Algebraic Approaches to Compositional Distributional Semantics

Daoud Clarke
University of Hertfordshire
daoud@metrica.net

David Weir
University of Sussex
davidw@sussex.ac.uk

Rudi Lutz
University of Sussex
rudil@sussex.ac.uk

Abstract

The question of how to compose meaning in distributional representations of meaning has recently been recognised as a central issue in computational linguistics. In this paper we describe three general and powerful tools that can be used to describe composition in distributional semantics: quotient algebras, learning of finite dimensional algebras, and the construction of algebras from semigroups.

1 Introduction

Vector based representations of meaning have wide application in natural language processing. While these techniques work well at the word level, for longer strings, data becomes extremely sparse. The question of how the principle of compositionality might apply for such representations has thus been recognised as an important one (Widdows, 2008; Clark et al., 2008).

Context-theoretic semantics (Clarke, 2007) is a framework for composing meanings in vector based semantics, in which the composition of the meaning of strings is described by a multiplication on a real vector space $\mathcal{A}$ that is bilinear with respect to the addition of the vector space, i.e.

$$x(y+z) = xy + xz \quad (x+y)z = xz + yz \quad (\alpha x)(\beta y) = \alpha\beta xy$$

where $x, y, z \in \mathcal{A}$ and $\alpha, \beta \in \mathbb{R}$. It is assumed that the multiplication is associative, but *not* commutative. The resulting structure is an associative algebra over a field — or simply an algebra when there is no ambiguity. Clarke (2007) gives a mathematical model of meaning as context, and shows that under this model, the meaning of natural language expressions can be described by an algebra. The framework is also applied to models of textual entailment, and logical and ontological representations of natural language meaning.

In this paper, we identify three general techniques for constructing algebras.

- Using quotient algebras to impose relations on a free algebra, as described in (Clarke et al., 2010).
- Defining finite-dimensional algebras using matrices. Any finite-dimensional algebra can be described in this way; we have investigated the possibility of learning such algebras using least squares regression.
- Constructing algebras from a semigroup to give it vector space properties. We sketch a possible method of using this technique, identified by Clarke (2007), to endow logical semantics with a vector space nature.

This paper presents a preliminary consideration of these general techniques, and our goal is simply to show that they are worthy of further exploration.

	apple	*big apple*	*red apple*	*city*	*big city*	*red city*	*book*	*big book*	*red book*
apple	1.0	0.26	0.24	0.52	0.13	0.12	0.33	0.086	0.080
big apple		1.0	0.33	0.13	0.52	0.17	0.086	0.33	0.11
red apple			1.0	0.12	0.17	0.52	0.080	0.11	0.33
city				1.0	0.26	0.24	0.0	0.0	0.0
big city					1.0	0.33	0.0	0.0	0.0
red city						1.0	0.0	0.0	0.0
book							1.0	0.26	0.24
big book								1.0	0.33
red book									1.0

Figure 1: Cosine similarity values between phrases

see red apple
buy apple
read big book
throw old small red book
buy large new book
see big city
visit big apple
modernise city
see modern city

Figure 2: The corpus used to compute the vectors that formed the generating set for the ideal.

2 Quotient Algebras

One commonly used bilinear multiplication operator on vector spaces is the tensor product (denoted $\otimes$), whose use as a method of combining meaning was first proposed by Smolensky (1990), and has been considered more recently by Clark and Pulman (2007) and Widdows (2008), who also looked at the direct sum (which Widdows calls the direct product, denoted $\oplus$).

The tensor algebra on a vector space V (where V is a space of context features) is defined as:

$$T(V) = \mathbb{R} \oplus V \oplus (V \otimes V) \oplus (V \otimes V \otimes V) \oplus \cdots$$

Any element of $T(V)$ can be described as a sum of components with each in a different tensor power of V. Multiplication is defined as the tensor product on these components, and extended linearly to the whole of $T(V)$.

Previous work has not made full use of the tensor product space; only tensor products are used, not sums of tensor products, giving us the equivalent of the product states of quantum mechanics. Our approach imposes relations on the vectors of the tensor product space that causes some product states to become equivalent to entangled states, containing sums of tensor products of different degrees. This allows strings of different lengths to share components. We achieve this by constructing a quotient algebra.

An ideal I of an algebra A is a sub-vector space of A such that $xa \in I$ and $ax \in I$ for all $a \in A$ and all $x \in I$. An ideal introduces a congruence $\equiv$ on A defined by $x \equiv y$ if and only if $x - y \in I$. For any set of elements $\Lambda \subseteq A$ there is a unique minimal ideal I_Λ containing all elements of Λ; this is called the ideal generated by Λ. The quotient algebra A/I is the set of all equivalence classes defined by this congruence. Multiplication is defined on A/I by the multiplication on A, since $\equiv$ is a congruence.

Elements that are congruent with respect to the ideal have equivalence classes that are equal in the quotient algebra. The construction is thus a way of imposing relations between vector elements: we simply choose a set of pairs that we wish to be equal, and put their difference in the generating set Λ.

