Language is used to describe concepts, and many of these concepts are hierarchical. Modelling words as vectors within distributional semantics does not intrinsically incorporate this hierarchical information. We model words as positive operators. These have an ordering which we interpret as modelling hierarchical information. We give a simple and intuitive method for building positive operators from off-the-shelf vectors together with hierarchical information from WordNet. The word representations built can be composed within the categorical compositional semantic framework, and give competitive results on sentence-level entailment datasets.

1 Introduction

Distributional semantics is effective and important within the area of computational modelling of language, particularly as regards to synonymy and paraphrasing. Within the field, at least two additional properties are desirable. Firstly, we would like a method by which we can compose vectors to form representations above the word level. Secondly, we would like a notion of lexical entailment, or hyponymy, with which we can capture a sense of the generality of concepts, and the notion of one concept being an instance of another. Furthermore, we would like these two properties to interact nicely with one another, so that the hyponymy relation is not lost when words are composed. In Bankova et al. [2016] the authors provide theory describing a notion of hyponymy that interacts well with compositionality, but do not provide experimental support. In Balkır et al. [2016] the authors suggest a measure of hyponymy based on entropy which also interacts well with compositionality and
provide experimental support. A compositional version of the distributional inclusion hypothesis (DIH) [Geffet and Dagan, 2005] is examined in Kartsaklis and Sadrzadeh [2016]. In this paper, we use the framework of Bankova et al. [2016] to build positive operators that represent words. The operators are built using GloVe vectors [Pennington et al., 2014] and information from WordNet [Miller, 1995]. As such, our approach is a hybrid approach, using both distributional and human-curated information. We give two new measures for graded hyponymy that provide a wider range of comparisons than the entropy-derived measure developed in Balkır et al. [2016] or the eigenvalue-related measure of Bankova et al. [2016], and give proposals for a number of compositional approaches. We test our models on the compositional dataset of Kartsaklis and Sadrzadeh [2016]. Our models perform very strongly on this dataset.

2 Background

Vector space models of meaning usually rely on some form of the distributional hypothesis: that words that occur in similar contexts have similar meanings. However, as well as deriving word meanings, we also need to give meanings to sentences and phrases. This means that we need some method for composing vector representations of words. The categorical compositional distributional (DisCoCat) model of Coecke et al. [2010] is based in the idea that grammatical formalisms such as pregroup grammar [Lambek, 1999] have the same categorical structure as the category $\text{FVect}$ whose objects are finite dimensional vector spaces and whose morphisms are linear maps. Moreover, DisCoCat is flexible with regard to the grammar and semantic representations it uses.

2.1 DisCoCat

We explain DisCoCat briefly. For more details, see Coecke et al. [2010], Preller and Sadrzadeh [2011]. A grammar for English is represented in a compact closed category. The grammar is then mapped via a strong monoidal functor, as described in [Preller and Sadrzadeh, 2011], to the category $\text{FVect}$ of finite-dimensional vector spaces and linear maps. The grammar we discuss here is pregroup grammar. It is possible to use other forms of grammar [Coecke et al., 2013] or $\lambda$-calculus [Muskens and Sadrzadeh, 2016]. Pregroup grammar is built over a set of types. We consider the set containing $n$ for noun and $s$
for sentence. Each type has adjoints $x^r$ and $x^l$. Complex types are built up by concatenation of types, and we often leave out the dot so that $xy = x \cdot y$. There is a unit type such that $1x = 1 = x1$. Types and their adjoints interact via:

$$
\begin{align*}
\epsilon^r_x : x \cdot x^r & \rightarrow 1, \\
\epsilon^l_x : x^l \cdot x & \rightarrow 1 \\
\eta^r_x : 1 & \rightarrow x^r \cdot x, \\
\eta^l_x : 1 & \rightarrow x \cdot x^l
\end{align*}
$$

(1)

A string of grammatical types $t_1,...t_n$ is grammatical if it reduces, via the morphisms above, to the sentence type $s$. For example, typing clowns as $n$, tell as $n^r sn^l$ and the truth as $n$, the sentence Clowns tell the truth has type $n(n^r sn^l)n$ and is shown to be grammatical as follows:

$$(\epsilon^r 1 \epsilon^l)n(n^r sn^l)n \rightarrow (\epsilon^r 1)(n n^r s 1) \rightarrow 1 s 1 = s$$

(2)

This grammar is mapped to $\textbf{FVect}$ by sending each atomic type to a vector space, concatenation to tensor product of vector spaces and type reduction to tensor contraction.

