Context Update for Lambdas and Vectors

Reinhard Muskens and Mehrnoosh Sadrzadeh*

Department of Philosophy, Tilburg University
r.a.muskens@gmail.com
School of Electronic Engineering and Computer Science,
Queen Mary University of London
mehrnoosh.sadrzadeh@qmul.ac.uk

Abstract. Vector models of language are based on the contextual aspects of words and how they co-occur in text. Truth conditional models focus on the logical aspects of language, the denotations of phrases, and their compositional properties. In the latter approach the denotation of a sentence determines its truth conditions and can be taken to be a truth value, a set of possible worlds, a context change potential, or similar. In this short paper, we develop a vector semantics for language based on the simply typed lambda calculus. Our semantics uses techniques familiar from the truth conditional tradition and is based on a form of dynamic interpretation inspired by Heim's context updates.

Keywords: Vector semantics · Simply typed lambda calculus · Context update · Context change potential · Compositionality

1 Introduction

Vector semantic models, otherwise known as distributional models, are based on the contextual aspects of language, the company each word keeps, and patterns of use in corpora of documents. Truth conditional models focus on the logical and denotational aspects of language, sets of objects with certain properties and application and composition of functions. Vector semantics and truth conditional models are based on different philosophies; in recent years there has been much effort to bring them together under one umbrella, see for example [1–3, 8, 9].

In a recent abstract [14], we sketched an approach to semantics that assigned vector meanings to linguistic phrases using a simply typed lambda calculus in the tradition of [10]. Our previous system was guided by a truth conditional interpretation and provided vector semantics very similar to the approaches of [1–3, 8, 9]. The difference was that the starting points of these latter approaches are categorial logics such as Pregroup Grammars and Combinatorial Categorial Grammar (CCG). Our reasoning for the use of lambda calculus was that it directly relates our semantics to higher order logic and makes standard ways of treating long distance dependencies and coordination accessible to vector-based semantics. In this short account, we follow the same lines as in our

* Support by EPSRC for Career Acceleration Fellowship EP/J002607/1 is gratefully acknowledged by M. Sadrzadeh.

previous work. But whereas in previous work we worked with a static interpretation of distributions, here, we focus on a dynamic interpretation.

The lambda calculus approach we use is based on the Lambda Grammars of [11, 12], which were independently introduced as Abstract Categorial Grammars (ACGs) in [5]. The theory developed here, however, can be based on any syntax-semantics interface that works with a lambda calculus based semantics. Our approach is agnostic as to the choice of a syntactic theory. Lambda Grammars/ACGs are just a framework for thinking about type and term homomorphisms and we are using them entirely in semantics here. In a longer paper we will show in more detail how lambda logical forms (the abstract terms) can be obtained: 1) from standard linguistic trees with the help of a procedure that is essentially that of Heim and Kratzer [7]; 2) from LFG f-structures by means of a 'glue logic'; 3) from Lambek proofs by means of semantic recipes; 4) and from CCG derivations by means of using the combinators associated with CCG rules.

The dynamic interpretation we work with here is the "context change potential" of [6]. We believe other dynamic approaches, such the update semantics of [16] and the continuation-based semantics of [4], can also be used; we aim to do these in future.

2 Heim's Files and Distributional Contexts

Heim describes her contexts as files that have some kind of information written on (or in) them. Context changes are operations that update these files, e.g. by adding or deleting information from the files. Formally, a context is taken to be a set of sequence-world pairs, in which the sequences come from some domain $\mathcal{D}_I$ of individuals, as follows:

$$ctx \subseteq \{(g, w) \mid g\colon \mathbb{N} \to \mathcal{D}_I, w \text{ a possible world}\}$$

(We follow Heim [6] here in letting the sequences in her sequence-world-pairs be infinite, although they are best thought of as finite.)

Sentence meanings are *context change potentials* (CCPs) in Heim's work, functions from contexts to contexts. A sentence S comes provided with a sequence of instructions that, given any context ctx, updates its information so that a new context denoted as

$$ctx + S$$

results. The sequence of instructions that brings about this update is derived compositionally from the constituents of S.

In distributional semantics, contexts are words somehow related to each other via their patterns of use, e.g. by co-occurring in a neighbourhood word window of a fixed size or via a dependency relation. In practice, one builds a context matrix M over $\mathbb{R}^2$, with rows and columns labeled by words from a vocabulary Σ and with entries taking values from $\mathbb{R}$, for a full description see ([15]). M can be seen as the set of its vectors:

$$\{\overrightarrow{v} \mid \overrightarrow{v}\colon \Sigma \to \mathbb{R}\}$$

where each $\overrightarrow{v}$ is a row or column in M.

