# Prior Disambiguation of Word Tensors for Constructing Sentence Vectors

**Dimitri Kartsaklis**
University of Oxford
Department of
Computer Science
Wolfson Building, Parks Road
Oxford, OX1 3QD, UK
dimitri.kartsaklis@cs.ox.ac.uk

**Mehrnoosh Sadrzadeh**
Queen Mary University of London
School of Electronic Engineering
and Computer Science
Mile End Road
London, E1 4NS, UK
mehrs@eecs.qmul.ac.uk

## Abstract

Recent work has shown that compositional-distributional models using element-wise operations on contextual word vectors benefit from the introduction of a prior disambiguation step. The purpose of this paper is to generalise these ideas to tensor-based models, where relational words such as verbs and adjectives are represented by linear maps (higher order tensors) acting on a number of arguments (vectors). We propose disambiguation algorithms for a number of tensor-based models, which we then test on a variety of tasks. The results show that disambiguation can provide better compositional representation even for the case of tensor-based models. Furthermore, we confirm previous findings regarding the positive effect of disambiguation on vector mixture models, and we compare the effectiveness of the two approaches.

## 1 Introduction

Distributional models of meaning have been proved extremely useful for a number of natural language processing tasks, ranging from thesaurus extraction (Curran, 2004) to topic modelling (Landauer and Dumais, 1997) and information retrieval (Manning et al., 2008), to name just a few. These models are based on the *distributional hypothesis* of Harris (1968), which states that the meaning of a word depends on its context. This idea allows the words to be represented by vectors of statistics collected from a sufficiently large corpus of text; each element of the vector reflects how many times a word co-occurs in the same context with another word of the vocabulary. However, due to the generative power of natural language, which is able to produce infinite new structures from a finite set of resources (words), no text corpus, regardless of its size, can provide reliable distributional representations for anything longer than single words or perhaps very short phrases consisting of two words; in other words, this technique cannot scale up to the phrase or sentence level.

Much research activity has been recently dedicated to provide a solution to this problem: although the direct construction of a sentence vector is not possible, we might still be able to synthetically create such a vectorial representation by somehow *composing* the vectors of the words that comprise the sentence. Towards this goal, researchers have employed a variety of approaches that roughly fall into two general categories. Following an influential work (Mitchell and Lapata, 2008), the models in the first category compute a sentence vector as a mixture of the original word vectors, using simple operations such as element-wise multiplication and addition; we refer to these models as *vector mixtures*. The main characteristic of these models is that they do not distinguish between the type-logical identities of the different words: an intransitive verb, for example, is of the same order as its subject (a noun), and both will contribute equally to the composite sentence vector.

However, this symmetric treatment of composition seems unjustified from a formal semantics point of view. Words with special meanings, such as verbs and adjectives, are usually seen as functions acting on, hence modifying, a number of arguments rather than lexical units of the same order as them; an adjective, for example, is a function that returns a modified version of its input noun. Inspired from this more-aligned-to-formal-semantics view, a second research direction aims to represent relational words as linear maps (tensors of various orders) that can be applied to one or more arguments (vectors). Baroni and Zamparelli (2010), for example, model adjectives as matrices which, when matrix-multiplied with a noun vector, will produce a vectorial representation of the specific adjective-noun compound. The notion of a framework where relational words are entities living in vector spaces of higher order than nouns, which are simple vectors, has been formalized by Coecke et al. (2010) in the context of the abstract mathematical framework of compact closed categories. We refer to this class of models as *tensor-based*.

Regardless of the way they approach the representation of relational words and their composition operation, however, most current compositional-distributional models do share a common feature: they all rely on ambiguous vector representations, where all the senses of a polysemous word, such as the verb 'file' (which can mean *register* or *smooth*), are merged into the same vector or tensor. At least for the vector mixture approach, this practice has been proved suboptimal: Reddy et al. (2011) and Kartsaklis et al. (2013) test a number of simple multiplicative and additive models using disambiguated vector representations on various tasks, showing that the introduction of a disambiguation step prior to actual composition can indeed increase the quality of the composite vectors. However, the fact that disambiguation can be beneficial for models based on vector mixtures is not very surprising. Both additive and multiplicative compositions are but a kind of average of the vectors of the words in the sentence, hence can directly benefit from the provision of more accurate starting points. Perhaps a more interesting question, and one that the current paper aims to address, is to what extent disambiguation can also provide benefits for tensor-based approaches, which in general constitute more powerful models for natural language (see discussion in Section 2).

