Principles of Natural Language, Logic, and Tensor Semantics

Mehrnoosh Sadrzadeh

School of Electronic Engineering and Computer Science, Queen Mary University of London, UK
mehrnoosh.sadrzadeh@qmul.ac.uk

Abstract

Residuated monoids model the structure of sentences. Vectors provide meaning representations for words. A functorial mapping between the two is obtained by lifting the vectors to tensors. The resulting sentence representations solve similarity, disambiguation and entailment tasks.

2012 ACM Subject Classification Theory of computation → Categorical semantics; Computing methodologies → Natural language processing; General and reference → Experimentation

Keywords and phrases Residuated Monoids, Vector Space Semantics, Corpora of Textual Data, Sentence Similarity and Disambiguation

Digital Object Identifier 10.4230/LIPIcs.CALCO.2019.3

Category Invited Paper

Funding Royal Academy of Engineering Industrial Fellowship Scheme $IFS1718\backslash 63$, Royal Society International Exchange Award $IE161631$

1 The Algebra of Grammatical Types

A partially ordered monoid is called *residuated* and is denoted by $(M, \cdot, 1, \leq, \to, \leftarrow)$, whenever for $b, c \in M$ we have $c \cdot c \to b \leq b$ and $b \leftarrow c \cdot c \leq b$. Given a set of basic types $\mathcal{B}$ and a vocabulary Σ, a *monoid grammar* is the tuple $(\mathcal{T}(\mathcal{B}), \Sigma, \mathcal{D}, \{s\})$, wherein $\mathcal{T}(\mathcal{B})$ is a residuated monoid generated over $\mathcal{B}$ and $\mathcal{D} \subseteq \Sigma \times \mathcal{T}(\mathcal{B})$ is a type assignment to the vocabulary. A string of words $w_1 w_2 \cdots w_n$ is *grammatical* in a monoid grammar, whenever for $(w_i, t_i) \in \mathcal{D}$, we have $t_1 \cdot t_2 \cdot \cdots \cdot t_n \leq s$, where s is an element of $\mathcal{B}$ and stands for the type of a sentence.

As an example, consider the vocabulary $\Sigma = \{\text{men}, \text{dogs}, \text{cute}, \text{kill}\}$ and the type dictionary $\mathcal{D} = \{(\text{men}, n), (\text{dogs}, n), (\text{cute}, n \leftarrow n), (\text{kill}, (n \to s) \leftarrow n)\}$. The sentence "men kill cute dogs" is grammatical, since we have

$$n \cdot ((n \to s) \leftarrow n) \cdot (n \leftarrow n) \cdot n \leq n \cdot ((n \to s) \leftarrow n) \cdot n \leq n \cdot (n \to s) \leq s$$

2 Tensor Semantics

Suppose W is a vector space with a set of fixed orthonormal basis $\{b_i\}_i$. Elements of W are vectors $\sum_i c_i b_i$ and elements of $\underbrace{W \otimes \cdots \otimes W}_{n}$ are tensors $T_{i_1 i_2 \cdots i_n} = \sum_{i_1 i_2 \cdots i_n} C_{i_1 i_2 \cdots i_n} b_{i_1} \otimes b_{i_2} \otimes \cdots \otimes b_{i_n}$. The action of a tensor on another tensor is called *tensor contraction* and is defined as $T_{i_1 i_2 \cdots i_n} T_{i_n i_{n+1} \cdots i_{n+k}} = T_{i_1 i_2 \cdots i_{n+1} \cdots i_{n+k}} \in \underbrace{W \otimes \cdots \otimes W}_{n+k-1}$.

We develop a mapping $\mathcal{F}$ between a monoid grammar and the tensor powers of W. To basic types $t \in \mathcal{B}$, we assign W, i.e., $\mathcal{F}(t) := W$; to words w with basic types t we assign elements of W, i.e., $\mathcal{F}(w) := T_i \in W$. To complex types, we assign tensors of W as follows

$$\mathcal{F}(t_1 \cdot t_2) = \mathcal{F}(t_1 \to t_2) = \mathcal{F}(t_1 \leftarrow t_2) := \mathcal{F}(t_1) \otimes \mathcal{F}(t_2)$$

8th Conference on Algebra and Coalgebra in Computer Science (CALCO 2019).
Editors: Markus Roggenbach and Ana Sokolova; Article No. 3; pp. 3:1–3:4
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Words with complex types are assigned elements of the tensor spaces of their types, that is, $\mathcal{F}(w) = T_{i_1 i_2 \cdots i_n} \in \underbrace{W \otimes \cdots \otimes W}_{n}$. Given a grammatical sentence $w_1 w_2 \cdots w_n$, its tensor meaning is defined as the tensor contraction of the tensor semantics of its words, that is, $\mathcal{F}(w_! w_2 \cdots w_n) := \mathcal{F}(w_1)\mathcal{F}(w_2) \cdots \mathcal{F}(w_n)$.