If we take Heim's domain of individuals $\mathcal{D}_I$ be the vocabulary of a distributional model of meaning, that is $\mathcal{D}_I := \Sigma$, then a context matrix can be seen as a so-called *quantized* version of a Heim context:

$$\{(\overrightarrow{g}, w) \mid \overrightarrow{g} : \Sigma \to \mathbb{R}, w \text{ a possible world}\}$$

Thus a distributional context matrix is obtainable by endowing Heim's contexts with $\mathbb{R}$. In other words, we are assuming that not only a file has a set of individuals, but also that these individuals take some kind of values, e.g. from reals.

The role of possible worlds in a distributional semantics is arguable, as vectors retrieved from a corpus are not naturally truth conditional. Keeping the possible worlds in the picture provides a machinery to assign a proposition to a distributional vector by other means and can become very useful. But for the rest of this abstract, we shall deprive ourselves from this advantage and only work with the following set as our context:

$$\{\overrightarrow{g} \mid \overrightarrow{g} : \Sigma \to \mathbb{R}, \overrightarrow{g} \in M\}$$

Distributional versions of Heim's CCP's can be defined based on the intuitions and definitions of Heim. In what follows we pan out how these instructions let contexts thread through vectorial semantics in a compositional manner.

3 Vectors, Matrices, Lambdas

Lambda Grammars of [11, 12] were independently introduced as Abstract Categorial Grammars (ACGs) in [5]. An ACG generates two languages, an *abstract* language and an *object* language. The abstract language will simply consist of all linear lambda terms (each lambda binder binds exactly one variable occurrence) over a given vocabulary typed with *abstract types*. The object language has its own vocabulary and its own types. It results from 1) specifying a *type homomorphism* from abstract types to object types and 2) specifying a *term homomorphism* from abstract terms to object terms. The term homomorphism must respect the type homomorphism. For more information about the procedure of obtaining an object language from an abstract language, see the papers mentioned or the explanation in [13].

Let the basic abstract types of our setting be D (for determiner phrases), S (for sentences), and N (for nominal phrases). Let the basic object types be I and R. The domain $\mathcal{D}_I$ corresponding to I can be thought of as a vocabulary, $\mathcal{D}_R$ models the set of reals $\mathbb{R}$. The usual operations on $\mathbb{R}$ can be defined using Tarski's axioms (in full models that satisfy these axioms $\mathcal{D}_R = \mathbb{R}$ will hold; in generalised models we get what boils down to a first-order approximation of $\mathbb{R}$). Objects of type $I \to R$ are abbreviated to IR; these are identified with *vectors* with a fixed basis.

We will associate simple words like names, nouns and verbs with vectors, i.e. with objects of type IR and will denote these with constants like $\overrightarrow{\mathsf{woman}}$, $\overrightarrow{\mathsf{smoke}}$, etc. The typed lambda calculus will be used to build certain functions with the help of these vectors that will then function as the meanings of those words. The meanings of content words will typically be functions that are completely given by some vector, but they will not (necessarily) be identified with vectors (see also Table 1 below).

a	τ	$H(a)$	$\rho(\tau)$
Anna	$(DS)S$	$\lambda Z.Z\overrightarrow{\mathsf{anna}}$	$(VU)U$
woman	N	$\lambda Z.Z\overrightarrow{\mathsf{woman}}$	$(VU)U$
tall	NN	$\lambda QZ.Q(\lambda vc.ZvF(\overrightarrow{\mathsf{tall}}, v, c))$	$((VU)U)(VU)U$
smokes	DS	$\lambda vc.G(\overrightarrow{\mathsf{smoke}}, v, c)$	VU
loves	DDS	$\lambda uvc.I(\overrightarrow{\mathsf{love}}, u, v, c)$	VVU
knows	SDS	$\lambda pvc.pJ(\overrightarrow{\mathsf{know}}, v, c)$	UVU
every	$N(DS)S$	$\lambda Q.Q$	$((VU)U)(VU)U$
who	$(DS)NN$	$\lambda Z'QZ.Q(\lambda vc.Zv(QZ'c))$	$(VU)((VU)U)(VU)U$
and	$(\overline{\alpha}S)(\overline{\alpha}S)(\overline{\alpha}S)$	$\lambda R'\lambda R\lambda\overline{X}\lambda c.R'\overline{X}(R\overline{X}c)$	$(\rho(\overline{\alpha})U)(\rho(\overline{\alpha})U)(\rho(\overline{\alpha})U)$