Specifically, this paper aims to: (a) propose disambiguation algorithms for a number of tensor-based distributional models; (b) examine the effect of disambiguation on tensors for relational words; and (c) meaningfully compare the effectiveness of tensor-based against vector mixture models in a number of tasks. Based on the generic procedure of Schütze (1998), we propose algorithms for a number of tensor-based models, where the composition is modelled as the application of linear maps (tensor contractions). Following Mitchell and Lapata (2008) and many others, we test our models on two disambiguation tasks similar to that of Kintsch (2001), and on the phrase similarity task introduced in (Mitchell and Lapata, 2010). In almost every case, the results show that disambiguation can make a great difference in the case of tensor-based models; they also reconfirm previous findings regarding the effectiveness of the method for simple vector mixture models.

## 2 Vectors vs tensors

The simple models of Mitchell and Lapata (2008) constitute the easiest and perhaps the most intuitive way of composing two or more vectors: each element of the resulting vector is computed as the sum or the product of the corresponding elements in the input vectors (left part in Figure 1). In the case of addition, the components of the output vector are simply the cumulative scores of the corresponding input components. So in a sense the output element embraces both input elements, resembling a *union* of the input features. On the other hand, the element-wise multiplication of two vectors can be seen as the *intersection* of their features: a zero element in one of the input vectors will eliminate the corresponding feature in the output, no matter how high the other input component was. In addition to failing to identify the special roles of words in a sentence, vector mixture models disregard grammar in another way: the commutativity of operators make them a bag-of-words approach, where the meaning of sentence 'dog bites man' is equated to that of 'man bites dog'.

On the contrary to the above element-wise treatment, a compositional approach based on linear maps computes each element of the resulting vec-

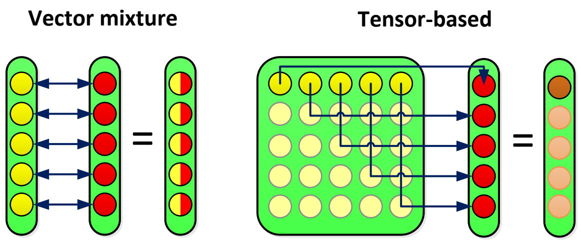

Figure 1: Vector mixture and tensor-based models for composition. In the latter approach, the $i$th element of the output vector is the linear combination of the input vector with the $i$th row of the matrix.

tor via a linear combination of *all* the elements of the input vector (right part of Figure 1); in other words, possible interdependencies between different features are also taken into account, offering (in principle) more power. Furthermore, by design, the bag-of-words problem is not present here. Overall, tensor-based models offer a more complete and linguistically motivated solution to the problem of composition. For example, one can consider building linear maps for prepositions and logical words, rather than treating them as noise and discard them, as commonly done in the vector mixture models.

## 3 Disambiguation in vector mixtures

For a compositional model based on vector mixtures, polysemy of words can be a critical factor. Pulman (2013) and Kartsaklis et al. (2013) point out that the element-wise combination of "ambiguous" vectors produces results that are hard to interpret; the composed vector is not a purely compositional representation but a product of two tasks that take place in parallel: composition and *some* amount of disambiguation that emerges as a side-effect of the compositional process, leaving the resulting vector in an intermediate state.

