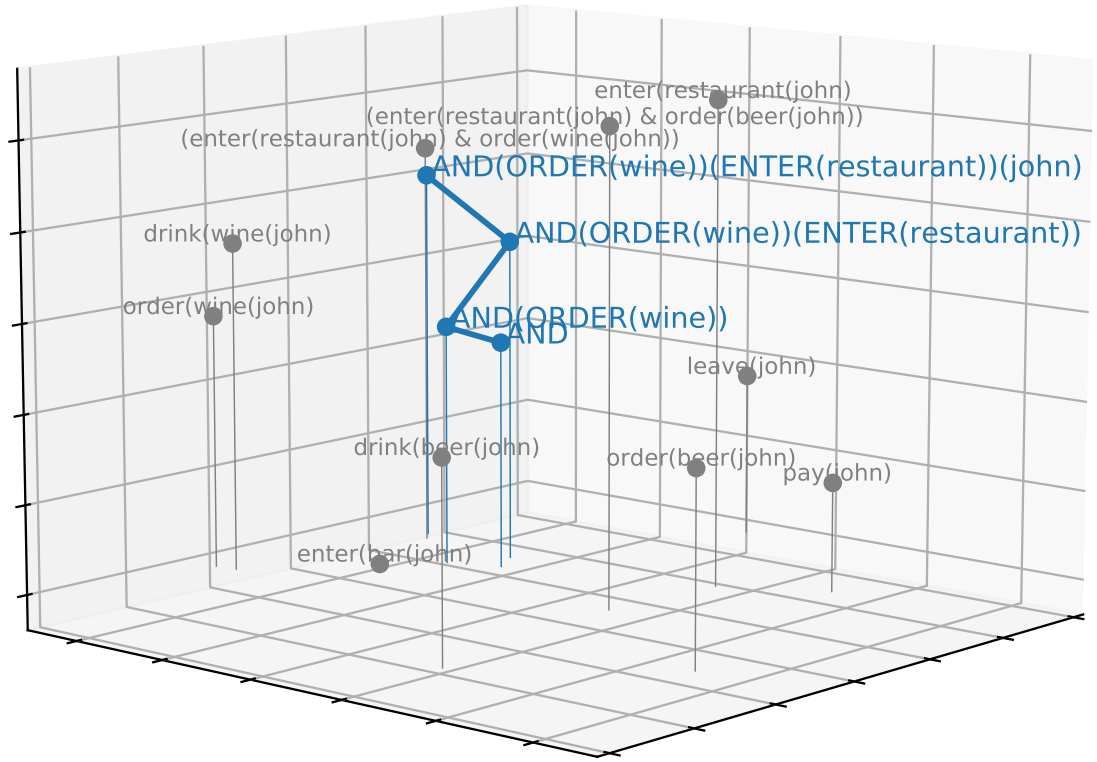


Figure 3.5: Trajectory of the compositional process through the dimensionality-reduced meaning space (using Multidimensional Scaling, described in more detail in Cox and Cox, 2008). The blue nodes are the set-averages of the given logical form. The coloured lines indicate the distance between respective points in the meaning space.

Chapter 4

Discussion

This section highlights some philosophical and semantic implications of λ-DFS and DFS as a whole. Furthermore I discuss some potential avenues for expanding λ-DFS to increase its coverage and theoretical rigor. First a reflection on sets as representational currency follows, then the relationship between set cardinality and entropy is briefly explored. I then shortly discuss the differences between compositional meaning construction and incremental meaning construction. Lastly, I shortly reflect on the data-driven approach to semantics employed in DFS and give a brief conclusion.

4.1 Are sets the right representational currency?

The decision to represent sub-propositional meanings as sets is an intuitive one, that is partially based on the observation that, in fig. 1.5, the neural network outputs at any timestep seem to lie in between their possible continuations. This implies that the meaning of a word or phrase is determined by its distribution in the meaning space. This is attractive for multiple reasons. Since DFS is a distributional framework, it only makes sense that sub-propositional meaning share this property. Secondly, sets already enjoy an important status in model-theoretic semantics and computation over them is relatively simple. The proposed mechanism uses the tools provided by typed λ-calculus to define β-reduction over sets, which crucially must accept sets as input and sets as output. The addition of types to the λ-expressions helps resolve quantification and allows us to locate the entities in a given meaning space.