As an example, suppose we assign vectors T_k^{dogs} and T_j^{men} in W to *men* and *dogs*, the matrix T_{lk}^{cute} in $W \otimes W$ to *cute* and the cube T_{ijk}^{kill} in $W \otimes W \otimes W$ to *kill*. The meaning of "men kill cute dogs" is computed via the following contraction of tensors $T_j^{\text{men}} T_{ijl}^{\text{kill}} T_{lk}^{\text{cute}} T_k^{\text{dog}}$. Recall that when we have a fixed set of orthonormal basis, $T_{ij} \cong T_{ji}$.

3 Implementation on Corpora of Textual Data

Given a corpus of text, e.g. the English Wikipedia, a set of target words T and a set of context words C, a vector space W is created over C. In this vector space, each target word has a vector, where each c_i is (a function of) the number of times w occurred with each basis vector in a neighbourhood window, e.g. 5 words to the left or right. As an example, suppose $C = \{\text{blood, grave, dead}\}$ and $T = \{\text{vampire, zombie, butterfly}\}$ and the following vectors

$$T_i^{\text{zombie}} = (17, 13, 10) \quad T_i^{\text{vampire}} = (15, 9, 8) \quad T_i^{\text{butterfly}} = (0, 1, 3)$$

Words that have complex types are modelled as tensors. The tensors are learnt by first building vector representations for phrases containing the words, then *learning* a tensor whose contraction with the tensors of other words in the phrase provides a reasonable approximation for the vector of the phrase. For example, in order to learn a matrix for the adjective *green*, we first build vectors for all the adjective-noun phrases with *green* as adjective, e.g. for *green grass*, *green dress*, *green space*. Machine learning algorithms such as *least squared distance* are employed to learn an approximation for T_{ij}^{green} such that

$$T_i^{\text{green grass}} \sim T_{ij}^{\text{green}} T_j^{\text{grass}} \quad T_i^{\text{green dress}} \sim T_{ij}^{\text{green}} T_j^{\text{dress}} \quad T_i^{\text{green space}} \sim T_{ij}^{\text{green}} T_j^{\text{space}}$$

Once the grammatical structure of a language is modelled in a monoid grammar and word vectors and tensors have been built for its vocabulary, tensor contraction is applied to obtain vector representations for its sentences. The cosine distances between these representations provide a measure of sentence similarity and are applied to paraphrasing and disambiguation tasks. For paraphrasing, one builds vectors for sentences such as "man shut doors", "gentleman closed eyes", "programme faces difficulty", "project hits problem" and uses their distances to decide that the latter two are more similar than the former two. For disambiguation, one builds vectors for sentences such as "man drew sword", "man sketched sword", "man pulled sword" to decide whether *drew* means *sketched* or *pulled*.

4 History and References

Similar to programming languages, natural languages have different characteristic features such as morphology, phenology, syntax, semantics, and pragmatics. Formal structures have been used to study these features and indeed ideas are shared between natural and programming semantics communities. An example is the setting of Context Free Grammars, introduced by Chomsky to analyse the grammatical structure of English [4] and subsequently applied to other languages and programming languages. The first algebraic approaches to natural language go back to the work of Ajdukiewicz [1], where structures similar to groups were used to provide a functional interpretation for grammatical types. These systems were later refined with a noncommutative multiplication by Bar-Hillel [2] and then Lambek developed a residuated monoid semantics and a cut-free sequent calculus for them [15]. The expressive powers of these two systems were proven equivalent by Pentus [20].

The vector space semantics of natural language is motivated by the *distributional* semantic ideas of Firth [8] and Harris [12], who argued that words that occur in the same contexts have similar meanings. These models were both implemented in Information Retrieval [27] and applied to Natural Language Processing [24].

Encoding a model of grammar in vector space semantics to obtain vector representations for sentences was an open problem until recently. In 2007 Clark and Pulman showed how a context free parse tree of a sentence can be assigned a tensor semantics by taking the Kronecker products of the vectors of the words therein and the symbolic vectors of their grammatical roles [5]. It was not clear, however, how to build vectors for grammatical roles. Between 2008 and 2011, with Clark, Coecke, and Preller we showed that if one uses Lambek's pregroup grammars [16, 23] one obtains a functorial semantics in the compact closed category of finite dimensional vector spaces and linear maps [6, 22]. Later with Coecke and Grefenstette, we showed how residuated monoid grammars also get a functorial semantics via the translation between a residuated monoid and a pregroup [7]. More recently, I showed how one can get by without using category theory and still be able to express this semantics using the language of tensor contraction [25]; this is via the $\mathcal{F}$ mapping that I have tried to spell out in this abstract.

Starting from 2011, we have implemented and experimented with the tensor models on large corpora of textual data in similarity, disambiguation, and entailment tasks and showed that in each case there is a tensor model that outperforms the vector models[10, 11, 13, 19, 14, 26]. The method that we have described here and which is used to learn the tensors was introduced by Baroni and Zamparelli for adjectives [3] and later extended to verbs [9, 21]. Maillard and Clark [17] showed how one can use neural networks and the Skipgram algorithm of Mikolov [18] to obtain much better results. In work in progress with Clark and Wijnholds, we are extending these models to arbitrary tensors.