This effect is demonstrated in Figure 2, which shows the composition of the ambiguous verb 'run' (with meanings *moving fast* and *dissolving*) with the subject 'horse'. The first three components of our toy vector space are related to the *dissolving* meaning, while the last three of them to the *moving fast* meaning. An ambiguous vector for 'run' will have non-zero values for every component. On the other hand, we would expect the vector for 'horse' to have high values for the 'race', 'gallop', and 'move' components, and very low values (but not necessarily zero) for the dissolving-related ones—it is always possible for the word 'horse' to appear in the same

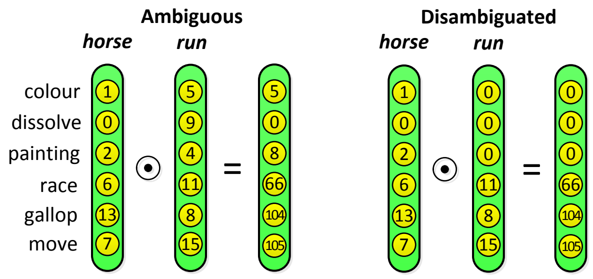

Figure 2: The effect of disambiguation on vector composition. The numbers are (artificial) co-occurrence counts of each target word with the 6 basis words on the left.

context with the word 'painting', for example. The left part of Figure 2 shows what happens when the ambiguous 'run' vector is used; the multiplication with the 'horse' vector will produce an impure result, half affected by composition and half by disambiguation. However, what we really want is a vector where all the dissolving-related components will be eliminated, since they are irrelevant to the way the word 'run' is used in the sentence. In order to achieve this, we have to introduce a disambiguation step prior to composition (right part of Figure 2).

These ideas are experimentally verified in the works of Reddy et al. (2011) and Kartsaklis et al. (2013); Pulman (2013) also presents a comprehensive analysis of the problem. What remains to be seen is if disambiguation can also provide benefits for the linguistically motivated setting of tensor-based models, the principles of which are shortly discussed in the next section.

## 4 Tensors as multilinear maps

A *tensor* is a geometric object that can be seen as the generalization of the familiar notion of a vector in higher dimensions. The *order* of a tensor is the number of its dimensions; in other words, the number of indices we need to fully describe a random element of the tensor. Hence, a vector is a tensor of order 1, a matrix is a tensor of order 2, and so on. Tensors and multilinear maps stand in one-to-one correspondence, as stated by the following well-known "map-state" isomorphism (Bourbaki, 1989):

$$f : V_1 \to \ldots \to V_j \to V_k \cong V_k \otimes V_j \otimes \ldots \otimes V_1 \quad (1)$$

This offers an elegant way to adopt a formal semantics view of natural language in vector spaces. Let nouns live in a basic vector space $N \in \mathbb{R}^D$; returning to our previous example, an adjective then can be seen as a map $f : N \to N$ which is isomorphic to $N \otimes N$ (that is, to a matrix). In general, the order of the tensor is equal to the number of arguments plus one dimension that carries the result; so a unary function (e.g. adjectives, intransitive verbs) is represented by a tensor of order 2 (a matrix), a binary function (e.g. a transitive verb) as an order 3 tensor, and so on. Due to the above isomorphism, function application (and hence our compositional operation) becomes a generalisation of matrix multiplication, formalised in terms of the inner product. In the case of a unary relational word, such as an adjective, this is nothing more than the usual notion of matrix multiplication between a matrix and a vector. The generalization of this process to tensors of higher order is known as *tensor contraction*. Given two tensors of orders $n$ and $m$, the tensor contraction operation will always produce a tensor of order $n + m - 2$.

Let us see an example of how this works for a simple transitive sentence. Let $\mathcal{V} \in \mathbb{R}^{I \times J \times K}$ be the tensor of order 3 for the verb and $\mathcal{S} \in \mathbb{R}^{I}$, $\mathcal{O} \in \mathbb{R}^{K}$ the tensors of order 1 (vectors) for the subject and the object of the verb, respectively. Then $\mathcal{V} \times \mathcal{O}$ will return a new tensor living in $\mathbb{R}^{I \times J}$ (i.e. a matrix)[1]; a further interaction of this result with the subject will return a vector for the whole transitive sentence living in $\mathbb{R}^{J}$. We should note that the order in which the verb is applied to its arguments is not important; so in general the meaning of a transitive sentence is given by:

$$(\mathcal{V} \times \mathcal{O})^{\mathsf{T}} \times \mathcal{S} = (\mathcal{V}^{\mathsf{T}} \times \mathcal{S}) \times \mathcal{O} \qquad (2)$$

where $\mathsf{T}$ denotes a transpose and makes indices match, since subject precedes the verb.