Table 1. Some abstract constants a typed with abstract types τ and their term homomorphic images $H(a)$ typed by $\rho(\tau)$ (where ρ is a type homomorphism, i.e. $\rho(AB) = \rho(A)\rho(B)$). Here Z is a variable of type VU, Q is of type $(VU)U$, v of type V, c of type M, and p and q are of type U. The functions F, G, I, and J are explained in the main text. In the schematic entry for and, we write $\rho(\overline{\alpha})$ for $\rho(\alpha_1)\cdots\rho(\alpha_n)$, if $\overline{\alpha} = \alpha_1\cdots\alpha_n$.

Sentences will be *context change potentials*. A context for us is a matrix, thus it has type I^2R. A sentence takes the type $(I^2R)(I^2R)$. We abbreviate IR as V, I^2R as M and the sentence type MM as U (for 'update'). Verbs take a vector for each of their arguments, plus an input context, and return a context as their output. For instance, an intransitive verb takes a vector for its subject plus a context and returns a modified context. Thus it takes type $VMM = VU$. A transitive verb takes a vector for its subject, a vector for its object and a context and returns a context. Thus it has type VVU. Nouns are essentially treated as vectors (V), but, since they must be made capable of dynamic behaviour, they are 'lifted' to the higher type $(VU)U$. Our dynamic type homomorphism ρ is defined by letting $\rho(N) = (VU)U$, $\rho(D) = V$ and $\rho(S) = U$. Some consequences of this definition can be found in Table 1.

4 Context Update for Lambda Binders

Object terms corresponding to a content word a may update a context matrix c with the information in $\overrightarrow{a}$ and the information in the vectors of arguments of a. The result is a new context matrix c', with different value entries.

$$\begin{pmatrix} m_{11} & \cdots & m_{1k} \\ m_{21} & \cdots & m_{2k} \\ \vdots & & \\ m_{n1} & \cdots & m_{nk} \end{pmatrix} + \overrightarrow{a}, u, v, \cdots = \begin{pmatrix} m'_{11} & \cdots & m'_{1k} \\ m'_{21} & \cdots & m'_{2k} \\ \vdots & & \\ m'_{n1} & \cdots & m'_{nk} \end{pmatrix}$$

An example of a set of elementary update instructions may be as follows.

- The function denoted by $\lambda vc.G(\overrightarrow{\mathsf{smoke}}, v, c)$ increases the value entry of m_{ij} of c, for i and j indices of smoke and its subject v.
- The function denoted by $\lambda uv.\lambda c.I(\overrightarrow{\mathsf{love}}, u, v, c)$ increases the value entries of m_{ij}, m_{jk}, and m_{ik} of c, for i, j, k indices of loves, its subject u and its object v.
- The function denoted by $\lambda vc.F(\overrightarrow{\mathsf{tall}}, v, c)$ increases the value entry of m_{ij} of c, for i and j indices of tall and its modified noun v. The entry for *tall* in Table 1 uses this function, but allows for further update of context.
- The function denoted by $\lambda vc.J(\overrightarrow{\mathsf{know}}, v, c)$ increases the value entry of m_{ij} of c, for i and j indices of know and its subject v. The updated matrix is made the input for further update (by the context change potential of the sentence that is known) in Table 1.

Logical words such as *every* and *and* are often treated as noise in distributional semantics and not included in the context matrix. We have partly followed this approach here by treating *every* as the identity function (the noun already has the required 'quantifier' type $(VU)U$). To see this, note that the entry for 'every', $\lambda Q.Q$, is the identity function; it takes a Q and then spits it out again. The alternative would be to have an entry along the lines of that of 'tall', but this would not make a lot of sense. It is the content words that seem to be important in a distributional setting, not the function words.