2.2 DisCoCat in $\textbf{CPM(\textbf{FVect})}$

In Piedeleu et al. [2015], Bankova et al. [2016], and Balkır et al. [2016] the DisCoCat model is lifted to the category $\textbf{CPM(\textbf{FVect})}$, which has the same objects as $\textbf{FVect}$, but whose morphisms are now completely positive maps. The $\textbf{CPM}$ construction is introduced in Selinger [2007]. Words are now represented as positive operators rather than as vectors, and that maps between them are completely positive maps. To explain, we use physicists’ convenient bra–ket notation. For unit vector $|v\rangle$, the projection operator $|v\rangle \langle v|$ onto the subspace spanned by $|v\rangle$ is called a pure state. In general, density matrices are given by a convex sum of pure states, describing a probabilistic mixture. Necessary and sufficient conditions for an operator $\rho$ to encode such a mixture are that $\forall v \in V. \langle v|\rho|v\rangle \geq 0$, $\rho$ is self–adjoint, and $\rho$ has trace 1. Operators satisfying the first two conditions are called positive operators. The third ensures that the operator represents a convex mixture of pure states. Relaxing this condition gives us different choices for normalization.

2.3 Ordering positive operators

The set of positive operators on a vector space has an ordering introduced by Löwner [1934]. For positive operators $A$ and $B$, we define:

$$A \preceq B \iff B - A \text{ is positive}$$
This ordering is interpreted in DisCoCat as an hyponymy relation. If we have a positive operator $[\text{mammal}]$ representing the word \textit{mammal}, and a positive operator $[\text{dog}]$ representing the word \textit{dog}, then we would like to see:

$$[\text{dog}] \sqsubseteq [\text{mammal}]$$

In Bankova et al. [2016] the authors introduce a notion of graded hyponymy. Consider the relationship between \textit{dog} and \textit{pet}. Not all dogs are pets: some are working dogs and some are wild. We therefore want to say that $[\text{dog}] \sqsubseteq [\text{pet}]$ up to some value $k \in [0,1]$. In Bankova et al. [2016], the hyponymy relation may be true up to some error term, as follows. If $A \sqsubseteq B$, then $B - A = D$, i.e. $A + D = B$, where $D$ is some positive operator. However, if this does not hold, it may also be possible to add in some error term $E$ so that $A \sqsubseteq B + E$. This is viewed as saying that $A$ entails $B$ to the extent $E$. Combining definitions, $A + D = B + E$, and so trivially $A$ entails $B$ to the extent $A$. We wish to find the smallest such error term.

In Bankova et al. [2016], the error term was of the form $(1 - k)A$ and the scalar $k \in [0,1]$ gave a graded notion of hyponymy. The effect of this scalar is to reduce the size of $A$ until it ‘fits inside’ $B$, giving a notion of graded hyponymy that says that $A$ is a $k$-hyponym of $B$, $A \sqsubseteq_k B$ if $B - kA$ is positive.

3 Methods

3.1 Measuring hyponymy

One of the drawbacks of the measure of graded entailment given in Bankova et al. [2016] is that if the space spanned by eigenvectors of $A$, called $\text{Span}(A)$, is not a subspace of $\text{Span}(B)$, then the value of $k$ must be 0. We therefore consider two new measures, which we now describe. If $B - A$ is not positive, it is possible to make it positive by adding in a positive operator constructed in the following manner. Firstly diagonalize $B - A$, resulting in a real-valued matrix, since $B - A$ is real symmetric. Construct a matrix $E$ by setting all positive eigenvalues of $B - A$ to 0 and changing the sign of all negative eigenvalues. Then $B - A + E$ will give us a positive matrix. This $E$ is our error term. The size of $E$ is bounded above by the size of $A$, since in the worst case $E = A$. We propose two different measures related to this error term that give us values in $\mathbb{R}$, giving a grading for hyponymy.
The first measure is

\[ k_{BA} = \frac{\sum_i \lambda_i}{\sum_i |\lambda_i|} \]  

(3)

where \( \lambda_i \) is the \( i \)th eigenvalue of \( B - A \) and \( |\cdot| \) indicates absolute value. This measures the proportions of positive and negative eigenvalues in the expression \( B - A \). If all eigenvalues are negative, \( k_{BA} = -1 \), and if all are positive, \( k_{BA} = 1 \). This measure is balanced in the sense that \( k_{BA} = -k_{AB} \).