Although the make-up of λ-expressions and functions depends on its type, and the exact contents of the sets that these expressions represent depend on their types (via existential closure), the representational structure of these sets is the same across types. For example, both an *et* and an *eet* predicate is a simplex set of meaning vectors even though their types dictate a strict difference in their logical make-up. As shown in this thesis, strongly typed sets are not necessary to achieve a decent coverage of natural language expressions given a meaning space. However, the 'weakly' typed implementation presented here has the effect that λ-expressions and sets operate on different levels of representations which are mediated via entailment and existential closure.

This begs the question whether it is possible to construe a mechanism that intertwines typed λ-calculus and set representation to a deeper extent. One possibility is to infer the set representation of complex types based on an intuitive definition of the basic types e and t and the most basic function *et*. Namely, type t expressions are meaning vectors, type e expressions are members of $D_e^{\mathcal{S}}$ as per eq. (1.1), and type *et* expressions are sets of meaning vectors exactly

has defined and used in this thesis. In more traditional type-theoretic notation, this comes down to ordinary functions that have a different assignment over the different models of the meaning space. Figure 4.1 explores this concept using a toy meaning space constructed for illustrative purposes only. An example *et* denotation in more traditional type-theoretic notation is given in eq. (4.1). Every element of this set is the assignment from entities to truth values of each model M_i in the example meaning space. Notice how the truth values coloured in red correspond exactly to the vector representing p_1 in the meaning space, this notation is thus equivalent to a set of all 'sing'-predicates (except notationally 'transposed').

$$\begin{array}{c} \\ M_1 \\ M_2 \\ M_3 \\ M_4 \end{array} \begin{array}{c} \begin{array}{ccccc} p_1 & p_2 & p_3 & p_4 & p_5 \end{array} \\ \begin{pmatrix} 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 \end{pmatrix} \end{array}$$

$D_e^{\mathcal{S}} = \{\text{Chet} = e_1, \text{Miles} = e_2, \text{Ella} = e_3, \text{trumpet} = e_4\}$
$D_t^{\mathcal{S}} = \{\vec{v} \mid \vec{v}_i \in \{0, 1\}\}$
$\mathcal{P} = \{\text{sing(Chet)}, \text{sing(Miles)}, \text{sing(Ella)}, \text{play(trumpet)(Chet)}, \text{play(trumpet)(Miles)}\}$

Figure 4.1: A toy meaning space with specification of the basic type domains.

$$[\![\text{sing}]\!] = \left\{ \begin{bmatrix} e_1 \to 1 \\ e_2 \to 0 \\ e_3 \to 1 \end{bmatrix}, \begin{bmatrix} e_1 \to 1 \\ e_2 \to 0 \\ e_3 \to 1 \end{bmatrix}, \begin{bmatrix} e_1 \to 0 \\ e_2 \to 0 \\ e_3 \to 1 \end{bmatrix}, \begin{bmatrix} e_1 \to 0 \\ e_2 \to 1 \\ e_3 \to 0 \end{bmatrix} \right\} \tag{4.1}$$

From these definitions we could infer the representation of more complex types and functions. For example an *eet* predicate is a set of assignments from entities to entities to truth values (eq. (4.2)).

$$[\![\text{play}]\!] = \left\{ \begin{bmatrix} e_4 \to \begin{bmatrix} e_1 \to 1 \\ e_2 \to 0 \end{bmatrix} \end{bmatrix}, \begin{bmatrix} e_4 \to \begin{bmatrix} e_1 \to 0 \\ e_2 \to 1 \end{bmatrix} \end{bmatrix}, \cdots \right\} \tag{4.2}$$

This definition is equivalent to a set of sets. More specifically, a set of sets of meaning vectors. The 'set of sets'-notation for this particular meaning space and this particular predicate is not so interesting given the simplicity of the space. But one could imagine a space in which entities may also play a saxophone in addition to a trumpet. Such a set would then be structured as follows.