Clarke et al. (2010), showed how an inner product can be computed for elements of the quotient algebra by taking the quotient of a finite dimensional subspace of the ideal and how a treebank could be used to identify suitable elements to put into the generating set for the ideal in such a way that strings of different lengths become comparable. Figure 1 shows similarities between adjective phrases computed using vectors derived from the corpus in figure 2. The construction allows many properties of the tensor product to carry over into the quotient algebra, for example the similarity of *red book* to *red apple* is the same as the similarity of *book* to *apple*, as we would expect from the tensor product. Unlike the tensor product, strings of different length are comparable, so for example, the similarity of *apple* to *red apple* is non-zero. The benefit of using quotient algebras for compositional distributional semantics lies in this ability to extend the favourable properties of the tensor product by imposing linguistically plausible relations between vectors.

3 Learning Finite-dimensional Algebras

Quotient algebras are useful constructions when we have a small number of relations which we wish to impose on the tensor algebra. In highly lexicalised grammars, the number of relations we wish to impose may become so large that the ideal generates the whole vector space, and is thus useless, since the resulting quotient space will be trivial. An alternative to this is to restrict the space of exploration to finite-dimensional algebras. In this case, we can explore the space of possible products in relation to the set of relations we wish to hold; in other words, we can view this as an optimisation problem in which we want to find the best possible product given the required relations.

We apply this to the situation where we obtain a vector $\hat{x}$ for each individual word and pair of words in sequence. We then find the product that best fits these observed vectors. Given a set $W = \{w_1, w_2 \dots w_m\}$ of words, we want to define a product $\odot$ to minimise the difference between $\hat{w}_i \odot \hat{w}_j$ and $\widehat{w_i w_j}$, for $1 \leq i, j \leq m$. Specifically, we can define this as minimising

$$\sum_{i,j} \|\widehat{w_i w_j} - \hat{w}_i \odot \hat{w}_j\|$$

If word vectors have n dimensions, then $\odot$ is defined by an n^3 dimensional vector, which we denote f_{rst} for $1 \leq r, s, t \leq n$, where $(e_r \odot e_s)_t = f_{rst}$ and e is the vector with 1 in every component, and v_t is the tth component of v.

We can view this as a linear model:

$$(\widehat{w_i w_j})_t = \epsilon_{ijt} + \sum_{r,s=1}^{n} (\hat{w}_i)_r (\hat{w}_j)_s f_{rst}$$

where we have m^2 statistical units to learn n^2 parameters relating to the tth component of the vector space. Since these parameters are independent for each value of t, each set of n^2 parameters can be learnt in parallel. We are currently exploring ways of learning these parameters. The form of the equation above suggests the use of least squares, and we have performed some experiments using this method using a corpus extracted from the ukWaC corpus (Ferraresi et al., 2008). We extracted a list of $verb$ $adjective^* noun$ sequences, and used latent semantic analysis (Deerwester et al., 1990) to generate n-dimensional vectors for the 160 most common adjectives and nouns, and pairs of these adjectives and nouns. Our initial results indicate that the learnt parameters tend to get very large when using least squares to find the parameters, leading to poor results; we plan to investigate other methods such as linear optimisation.

Guevara (2010) proposed a related method of learning composition which used linear regression to learn how components compose. His model is however much more restrictive than ours in that the value of a component in the product depends only on that same component in the composed vectors, whereas in our model, the value of the component can depend on all components in the composed vectors.

Baroni and Zamparelli (2010) took a similar approach, in which adjectives are modelled as matrices acting on the space of nouns, and the matrices are learnt using least squares regression. The algebra products we propose learning are more general than matrix products; in addition we do not need to distinguish between words which are represented as matrices and words which are represented as vectors.

4 Constructing Algebras from Semigroups

Whilst the previous two techniques we have discussed are very general, and allow corpus data to be easily incorporated into the composition definition, our implementations are currently a long way from being able to represent the complexities of natural language semantics that is currently possible with logical semantics. This has become the standard method of representing natural language meaning, originating in the work of Montague (1973), however there is currently no way to incorporate statistical features of meaning that are described by the distributional hypothesis of Harris (1968).

Term	Context vector
fish	$(0, 0, 1)$
big	$(1, 2, 0)$

Figure 3: Example context vectors for terms.

$$\begin{aligned} n_i &= (N, \lambda x\ \mathrm{noun}_i(x)) \\ a_i &= (N/N, \lambda p \lambda y\ \mathrm{adj}_i(y) \wedge p.y) \end{aligned}$$

Figure 4: Equations describing syntax and semantics of adjectives and nouns.

In related work, Clark et al. (2008) described a method of composing meanings which they noted was a generalisation of Montague semantics. However, their version of Montague semantics assumed a particular model, and thus effectively mapped sentences to truth values. This omits much of the power of Montague semantics in which sentences are mapped to logical forms which then provide restrictions on the set of allowable models, allowing, for example, entailments to be computed between sentences.

We will sketch a method by which Montague semantics can be described within the context-theoretic framework. We follow a standard method of representing logic in language, but instead of representing words using logic, we represent an individual dimension of meaning of a word by a logical form — we call this dimension a "aspect". The general scheme is to represent aspects as elements of a semigroup, from which we form an algebra. Words are then represented as weighted sums over individual aspects.