Secondly, we propose

\[ k_E = 1 - \frac{||E||}{||A||} \]  

(4)

where \( ||\cdot|| \) denotes the Frobenius norm. This measures the size of the error term as a proportion of the size of \( A \). Since \( A = E \) in the worst case, this measure ranges from 0 when \( E = A \) to 1 when \( E = 0 \).

3.2 Constructing positive operators from a corpus

We represent words as positive matrices following the approach outlined in Bankova et al. [2016]. In that work, the authors observe that each word vector has a corresponding pure matrix:

\[ |\text{cat}\rangle \rightarrow |\text{cat}\rangle \langle \text{cat}| \]

Words which are more general can then be built up by summing over the projectors corresponding to the hyponyms of that word.

In general, the meaning of a word \( w \) is considered to be given by a collection of unit vectors \( \{|w_i\}\)\), where each \( |w_i\)\) represents an instance of the concept expressed by the word. Then the operator:

\[ \|w\| = \sum_i p_i |w_i\rangle \langle w_i| \in W \otimes W \]  

(5)

represents the word \( w \). The \( p_i \) are weightings derived from the text, and there are various choices about what these should be.

We build representations of words as positive operators in the following manner. Suppose we have a dictionary of word vectors \( \{v_i : |v_i\rangle \in W\} \), derived from a corpus using standard distributional or embedding techniques, for example GloVe, Pennington et al. [2014], FastText Bojanowski et al. [2017], or weighted co-occurrence vectors. To build a representation of a word, we obtain a set of hyponyms that are instances of that word. In this paper, we use WordNet Miller [1995], a human-curated database of word relationships.
including hyponym–hypernym pairs. The WordNet hyponymy relationship is naturally arranged as a directed graph with a root (it is not quite a tree). For the noun subset of the database, the root is the most general noun entity, and the leaves are specific nouns. For example, under the word rocket there are (inter alia): test_instrument_vehicle, Stinger, takeoff_booster, arugula. Notice that here we have different meanings of the word rocket, one as a projectile and one as a vegetable. There are also less supervised ways of obtaining these relationships using patterns derived from text, see Hearst [1992], Roller et al. [2018], Lewis [2019] for examples.

To build a positive operator for a word $w$, we go through the WordNet hierarchy and collect all hyponyms $w_i$ of $w$ at all levels. We then form $\|w\|$ as in equation (5), with $p_i = 1$ for all $i$. When we build these operators, between 1/3 and 1/2 of the hyponyms listed in WordNet are available in GloVe, and we therefore miss a large proportion of the information included in WordNet.

An important parameter choice is the type of normalization to use. In Bankova et al. [2016] two choices are discussed: normalizing operators to trace 1, or normalizing operators to have maximum eigenvalue less than or equal to 1. In previous work [Lewis, 2019] we have shown that extremely good results can be obtained using no normalization at all. However, applying a maximum eigenvalue normalization means that further operations like applying negation are likely to become easier, and hence in this paper we investigate how well our models can do with normalization.

3.3 Composing positive operators

One of the strengths of DisCoCat is its formal approach to composition. Within the category $\text{CPM}(\text{FVect})$ objects are finite-dimensional vector spaces and morphisms are completely positive maps.

In order to build these maps we use the type-lifting methods outlined in Kartsaklis et al. [2012]. A Frobenius algebra over a finite-dimensional vector space with bases $\{\vec{n}_i\}$, is given by

$$\Delta :: \vec{n}_i \mapsto \vec{n}_i \otimes \vec{n}_i \quad \iota :: \vec{n}_i \mapsto 1 \quad \mu :: \vec{n}_i \otimes \vec{n}_i \mapsto \vec{n}_i \quad \xi :: 1 \mapsto \vec{n}_i$$

A vector $\vec{v} \in V$ can be lifted to a higher-order representation $\vec{v}' \in V \otimes V$ by applying the map $\Delta$. The Frobenius algebra interacts with the type reduction morphism $\epsilon_N$ in such a way that the result of lifting an adjective and then
composing with a noun is to apply the \( \mu \) multiplication to the tensor product of the adjective and the noun vectors.