$$\Big\{ \{\text{play(trumpet)(Chet)}, \text{play(trumpet)(Miles)}, \ldots\}, \{\text{play(saxophone)(Chet)}, \text{play(saxophone)(Miles)}, \ldots\} \Big\}$$

The structure of other complex types can be inferred in a similar fashion. This has the theoretical advantage of intertwining typed λ-calculus and set representations. Defining sub-propositional expressions in this fashion has considerable consequences for the mechanism proposed in the rest of this thesis and it is not exactly clear how some of the operations defined, like existential closure, would percolate to this extension and whether the coverage of natural language expressions would change significantly. However, this extension of λ-DFS seems

promising at first sight. Given the scope of this thesis it is not formalized further, but left as a potential avenue for research building on compositionality in DFS.

4.2 Entropy

One of the open issues of λ-DFS and its set-based implementation is relationship between entropy, information gain and probability theory defined in vanilla DFS. Venhuizen et al. (2021) succinctly describe the relationship between the neural network outputs in terms of entropy. Namely, the cosine similarity between two meaning vectors is proportional to their conditional entropy. Here conditional entropy means the logarithm of the probability of a word w given the utterance up until that point (see eq. (1.9)), which mathematically indicates the uncertainty one has over the possible continuation of an utterance. Since λ-DFS defines a trajectory through the meaning space just like the RNN-solution does, the question arises whether there is a similar relationship to conditional entropy between each β-reduction given the set denotations.

The RNN finds a statistical solution for mapping words to vectors based on a training set. The correspondence between the outputs and the conditional entropy of utterances makes sense, since RNN's essentially estimate a probability distribution over the outputs to begin with. Additionally one can easily estimate the probability of the raw training set if one has such data and compare it to the outputs. The compositional mechanism I have tried to develop has no such luxury, since there is no training data to help estimate conditional entropy. The only input to the system is some sort of syntactic structure and a meaning space.

There is one observation in the behavior of DFS that seems to be related to the notion of entropy. Namely, the development of set cardinality as an utterance reaches full reduction. Given how sets are constructed in λ-DFS, based on entailment, it follows that denotations whose existential closure are more trivially satisfiable in the world tend to have larger sets (there are more propositions that entail this definition). As the variables in the denotation are assigned to constants, the denotation becomes less general and more specific (consider 'someone orders something' versus 'someone orders beer') and thus the cardinality of the resulting sets tend to become smaller as the composition continues.

Exactly by how much the cardinality of the set decreases with each step depends on the world knowledge constraints on the meaning space and the specific utterance being derived. To illutrate, let us take an example meaning space in which it is very commonly defined that someone drinks beer and very uncommonly defined that someone drinks water (say, your average student fraternity). The *eet* predicate 'drink' may then look like as follows.

(8) $\text{drink} = \{\text{drink(water,ellen)}, \text{drink(beer,john)}, \text{drink(beer,mike)}, \text{drink(beer,emma)}\}$

In this case, composing 'drink' with either 'water' or 'beer' incurs different cardinality changes (namely $4 \rightarrow 1 = 3$ and $4 \rightarrow 3 = 1$) respectively. The lesser defined and therefore more unlikely utterence-final meaning involving drinking *water* incurs a higher change in set cardinality than drinking *beer*. This seems to be a good candidate for some sort of proportionality to information gain (or surprisal) as improbable events incur a high surprisal. However, one issue with this approach that it relies heavily on the exact formulation of the set of atomic propositions A. In a similar meaning space, the unlikelihood of drinking water for John, Mike and Emma could just as well be represented by meaning vectors which happen to have a very low probability. This would crucially not be represented in the set 'drink'.

Since sets by default do not provide the necessary machinery to capitalize on the probability of meaning vectors, we could try and back-off to the centroid of sets and see whether there is a correspondence between the conditional entropy of two centroids (over which Venhuizen et al. (2021) define surprisal, as they are just points in meaning space) and the change in set cardinality as a result of composition. However, the relationship between a value in the centroid and the number of elements in a set is not transparent. Consider for example the set a with two propositions which assign 0 and 1 to M_1 respectively and the set b with 100 propositions, the first half assigning 0 to M_1 and the second half assigning 1 to M_1. The centroids of both of these sets assign 0.5 to M_1, even though treating a as some composition of b results in a cardinality decrease of 98. This observation about centroids begs the question whether the relationship between real-valued vectors, entropy and processing difficulty holds at all for λ-DFS, which cannot base its outputs on a probability distribution over a training dataset (the equivalent of 'linguistic experience' in Venhuizen et al., 2021).