We define a set S of all aspects as the set of pairs (s, σ), where s is the syntactic type of an aspect (for example in the Lambek calculus) and σ is the semantics of the aspect (for example described in the lambda calculus). We can extend S by defining a product on such pairs reducing each element to a normal form. This defines a semigroup: the Lambek calculus can be described in terms of a residuated lattice, which is a partially ordered semigroup (Lambek, 1958), and the lambda calculus is equivalent to a Cartesian closed category under β-equivalence (Lambek, 1985), which can be considered as a semigroup with additional structure.

Given any semigroup S we can construct an algebra $L^1(S)$ of real-valued functions on S which are finite under the L^1 norm with multiplication defined by convolution:

$$(u \cdot v)(x) = \sum_{y,z \in S : yz = x} u(y)v(z).$$

For example, suppose we have context vectors for the terms *big* and *fish* as described in Figure 3. We represent the syntax and semantics of adjectives and nouns by elements a_i and n_i respectively of a semigroup S (Figure 4), where we assume equivalence under β-reduction is accounted for. The predicates adj_i and $\mathrm{noun_j}$ correspond to aspects, in this case each dimension i of the three dimensions in the context vectors has a corresponding adj_i and noun_i. We may then represent the vectors for these terms as elements of the algebra $\widehat{big} = a_1 + 2a_2$ and $\widehat{fish} = n_3$, where we equate an element u of the semigroup with the function in the algebra $L^1(S)$ which maps u to 1 and every other element to zero. Then $\widehat{big}\ \widehat{fish} = a_1 n_3 + 2a_2 n_3$, where

$$a_i n_j = (N, \lambda x(\mathrm{noun}_j(x) \wedge \mathrm{adj}_i(x))).$$

Note that the elements a_i form a commutative, idempotent subsemigroup of S, so they have a semilattice structure. In order for this structure to carry over to the vector structure in the algebra, we would need a more sophisticated construction, such as a C* enveloping algebra; we leave the investigation of this possibility to further work.

5 Discussion

We have presented our initial investigations into the application of three powerful methods of constructing algebras to representing natural language semantics. Each of these approaches has potential use in representing meaning; here we have only touched the surface of what is possible with each technique. We

hope that with further work, these methods will lead to a true synthesis between logical and distributional approaches to natural language semantics.

6 Acknowledgments

We are grateful to Peter Hines, Stephen Clark and Peter Lane for useful discussions. The first author also wishes to thank Metrica for supporting this research.

References

Baroni, M. and R. Zamparelli (2010). Nouns are vectors, adjectives are matrices: Representing adjective-noun constructions in semantic space. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2010), East Stroudsburg PA: ACL*, pp. 1183–1193.

Clark, S., B. Coecke, and M. Sadrzadeh (2008). A compositional distributional model of meaning. In *Proceedings of the Second Quantum Interaction Symposium (QI-2008)*, Oxford, UK, pp. 133–140.

Clark, S. and S. Pulman (2007). Combining symbolic and distributional models of meaning. In *Proceedings of the AAAI Spring Symposium on Quantum Interaction*, Stanford, CA, pp. 52–55.

Clarke, D. (2007). *Context-theoretic Semantics for Natural Language: an Algebraic Framework*. Ph. D. thesis, Department of Informatics, University of Sussex.

Clarke, D., R. Lutz, and D. Weir (2010, July). Semantic composition with quotient algebras. In *Proceedings of the 2010 Workshop on GEometrical Models of Natural Language Semantics*, Uppsala, Sweden, pp. 38–44. Association for Computational Linguistics.

Deerwester, S., S. Dumais, G. Furnas, T. Landauer, and R. Harshman (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Science 41*(6), 391–407.

Ferraresi, A., E. Zanchetta, M. Baroni, and S. Bernardini (2008). Introducing and evaluating ukwac, a very large web-derived corpus of english. In *Workshop Programme*, pp. 47.

Guevara, E. (2010). A Regression Model of Adjective-Noun Compositionality in Distributional Semantics. *ACL 2010*, 33.

Harris, Z. (1968). *Mathematical Structures of Language*. Wiley, New York.

Lambek, J. (1958). The mathematics of sentence structure. *American Mathematical Monthly 65*, 154–169.

Lambek, J. (1985, May). Cartesian closed categories and typed lambda-calculi. In G. Cousineau, P.-L. Curien, and B. Robinet (Eds.), *Combinators and Functional Programming Languages*, Lecture Notes in Computer Science. Springer-Verlag.

Montague, R. (1973). *The proper treatment of quantification in ordinary English*. Dordrecht, Holland: D. Reidel Publishing Co.

Smolensky, P. (1990, November). Tensor product variable binding and the representation of symbolic structures in connectionist systems. *Artificial Intelligence 46*(1-2), 159–216.

Widdows, D. (2008). Semantic vector products: Some initial investigations. In *Proceedings of the Second Symposium on Quantum Interaction, Oxford, UK*.