In \( \text{FVect} \) the multiplication \( \mu \) implements pointwise multiplication of the two vectors. However in \( \text{CPM(FVect)} \) we have different choices for the multiplication \( \mu \). One is composition of the two operators. This results in a matrix that is no longer self-adjoint, and so Piedeleu [2014] suggests using the non-commutative and non-associative operator \( \frac{1}{2} p_2^* p_1 p_2^* \) in its place. Piedeleu [2014] also notes that the pointwise multiplication of two positive operators is a completely positive map, giving us another choice for composition.

Following Kartsaklis et al. [2012], this gives us a method for building verb operators from their lower-level operators. Firstly, we assume the noun space \( N \otimes N \) to be equal to the sentence space \( S \otimes S \), and refer to these both as \( W \otimes W \). Given a representation of an intransitive verb \([\text{verb}] \in W \otimes W\), we lift it to \( \Delta([\text{verb}]) \in W \otimes W \otimes W \otimes W \). Composing with a noun implements \([\text{noun} \text{ verb}] = \mu([\text{noun}] \otimes [\text{verb}])\).

Lastly, we can form a completely positive map from a positive matrix \( A \) by decomposing \( A \) into a weighted sum of orthogonal projectors \( A = \sum_i p_i P_i \), and then forming the map
\[
A(-) = \sum_i p_i P_i \circ - \circ P_i
\]
The same proposal for composition is given in Coecke [2019].

For intransitive verbs we combine the noun and the verb via three operations \textbf{Mult}, \textbf{BMult}, \textbf{KMult}.

\begin{align*}
\textbf{Mult}: [\text{noun } \text{verb}] &= [\text{verb}([\text{noun}])] = [\text{noun}] \odot [\text{verb}] \\
\textbf{BMult}: [\text{noun } \text{verb}] &= [\text{verb}([\text{noun}])] = [\text{verb}]^\frac{1}{2} [\text{noun}] [\text{verb}]^\frac{1}{2} \\
\textbf{KMult}: [\text{noun } \text{verb}] &= [\text{verb}([\text{noun}])] = \sum_i p_i P_i [\text{noun}] P_i
\end{align*}
where in \( \text{KMult} [\text{verb}] = \sum_i p_i P_i \). We also investigate switched versions of \textbf{BMult} and \textbf{KMult}, where the order of composition is switched.

For transitive verbs there is one possibility for pointwise multiplication of the operators, since this is both commutative and associative. For \textbf{BMult} and \textbf{KMult} there are a number of composition orders. We will concentrate on two which reflect the difference between viewing verb as operator and viewing nouns as operator. Both compose verb and object, then verb phrase and
subject. We therefore have:

\[ \text{Mult: } [\text{subj verb obj}] = [\text{subj}] \odot [\text{verb}] \odot [\text{obj}] \quad (9) \]
\[ \text{BMult-V: } [\text{subj verb obj}] = [\text{vp}]^{\frac{1}{2}} [\text{subj}] [\text{vp}]^{\frac{1}{2}} \text{ where } [\text{vp}] = [\text{verb}] ([\text{obj}]) \quad (10) \]
\[ \text{KMult-V: } [\text{subj verb obj}] = \sum_i p_i P_i [\text{subj}] P_i \text{ where } \sum_i p_i P_i = [\text{verb}] ([\text{obj}]) \quad (11) \]
\[ \text{BMult-N: } [\text{subj verb obj}] = \left[ [\text{subj}]^{\frac{1}{2}} [\text{vp}] [\text{subj}]^{\frac{1}{2}} \text{ where } [\text{vp}] = [\text{obj}] ([\text{verb}]) \right] \quad (12) \]
\[ \text{BMult-N: } [\text{subj verb obj}] = \sum_i p_i P_i [\text{vp}] P_i \text{ where } \sum_i p_i P_i = [\text{subj}] \quad (13) \]

The notation \([A]([B])\) refers to the corresponding two-place operation, either \(\text{KMult}\) or \(\text{BMult}\).

4 Experimental setting

We test our word representations and composition methods on the compositional datasets of Sadrzadeh et al. [2018]. This is a series of three datasets, covering simple intransitive sentences, transitive sentences, and verb phrases. The intransitive verb dataset consists of paired sentences consisting of a subject and a verb. In half the cases the first sentence entails the second, and in the other half of cases, the order of the sentences is reversed. For example:

summer finish, season end, T
season end, summer finish, F

The first sentence is marked as entailing, whereas the second is marked as not entailing. The dataset is created by selecting nouns and verbs from WordNet that stand in the correct relationship. The transitive verb and verb phrase datasets are similarly created.