To investigate this issue further, I see a number of potential avenues. If one had some sort of corpus of logical forms, whose frequency occurrence is consistent with the world knowledge constraints of a given meaning space, then one could easily calculate the probability distribution over each word given its components and empirically compare them to the cardinalities of the sets produced by λ-DFS. Secondly, one could try to enrich the representation of sets with more probabilistic information so the proportionality follows even in the case of a fully specified set of atomic propositions (whose probabilities differ). Lastly, a potential addition to λ-DFS is to employ a richer mapping to real-space (something more sophisticated than the centroid) so that set cardinality is reflected in the real-valued equivalent. One thing we can conclude from this brief and shallow exploration is that trivially true denotations (low surprisal words) tend to have a high cardinality in their sets and that this is likely somehow related to the notion of surprisal.

4.3 Compositionality versus incrementality

It is now an appropriate time to ponder whether the proposed mechanism satisfied the definition of compositionality given in ex. 2, repeated here as ex. 9 for clarity. In general, this thesis has provided a function that defines a denotation for all well-formed instances of first-order formulas and all partial functions of well-formed first order logical formulas (λ-calculus). Furthermore, I have introduced a (somewhat non-trivial) function that produces an output given any pair of well-formed arguments. To what extent these outputs are of desirable semantic nature is a question that partially remains, although I have tried to make a reasonable case for this in chapter 3. The interpretability of these outputs is mediated through a mapping from denotations to sets, and a mapping from sets to real-valued vectors.

(9) There is a function **fa**() to the meaning of complex utterances from the individual meaning of its constituents and the way in which those meanings are combined.

One of the prerequisites for the compositional mechanism was that it produces an output pattern analogous to the RNN. Chapter 3 attempts to show this, but there are still relevant differences between these trajectories. One of the key differences relates to the syntax of utterances in λ-DFS and the RNN respectively. Throughout this thesis, the structure of sentences has been represented using binary trees which is standard in many compositional frameworks. The neural

network derives meaning from linearized structures, meaning that the utterance is presented incrementally. This has the effect that the trajectory defined by λ-DFS and by the RNN do not have the same ordering.

Incrementality has the added benefit of directly representing the way in which speakers perceive language, whereas compositionality based on binary trees has the added prerequisite of requiring speakers to induce a structure from a linearized utterance. However, it could still be true that the RNN finds a solution in which non-linear dependencies must be formed in order to produce the observed output. In fact, large scale neural language models are suspected to able to infer parts of natural language hierarchies from linearized data (Manning et al., 2020). This would imply that the RNN, perhaps also in the case of Venhuizen et al. (2021), could use structure to infer sub-propositional and full propositional meanings in DFS.

If this is the case, then the RNN and λ-DFS are not mutually incompatible. While λ-DFS is structure-dependent, it is also structure-agnostic. That is, as long as one provides a lexicon of elementary denotations and some ordering in which to combine them, λ-DFS will produce an output. What exactly this ordering is does not matter, it could be CFG-style binary trees or based on some sort of dependency grammar. Figuring out in what way structure plays a role in the RNN solution to DFS meanings and whether that is compatible with λ-DFS is related to the larger research topic of explainability in neural networks (e.g. Roscher et al., 2020), which is an interesting avenue of broader machine learning research from which DFS and computational semantics as a whole can benefit.

4.4 Data-driven semantics and conclusion

One of the aspects of modern research in semantics research this thesis highlights is the symbiosis between formal, symbolic methods and empirical data-driven methods. Language is not just a phenomenon of reference, it also a phenomenon of distribution. Meaning that some elements of linguistic meaning are shaped by usage and frequency, rather than logic. As illustrated in chapter 1, both of these aspects are desirable for a maximally encompassing theory of meaning.