## 5 Creating verb tensors

In this section we review a number of proposals regarding concrete methods of constructing tensors for relational words in the context of the frameworks of Coecke et al. (2010) and Baroni and Zamparelli (2010), which both comply to the setting of Section 4.[2]

**Relational** Following ideas from the set-theoretic view of formal semantics, Grefenstette and Sadrzadeh (2011a) suggest that the meaning of a relational word should be represented as the sum of its arguments. The meaning of adjective ‘red’, for example, becomes the sum of the vectors of all the nouns that ‘red’ modifies in the corpus; so $\overrightarrow{red} = \sum_i \overrightarrow{noun_i}$, where $i$ iterates through all the occurrences of ‘red’. This can be generalised to relational words of any arity, by summing the tensor product of their arguments. So for a transitive verb we have:

$$\overline{verb}^2 = \sum_i (\overrightarrow{subj}_i \otimes \overrightarrow{obj}_i) \qquad (3)$$

[1]The symbol $\times$ denotes tensor contraction.

[2]In what follows we use the case of a transitive verb as an example; however the descriptions apply to any relational word of any arity. A vector (an order-1 tensor) is denoted as $\overrightarrow{x}$; tensors of order $n > 1$ are shown as $\overline{x}^n$ for clarity.

where $i$ again iterates over all occurrences of the specific verb in the corpus and the superscript denotes the order of the tensor.

In order to achieve a more expressive representation for the sentences, the authors used the convention that the arity of the head word in a sentence will also determine the order of the sentence space; that is, the space of intransitive sentences will be of order 1, of transitive ones will be of order 2, and so on. Recall from Section 4 that for the transitive case this increases the order of the verb tensor to 4 (2 dimensions for the arguments plus another 2 for the result). In spite of this, however, note that the method of Equation 3 produces a matrix. The other two dimensions of the tensor remain empty (filled with zeros), a fact that simplifies the calculations but also considerably weakens the expressive power of the model. This simplification transforms Equation 2 to the following:

$$\overline{subj\ verb\ obj}^2 = (\overrightarrow{subj} \otimes \overrightarrow{obj}) \odot \overline{verb}^2 \qquad (4)$$

where $\otimes$ denotes the tensor product and $\odot$ element-wise multiplication.

**Kronecker** In a subsequent work (Grefenstette and Sadrzadeh, 2011b), the same team proposes the creation of a verb matrix as the Kronecker product of the verb’s contextual vector with itself:

$$\overline{verb}^2 = \overrightarrow{verb} \otimes \overrightarrow{verb} \qquad (5)$$

Again in this model the sentence space is of order 2, and the meaning of a transitive sentence is calculated using Equation 4.

**Frobenius** The previous models bring the important limitation that only sentences of the same structure can be meaningfully compared; it is not possible, for example, to compare an intransitive sentence (e.g. ‘kids play’) with a transitive one (‘children play football’), since the former is a vector and the latter a matrix. Using Frobenius algebras, Kartsaklis et al. (2012) provide a unified sentence space for every sentence regardless of its type. These models turn the matrix of Equation 3 to a tensor of order 3 (as required by the type-logical identities) by copying one of the existing dimensions. When the dimension of rows (corresponding to subjects) is copied, the calculation of a vector for a transitive sentence becomes:

$$\overrightarrow{subj\ verb\ obj} = \overrightarrow{subj} \odot (\overline{verb}^2 \times \overrightarrow{obj}) \qquad (6)$$

Copying the column dimension (objects) gives:

$$\overrightarrow{subj\ verb\ obj} = \overrightarrow{obj} \odot \left( (\overline{verb}^2)^\mathsf{T} \times \overrightarrow{subj} \right) \quad (7)$$