The word *and* does have a function here though—it is treated as a generalised form of function composition. The entry for the word in Table 1 is schematic, as *and* does not only conjoin sentences, but also other phrases of any category. So, the type of the abstract constant connected with the word is $(\overline{\alpha}S)(\overline{\alpha}S)(\overline{\alpha}S)$, in which $\overline{\alpha}$ can be any sequence of abstract types. Ignoring this generalisation for the moment, we obtain SSS as the abstract type for sentence conjunction, with a corresponding object type UUU, and meaning $\lambda pqc.p(qc)$, which is just function composition. This is defined in a way such that the context updated by *and*'s left argument will be further updated by its right argument. So 'Sally smokes and John eats bananas' will, given an initial matrix c, first update c to $G(\mathsf{Sally}, \mathsf{smoke}, c)$, which is a matrix, and then update this further with 'John eats bananas' to $I(\mathsf{eat}, \mathsf{John}, \mathsf{bananas}, G(\mathsf{smoke}, \mathsf{Sally}, c))$.

This treatment is easily extended to coordination in all categories. For example, the reader may check that $\mathsf{and\ admires\ loves}$ (which corresponds to *loves and admires*) has $\lambda uvc.I(\overrightarrow{\mathsf{admire}}, u, v, I(\overrightarrow{\mathsf{love}}, u, v, c))$ as its homomorphic image.

The update instructions fall through the semantics of phrases and sentences compositionally. The sentence *every tall woman smokes*, for example, will be associated with the following lambda expression:

$$(\mathsf{every\ tall\ woman})\lambda\zeta.(\mathsf{smokes}\ \zeta)$$

This in its turn has a term homomorphic image that is β-equivalent with the following:

$$\lambda c.G\left(\overrightarrow{\mathsf{smoke}}, \overrightarrow{\mathsf{woman}}, F(\overrightarrow{\mathsf{tall}}, \overrightarrow{\mathsf{woman}}, c)\right)$$

which describes a distributional context update for it. This term describes a first update of the context c according to the rule for the constant tall, and then a second update according to the rule for the constant smokes. As a result of these, the value entries

at the crossings of $\langle$tall, woman$\rangle$ and $\langle$woman, smokes$\rangle$ get increased. Much longer chains of context updates can be 'threaded' in this way.

In the following we give some examples. In each case the a. sentence is followed by an abstract term in b. which captures its syntactic structure. The update potential that follows in c. is the homomorphic image of this abstract term.

(1) a. Sue loves and admires a stockbroker
b. (a stockbroker)$\lambda\xi$.Sue(and admires loves ξ)
c. $\lambda c.I(\overrightarrow{\text{admire}}, \overrightarrow{\text{stockbroker}}, \overrightarrow{\text{sue}}, I(\overrightarrow{\text{love}}, \overrightarrow{\text{stockbroker}}, \overrightarrow{\text{sue}}, c))$

(2) a. Bill admires but Anna despises every cop
b. (every cop)$\lambda\xi$.and(Anna(despise ξ))(Bill(admire ξ))
c. $\lambda c.I(\overrightarrow{\text{despise}}, \overrightarrow{\text{cop}}, \overrightarrow{\text{anna}}, I(\overrightarrow{\text{admire}}, \overrightarrow{\text{cop}}, \overrightarrow{\text{bill}}, c))$

(3) a. The witch who Bill claims Anna saw disappeared
b. the(who($\lambda\xi$.Bill(claims(Anna(saw ξ))))witch)disappears
c. $\lambda c.G(\overrightarrow{\text{disappear}}, \overrightarrow{\text{witch}}, I(\overrightarrow{\text{see}}, \overrightarrow{\text{witch}}, \overrightarrow{\text{anna}}, J(\overrightarrow{\text{claim}}, \overrightarrow{\text{bill}}, c)))$

5 Conclusion and Future Directions

In previous work, we showed how a static interpretation of the lambdas will provide vectors for phrases and sentences of language. There, the object type of the vector of a word depended on its abstract type and could be an atomic vector, a matrix, or a cube, or a tensor of higher rank. Means of combinations thereof then varied based on the tensor rank of the type of each word. For instance one could take the matrix multiplication of the matrix of an intransitive verb with the vector of its subject, whereas for a transitive verb the sequence of operations were a contraction between the cube of the verb and the vector of its object followed by a matrix multiplication between the resulting matrix and the vector of the subject. A toolkit of functions needed to perform these operations was defined in previous work. That toolkit can be restated here for the type I^2R, rather than the previous IR, to provide means of combining matrices and their updates, if needed.