DFS representations lend themselves well to this style of theory due to their distributed nature, while still being grounded in formal models. For example, probability theory is very intuitively defined over these distributed logical representations. Probability theory is powerful machinery in computational linguistics as a whole, and yet often not in the toolbox of traditional semanticists. However, DFS representations are still only sampled top-down, meaning from a world specification in terms of constraints, rather than bottom-up based on data. The latter could be approached in two different ways. One could try to infer the constraints of some world based on a data set, and sample normally with the inferred set of constraints. On the other hand one could try and infer the observational models directly and construct a DFS-space based on the inferred formal models. Doing so based on pure text is intuitively not an easy feat. This is especially so because so many observations about the world are not directly expressed through language, but rather implied through social and pragmatic rules. Consider for example natural language utterance describing a driver running a red light versus a driver running a green light. Despite the fact that the latter is much more frequent and in fact describes an important rule of a standard traffic model, it is very unlikely to be expressed as frequently than the former (if at all), because it is pragmatically or socially redundant to say so. Hence data for empirically

sampling DFS-spaces must be semantically enriched or annotated.

The machinery of λ-DFS proposed in this thesis hopes to take a step further in this symbiosis. It is essentially formal and symbolic machinery implemented over distributed empirical representations, and its effects can also be evaluated based on empirical probabilistic tools (such as inference scores). With something like λ-DFS, one only needs a lexicon and a meaning space to start deriving meanings of arbitrary complexity meaning units. This is in contrast to the neural network, which ideally needs curated data in order to begin finding a solution and careful tuning to make sure the found solution is somewhat appropriate and general. Moreover, systems like λ-DFS offer complete control over the mechanism and outputs, which can be easily modified and adjusted to ones needs. This allows DFS to become more practical for semanticists to use in their research on a larger scale, whatever its particular goal. Reconciling the empirical aspect of language with the formal and logical rigor of traditional semantics is an important step towards pursuing the common goal of inquiring truth and meaning in natural language.

Bibliography

Bakarov, A. (2018). A survey of word embeddings evaluation methods. *CoRR*, *abs/1801.09536*. http://arxiv.org/abs/1801.09536

Baroni, M., Bernardi, R., Do, N.-Q. & Shan, C.-c. (2012). Entailment above the word level in distributional semantics. *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, 23–32. https://www.aclweb.org/anthology/E12-1004

Baroni, M., Bernardi, R. & Zamparelli, R. (2014). Frege in space: A program for composition distributional semantics. *Linguistic Issues in Language Technology, Volume 9, 2014 - Perspectives on Semantic Representations for Textual Inference*. https://www.aclweb.org/anthology/2014.lilt-9.5

Bender, E. M. & Koller, A. (2020). Climbing towards NLU: On meaning, form, and understanding in the age of data. *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 5185–5198. https://doi.org/10.18653/v1/2020.acl-main.463

Brouwer, H., Delogu, F., Venhuizen, N. J. & Crocker, M. W. (2021). Neurobehavioral correlates of surprisal in language comprehension: A neurocomputational model. *Frontiers in Psychology*, *12*, 110. https://doi.org/10.3389/fpsyg.2021.615538

Champollion, L. (2011). Quantification and negation in event semantics. *Formal Semantics and Pragmatics*, *6*, 1–23.

Cox, M. A. A. & Cox, T. F. (2008). Multidimensional scaling. *Handbook of data visualization* (pp. 315–347). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-540-33037-0_14

Elman, J. L. (1990). Finding structure in time. *Cognitive Science*, *14*(2), 179–211. https://doi.org/10.1207/s15516709cog1402_1

Emerson, G. & Copestake, A. (2017). Semantic composition via probabilistic model theory. *IWCS 2017 - 12th International Conference on Computational Semantics - Long papers*. https://aclanthology.org/W17-6806

Firth, J. R. (1957). A synopsis of linguistic theory 1930-55. *1952-59*, 1–32.

Frank, S. L., Koppen, M., Noordman, L. G. & Vonk, W. (2003). Modeling knowledge-based inferences in story comprehension. *Cognitive Science*, *27*(6), 875–910. https://doi.org/10.1207/s15516709cog2706_3

Frege, G. (1892). Über sinn und bedeutung. *Zeitschrift für Philosophie Und Philosophische Kritik*, *100*(1), 25–50.