**Linear regression** None of the above models create tensors that are fully populated: one or more dimensions will always remain empty. Following an idea first introduced by Baroni and Zamparelli (2010) for the creation of adjective matrices, Grefenstette et al. (2013) use linear regression in order to learn full tensors of order 3 for transitive verbs. Linear regression is a supervised method of learning, so it needs a number of exemplar data points. In the case of the adjective 'red', for example, we would need a set of the form $\langle\overrightarrow{car}, \overrightarrow{red\ car}\rangle$, $\langle\overrightarrow{shirt}, \overrightarrow{red\ shirt}\rangle$, $\langle\overrightarrow{shoe}, \overrightarrow{red\ shoe}\rangle$ and so on, where the second vector in each pair is the contextual vector of the whole phrase created exactly as if it were a single word. The goal of the learning process is to find the parameters $\overline{adj}^2$ and $\overrightarrow{b}$ such that:

$$\overrightarrow{adj\ noun} \approx \overline{adj}^2 \times \overrightarrow{noun} + \overrightarrow{b} \quad (8)$$

for all nouns modified by the specific adjective. In practice, the bias $\overrightarrow{b}$ is embedded in $\overline{adj}^2$, hence the above procedure provides us with a matrix for the adjective. One can generalize this procedure to tensors of higher order by proceeding step-wise, as done by Grefenstette et al. (2013). For the case of a transitive verb, they first use exemplar pairs of the form $\langle\overrightarrow{subj}, \overrightarrow{subj\ verb\ obj}\rangle$ to learn a matrix $\overline{verb\ obj}^2$ for the verb phrase; then, they perform a new training session with exemplars of the form $\langle\overrightarrow{obj}, \overline{verb\ obj}^2\rangle$, the result of which is an order 3 tensor for the verb.

## 6 Generic context-based disambiguation

In all of the models of Section 5, the training of a relational word tensor is based on the set of contexts where this word occurs. Hence, in these models the problem of creating disambiguated versions of tensors can be recasted to that of further breaking the set of contexts in a way that each subset reflects a different sense of the word in the corpus. If, for example, $S$ is the whole set of sentences for a word $w$ that occurs in the corpus under $n$ different senses, then the goal is to create $n$ subsets $S_1, \ldots S_n$ such that $S_1$ contains the sentences where $w$ appears under the first sense, $S_2$ the sentences where $w$ occurs under the second sense, and so on. Each one of these subsets can then be used to train a tensor for a specific sense of the target relational word.

Towards this purpose we use a variation of the effective procedure of Schütze (1998): first, each context for a target word $w_t$ is represented by a *context vector* of the form $\frac{1}{n}(\overrightarrow{w_1} + \ldots + \overrightarrow{w_n})$, where $\overrightarrow{w_i}$ is the lexical vector of some other word $w_i \neq w_t$ in the same context. Next, we apply a clustering method on this set of vectors in order to discover the latent senses of $w_t$. The assumption is that the contexts of $w_t$ will vary according to the specific sense this word is used: 'bank' as a financial institution should appear in quite different contexts than as land.

The above procedure will give us a number of clusters, each consisting of context vectors; we use the centroid of each cluster as a vectorial representation of the corresponding sense. So in our model each word $w$ is initially represented by a tuple $\langle\overrightarrow{w}, S\rangle$, where $\overrightarrow{w}$ is the lexical vector of the word as created by the usual distributional practice, and $S$ is a set of sense vectors (centroids of context vector clusters) produced by the above procedure. The disambiguation of a new word $w$ under a context $C$ can now be accomplished as follows: we create a context vector $\overrightarrow{c}$ for $C$ as above, and we compare it with every sense vector of $w$; the word is assigned to the sense corresponding to the closest sense vector. Specifically, if $S_w$ is the set of sense vectors for $w$, $\overrightarrow{c}$ the context vector for $C$, and $d(\overrightarrow{v}, \overrightarrow{u})$ our vector distance measure, the preferred sense $\hat{s}$ is given by:

$$\hat{s} = \underset{\overrightarrow{v_s} \in S_w}{\arg\min}\, d(\overrightarrow{v_s}, \overrightarrow{c}) \quad (9)$$