References

1 K. Ajdukiewicz. Die syntaktische Konnexitat. *Studia Philosophica*, 1:1–27, 1935.

2 Y. Bar-Hillel. A quasi-arithmetical notation for syntactic description. *Language*, 29:47–58, 1953.

3 M. Baroni and R. Zamparelli. Nouns are vectors, adjectives are matrices: Representing adjective-noun constructions in semantic space. In *Conference on Empirical Methods in Natural Language Processing*, Cambridge, MA, 2010.

4 Noam Chomsky. Three models for the description of language. *IRE Transactions on Information Theory*, 2:113–124, 1956.

5 Stephen Clark and Stephen Pulman. Combining Symbolic and Distributional Models of Meaning. In *Proceedings of the AAAI Spring Symposium on Quantum Interaction*, pages 52–55, 2007.

6 B. Coecke, M. Sadrzadeh, and S. Clark. Mathematical Foundations for Distributed Compositional Model of Meaning. Lambek Festschrift. *Linguistic Analysis*, 36:345–384, 2010.

7 Bob Coecke, Edward Grefenstette, and Mehrnoosh Sadrzadeh. Lambek vs. Lambek: Functorial vector space semantics and string diagrams for Lambek calculus. *Annals of Pure and Applied Logic*, 164(11):1079–1100, 2013. Special issue on Seventh Workshop on Games for Logic and Programming Languages.

8 J.R. Firth. A Synopsis of Linguistic Theory 1930–1955. In *Studies in Linguistic Analysis*. Longmans, 1957.

9 E. Grefenstette, G. Dinu, Y. Zhang, M. Sadrzadeh, and M. Baroni. Multi-Step Regression Learning for Compositional Distributional Semantics. In *10th International Conference on Computational Semantics*, Postdam, 2013.

10 E. Grefenstette and M. Sadrzadeh. Experimental Support for a Categorical Compositional Distributional Model of Meaning. In *Proceedings of Conference on Empirical Methods in Natural Language Processing*, pages 1394–1404, 2011.

11 Edward Grefenstette and Mehrnoosh Sadrzadeh. Concrete Models and Empirical Evaluations for the Categorical Compositional Distributional Model of Meaning. *Computational Linguistics*, 41:71–118, 2015.

12 Z.S. Harris. Distributional structure. *Word*, 1954.

13 D. Kartsaklis and M. Sadrzadeh. Prior Disambiguation of Word Tensors for Constructing Sentence Vectors. In *Proceedings of Conference on Empirical Methods in Natural Language Processing*, 2013.

14 Dimitri Kartsaklis, Nal Kalchbrenner, and Mehrnoosh Sadrzadeh. Resolving Lexical Ambiguity in Tensor Regression Models of Meaning. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, Baltimore, MD, USA, Volume 2: Short Papers*, pages 212–217, 2014.

15 J. Lambek. The mathematics of sentence structure. *American Mathematics Monthly*, 65, 1958.

16 J. Lambek. Type grammars revisited. In *proceedings of LACL 97*, volume 1582 of *Lecture Notes in Artificial Intelligence*. Springer Verlag, 1997.

17 Jean Maillard and Stephen Clark. Learning adjective meanings with a tensor-based skip-gram model. In *Proceedings of the Nineteenth Conference on Computational Natural Language Learning*, pages 327–331, 2015.

18 Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

19 Dmitrijs Milajevs, Dimitri Kartsaklis, Mehrnoosh Sadrzadeh, and Matthew Purver. Evaluating Neural Word Representations in Tensor-Based Compositional Settings. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 708–719. Association for Computational Linguistics, 2014.

20 Mati Pentus. Lambek Grammars Are Context Free. In *In Proceedings of the Eighth Annual IEEE Symposium on Logic in Computer Science*, pages 429–433. IEEE Computer Society Press, 1993.

21 Tamara Polajnar, Luana Fagarasan, and Stephen Clark. Reducing dimensions of tensors in type-driven distributional semantics. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1036–1046, 2014.

22 A. Preller and M. Sadrzadeh. Bell States and Negative Sentences in the Distributed Model of Meaning. In P. Selinger B. Coecke, P. Panangaden, editor, *Electronic Notes in Theoretical Computer Science, Proceedings of the 6th Workshop on Quantum Physics and Logic*, volume 270, pages 141–153, 2010.

23 Anne Preller and Joachim Lambek. Free compact 2-categories. *Mathematical Structures in Computer Science*, 17:309–340, 2007.

24 H. Rubenstein and J.B. Goodenough. Contextual Correlates of Synonymy. *Communications of the ACM*, 8(10):627–633, 1965.

25 M. Sadrzadeh. Unifying the Mathematics of Natural Language Grammar and Data. *London Mathematical Society News Letter*, pages 25–31, 2018.

26 Mehrnoosh Sadrzadeh, Dimitri Kartsaklis, and Esma Balkır. Sentence entailment in compositional distributional semantics. *Annals of Mathematics and Artificial Intelligence*, 82(4):189–218, 2018.

27 G. Salton, A. Wong, and C. S. Yang. A Vector Space Model for Automatic Indexing. *Commun. ACM*, 18:613–620, 1975.