To test our models, we build the basic word representations as in equation (5). We then use the compositional methods outlined in section 3.3 to create the sentence representations. We calculate the graded entailment value between the composed sentence representations, and in results report area under ROC curve for comparison with previous literature. We use GloVe vectors in 50 dimensions. Modest improvements can be obtained using larger numbers of dimensions, but qualitative results are similar. The basic operators we build are normalised to have maximum eigenvalue 1. We want to retain the property of having maximum eigenvalue less than or equal to 1. The \(\text{BMult}\), \(\text{KMult}\)
Table 1: Area under ROC curve on the KS2016 datasets. For the SV and VO datasets, BMult1 and KMult1 refer to the models described in equations (7) and (8). BMult2 and KMult2 refer to the models described in equations (??) and (??). For SVO, BMult1 and KMult1 refer to the models described in equations (10) and (11) and BMult2 and KMult2 refer to the models described in equation (12) and (13).

<table>
<thead>
<tr>
<th>Model</th>
<th>$k_E$ measure</th>
<th>$k_B A$ measure</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SV</td>
<td>VO</td>
</tr>
<tr>
<td>KS2016 best</td>
<td>0.84</td>
<td>0.82</td>
</tr>
<tr>
<td>Verb only</td>
<td>0.601</td>
<td>0.677</td>
</tr>
<tr>
<td>Addition</td>
<td>0.646</td>
<td>0.645</td>
</tr>
<tr>
<td>Mult</td>
<td>0.850</td>
<td>0.817</td>
</tr>
<tr>
<td>BMult1</td>
<td>0.725</td>
<td>0.704</td>
</tr>
<tr>
<td>BMult2</td>
<td>0.750</td>
<td>0.677</td>
</tr>
<tr>
<td>KMult1</td>
<td>0.886</td>
<td>0.766</td>
</tr>
<tr>
<td>KMult2</td>
<td>0.798</td>
<td>0.863</td>
</tr>
</tbody>
</table>

operators and their variants preserve this property by Weyl’s inequalities [Weyl, 1912] and the orthogonality of projectors in KMult. For the other operators, if the maximum eigenvalue of the composed expression is greater than 1, we normalize, else we leave it as is.

5 Results

On the KS2016 compositionality datasets results are reported in terms of area under ROC curve (Table 1). Overall, the $k_B A$ measure works best with the composition operators, with every operator outperforming the previous best results on this dataset. Across both measures, the KMult operators perform well. There is a noticeable pattern that different phrase types perform better with different composition orders. Specifically, subject–verb phrases work well with the verb as operator, and SVO and verb–object phrases work better interpreting the nouns as operators.

The good performance of the KMult model is likely to be due to the fact that both the dataset and our word representations were constructed from WordNet, and hence the high performance is to be expected. However, it is still interesting that our representations work so well with the compositional operations.
6 Discussion and further work

We have suggested a mechanism for building the positive operators needed for the theory presented in Bankova et al. [2016], together with two novel measures of graded hyponymy. The representations and the measures we have developed perform competitively on phrase and sentence datasets. The type of representation we have developed is a hybrid representation in the sense that we use off-the-shelf distributional vectors, but also human-provided information from WordNet. The representations are extremely quick to build, with no training time.

Exploring the ordering of composition will be an important area of further work – it is unclear what it means to interpret the nouns as operators rather than the verbs. One line of inquiry could be to look into understanding this via the type-raising operations of categorial grammar.

Similarities to our approach can be found in the notion of words being represented as Gaussians [Jameel and Schockaert, 2017, Vilnis and McCallum, 2014]. The positive operators we build have the same structure as covariance matrices and, if appropriately normalized, are interpreted as representing a probability distribution over vectors. Exploring these connections is an area of further work.

Finally, a crucial extension to this whole approach is to be able to model hyponymy, composition, and their interaction in downwardly monotone contexts, using the natural logic introduce in Barwise and Cooper [1981], MacCartney and Manning [2007]. This is an important area of future research.

References


Bill MacCartney and Christopher D Manning. Natural logic for textual infer-