For the actual clustering step we follow the setting of Kartsaklis et al. (2013), which worked well in tasks very similar to ours. Specifically, we perform hierarchical agglomerative clustering (HAC) using Ward's method as the inter-cluster distance, while the distance between vectors is measured with Pearson's correlation.[3] In the above work, this configuration has been found to return the highest V-measure (Rosenberg and Hirschberg, 2007) on the noun set of SEMEVAL 2010 Word Sense Induction & Disambiguation Task (Manandhar et al., 2010). As context for a word, we consider the sentence in which this word occurs. The output of HAC is a *dendrogram* embedding all the possible partitionings of the

[3]Informal experimentation with more robust probabilistic techniques, such as Dirichlet process gaussian mixture models, revealed no significant benefits for our setting.

data. In order to select the optimal partitioning, we rely on the Caliński/Harabasz index (Caliński and Harabasz, 1974), also known as variance ratio criterion (VRC). VRC is calculated as the ratio of the sum of the inter-cluster variances over the sum of the intra-cluster variances, bearing the intuition that the optimal partitioning should be the one that results in the most compact and maximally separated clusters. We compute the VRC for a range of different partitionings (from 2 to 10 clusters) and keep the partitioning with the highest score.

## 7 Constructing unambiguous verb tensors

The procedure described in Section 6 provides us with $n$ clusters of context vectors for a target word. Since in our case each context vector corresponds to a distinct sentence, the output of the clustering scheme can also be seen as $n$ subsets of sentences, where the word appears under different senses. It is now quite straightforward to use this partitioning of the training corpus in order to learn unambiguous versions of verb tensors, as detailed below.

**Relational/Frobenius** Both the Relational and the Frobenius models use the same way of creating an initial verb matrix (Equation 3) which then they expand to a higher order tensor. Let $S_1 \dots S_n$ be the sets of sentences returned by the clustering step for a verb. Then, the verb tensor for the $i$th sense is:

$$\overline{verb}_i^2 = \sum_{s \in S_i} (\overrightarrow{subj_s} \otimes \overrightarrow{obj_s}) \tag{10}$$

where $subj_s$ and $obj_s$ refer to the subject/object pair that occurred with the verb in sentence $s$. This can be generalized to any arity $n$ as follows:

$$\overline{word}_i^n = \sum_{s \in S_i} \bigotimes_{k=1}^{n} \overrightarrow{arg_{k,s}} \tag{11}$$

where $arg_{k,s}$ denotes the $k$th argument of the target word in sentence $s$.

**Kronecker** For a given verb $v$ in a context $C$, let $\overrightarrow{v_i}$ be the sense vector of $v$ given $C$ corresponding to the sense $i$ returned by Equation 9. Then we have:

$$\overline{verb}_i^2 = \overrightarrow{v_i} \otimes \overrightarrow{v_i} \tag{12}$$

The generalized version to arity $n$ is given by:

$$\overline{word}_i^n = \bigotimes_{k=1}^{n} \overrightarrow{v_i} \tag{13}$$

**Linear regression** Creating unambiguous full tensors using linear regression is also quite straightforward. Let us assume again that the clustering step for a verb $v$ returns $n$ sets of sentences $S_1 \dots S_n$, where each sentence set corresponds to a different sense. Then, we have $n$ different regression problems, each one of which will be trained on exemplar pairs derived exclusively from the sentences of the corresponding set. This will result in $n$ verb tensors, which will correspond to the different senses of the verb. Generalization to higher arities is a straightforward extension of the step-wise process in Section 5 for transitive verbs.

## 8 Experiments

In this section we will test the effect of disambiguation on the models of Section 5 in a variety of tasks. Due to the significant methodological differences of the linear regression model from the other approaches and the variety of its set of parameters, we decided that it would be better if this was left as the subject of a distinct work.