In this work, we show how a dynamic interpretation of the lambdas will also provide vectors for phrases and sentences of language. Truth conditional and vector models of language follow two very different philosophies. The vector models are based on contexts, the truth models on denotations. The dynamic interpretations of language, e.g. the approach of Heim, are also based on context update, hence these seem a more appropriate choice. In this paper, we showed how Heim's files can be turned into vector contexts and how her context change potentials can be used to provide vector interpretations for phrases and sentences. Our context update instructions were defined such that they would let contexts thread through vector semantics in a compositional manner.

Amongst the things that remain to be done in a long paper is to develop a vector semantics for the lambda terms obtained via other syntactic models, e.g. CCG, LFG, and Lambek Grammars, as listed at the end of the introduction section. We also aim to work with other update semantics, such as continuation-based approaches. One could also have a general formalisation wherein both the static approach of previous work and

the dynamic one of this work cohabit. This can be done by working out a second pair of type-term homomorphisms that will also work with Heim's possible world part of the contexts. In this setting, the two concepts of meaning: truth theoretic and contextual, each with its own uses and possibilities, can work in tandem.

Acknowledgements We wish to thank the anonymous referees for excellent feedback.

References

1. Baroni, M., Bernardi, R., Zamparelli, R.: Frege in space: A program for compositional distributional semantics. Linguistic Issues in Language Technology 9, 5–110 (2014)
2. Coecke, B., Sadrzadeh, M., Clark, S.: Mathematical Foundations for Distributed Compositional Model of Meaning. Lambek Festschrift. Linguistic Analysis 36, 345–384 (2010)
3. Grefenstette, E., Sadrzadeh, M.: Concrete models and empirical evaluations for the categorical compositional distributional model of meaning. Computational Linguistics 41, 71–118 (2015)
4. de Groote, P.: Towards a Montagovian Account of Dynamics. In: Proceedings of the 16th Semantics and Linguistic Theory Conference (SALT 16). pp. 1–16 (2006)
5. de Groote, P.: Towards Abstract Categorial Grammars. In: Association for Computational Linguistics, 39th Annual Meeting and 10th Conference of the European Chapter, Proceedings of the Conference. pp. 148–155. ACL, Toulouse, France (2001)
6. Heim, I.: On the projection problem for presuppositions. In: Portner, P., Partee, B.H. (eds.) Formal Semantics - the Essential Readings, pp. 249–260. Blackwell (1983)
7. Heim, I., Kratzer, A.: Semantics in generative grammar. Blackwell textbooks in linguistics, Blackwell publishers, Cambridge (Mass.), Oxford (1998)
8. Krishnamurthy, J., Mitchell, T.M.: Vector space semantic parsing: A framework for compositional vector space models. In: Proceedings of the 2013 ACL Workshop on Continuous Vector Space Models and their Compositionality (2013)
9. Maillard, J., Clark, S., Grefenstette, E.: A type-driven tensor-based semantics for CCG. In: Proceedings of the EACL 2014 Type Theory and Natural Language Semantics Workshop (2014)
10. Montague, R.: The Proper Treatment of Quantification in Ordinary English. In: Thomason, R. (ed.) Formal Philosophy. Selected Papers of Richard Montague, pp. 247–270. Yale University Press (1974)
11. Muskens, R.A.: Categorial Grammar and Lexical-Functional Grammar. In: Butt, M., King, T.H. (eds.) Proceedings of the LFG01 Conference, University of Hong Kong. pp. 259–279. CSLI Publications, Stanford CA (2001), tinyurl.com/jrc3nnw
12. Muskens, R.A.: Language, Lambdas, and Logic. In: Kruijff, G.J., Oehrle, R. (eds.) Resource Sensitivity in Binding and Anaphora, pp. 23–54. Studies in Linguistics and Philosophy, Kluwer (2003)
13. Muskens, R.: New Directions in Type-Theoretic Grammars. Journal of Logic, Language and Information 19(2), 129–136 (2010)
14. Muskens, R., Sadrzadeh, M.: Lambdas and vectors. Workshop on Distributional Semantics and Linguistic Theory (DSALT), 28th European Summer School in Logic, Language and Information (ESSLLI) (August 2016), free University of Bozen-Bolzano
15. Rubenstein, H., Goodenough, J.: Contextual Correlates of Synonymy. Communications of the ACM 8(10), 627–633 (1965)
16. Veltman, F.: Defaults in update semantics. Journal of Philosophical Logic 25(3), 221–261 (1996)