Goodman, N. D. & Lassiter, D. (2015). Probabilistic semantics and pragmatics uncertainty in language and thought. *The handbook of contemporary semantic theory* (pp. 655–686). John Wiley & Sons, Ltd. https://doi.org/10.1002/9781118882139.ch21

Heim, I. (1982). *The semantics of definite and indefinite noun phrases* (Doctoral dissertation). University of Massachusetts.

Herbelot, A. (2020). How to stop worrying about compositionality. *The Gradient*. https://thegradient.pub/how-to-stop-worrying-about-compositionality-2.

Kamp, H. & Partee, B. (2002). *Context-dependence in the analysis of linguistic meaning*. Brill. https://brill.com/view/title/23318

Kamp, H. & Reyle, U. (1993). *From discourse to logic*. Springer Netherlands. https://doi.org/10.1007/978-94-017-1616-1

Manning, C. D., Clark, K., Hewitt, J., Khandelwal, U. & Levy, O. (2020). Emergent linguistic structure in artificial neural networks trained by self-supervision. *Proceedings of the National Academy of Sciences*, *117*(48), 30046–30054. https://doi.org/10.1073/pnas.1907367117

Montague, R. (1973). The proper treatment of quantification in ordinary english. *Philosophy, language, and artificial intelligence* (pp. 141–162). Springer Netherlands. https://doi.org/10.1007/978-94-009-2727-8_7

Muskens, R. (1996). Combining montague semantics and discourse representation. *Linguistics and Philosophy*, *19*(2), 143–186. https://doi.org/10.1007/bf00635836

Parsons, T. (2017). The Traditional Square of Opposition. In E. N. Zalta (Ed.), *The Stanford encyclopedia of philosophy* (Summer 2017). Metaphysics Research Lab, Stanford University.

Partee, B. H. (2004). *Compositionality in formal semantics: Selected papers by barbara partee (explorations in semantics ser.)* Wiley-Blackwell.

Potts, C. (2018). A case for deep learning in semantics. *CoRR*, *abs/1809.03068*. http://arxiv.org/abs/1809.03068

Roscher, R., Bohn, B., Duarte, M. F. & Garcke, J. (2020). Explainable machine learning for scientific insights and discoveries. *IEEE Access*, *8*, 42200–42216. https://doi.org/10.1109/ACCESS.2020.2976199

Russell, B. (1940). *An inquiry into meaning and truth*. Routledge.

Spivey, M. (2006). *The continuity of mind*. Oxford University Press. https://doi.org/10.1093/acprof:oso/9780195170788.001.0001

Szabó, Z. (2012). The case for compositionality.

Turney, P. D. & Pantel, P. (2010). From frequency to meaning: Vector space models of semantics. *Journal of Artificial Intelligence Research*, *37*, 141–188. https://doi.org/10.1613/jair.2934

Venhuizen, N. J., Hendriks, P., Crocker, M. W. & Brouwer, H. (2021). Distributional formal semantics. *Information and Computation (submitted)*.

Venhuizen, N. J., Crocker, M. W. & Brouwer, H. (2018). Expectation-based comprehension: Modeling the interaction of world knowledge and linguistic experience. *Discourse Processes*, *56*(3), 229–255. https://doi.org/10.1080/0163853x.2018.1448677

Venhuizen, N. J., Crocker, M. W. & Brouwer, H. (2019). Semantic entropy in language comprehension. *Entropy*, *21*(12), 1159. https://doi.org/10.3390/e21121159

Venhuizen, N. J., Hendriks, P., Crocker, M. W. & Brouwer, H. (2019). A framework for distributional formal semantics. In R. Iemhoff, M. Moortgat & R. de Queiroz (Eds.), *Logic, language, information, and computation* (pp. 633–646). Springer Berlin Heidelberg.

Winter, Y. (2016). *Elements of formal semantics: An introduction to the mathematical theory of meaning in natural language*. Edinburgh University Press.

List of Figures

List of Tables