**Experimental setting** We train our vectors using ukWaC (Ferraresi et al., 2008), a corpus of English text with 2 billion words (100m sentences). We use 2000 dimensions, with weights calculated as the ratio of the probability of the context word given the target word to the probability of the context word overall. The context here is a 5-word window on both sides of the target word. The vectors are disambiguated both syntactically and semantically: first, separate vectors have been created for different syntactic usages of the same word in the corpus; for example, the word 'book' has two vectors, one for its noun sense and one for its verb sense. Furthermore, each word is semantically disambiguated according to the method of Section 6.

**Models** We compare the tensor-based models of Section 5 with the multiplicative and additive models of Mitchell and Lapata (2008), reporting results for both ambiguous and disambiguated versions. For all the disambiguated models, the best sense for each word in the sentence or phrase is first selected by applying the procedure of Section 6 and Equation 9. If the model is based on a vector mixture, the sense vectors corresponding to these senses are multiplied or added to form the composite representation for the sentence or phrase. For the tensor-based models, the composite meanings are calculated according to the equations of Section 5, using verb tensors created by the procedures of Section 7. The semantic similarity of two phrases or sentences is measured as the cosine distance between their composite vectors. For models that return a matrix (e.g. Relational, Kronecker), the distance is based on the Frobenius inner product.

**Implementation details** Our code is mainly written in Python and C++, and for the actual clustering step we use the Python interface of the efficient FASTCLUSTER library (Müllner, 2013). In a shared 24-core Xeon machine with 72 GB of memory, and with a fair amount of parallelism applied, the average processing time per word was about 4 minutes; this is roughly translated to 12-13 hours of training on average per dataset.

## 8.1 Verb disambiguation task

Perhaps surprisingly, one of the most popular tasks for testing compositionality in distributional models is based on disambiguation. This task, originally introduced by Kintsch (2001), has been adopted by Mitchell and Lapata (2008) and others for evaluating the quality of composition in vector spaces. Given an ambiguous verb such as 'file', the goal is to find out to what extent the presence of an appropriate context will disambiguate its intended meaning. The context (e.g. a subject/object pair) is composed with two landmark verbs corresponding to the different senses ('smooth' and 'register') to create simple sentences. The assumption is that a good compositional model should be able to reflect that 'woman files application' is closer to 'woman registers application' than to 'woman smooths application'.

In this paper we test our models on two different datasets of transitive sentences, that of Grefenstette and Sadrzadeh (2011a) and Kartsaklis et al. (2013)[4]. Specific details about the creation of the datasets can be found in the above papers; for the purposes of the current work it is sufficient to mention that their main difference is that in the former the verbs and their alternative meanings have been selected automatically using the JCN metric of semantic similarity (Jiang and Conrath, 1997), while in the latter the selection was based on human judgements from the work of Pickering and Frisson (2001). So, while in the first dataset many verbs cannot be considered as genuinely ambiguous (e.g. 'say' with meanings *state* and *allege* or 'write' with meanings *publish* and *spell*), the landmarks in the second dataset correspond to clearly separated senses (e.g. 'file' with meanings *register* and *smooth* or 'charge' with meanings *accuse* and *bill*). Furthermore, subjects and objects of this latter case are modified by appropriate adjectives, overall creating a richer and more linguistically balanced dataset.

In both cases the evaluation methodology is the same: each entry of the dataset has the form $\langle$subject, verb, object, high-sim landmark, low-sim landmark$\rangle$. The context is combined with the verb and the two landmarks, creating three simple transitive sentences. The main-verb sentence is paired with both the landmark sentences, and these pairs are randomly presented to human evaluators, the duty of which is to evaluate the similarity of the sentences within a pair in a scale from 1 to 7. The scores of the compositional models are the cosine distances (or the Frobenius inner products, in the case of matrices) between the composite representations of the sentences of each pair. As an overall score for each model, we report its Spearman's $\rho$ correlation with the human judgements. Both datasets consist of 200 pairs of sentences (10 main verbs $\times$ 2 landmarks $\times$ 10 contexts).

**Results** The results for the G&S dataset are shown in Table 1.[5] The verbs-only model (BL) refers to a non-compositional evaluation, where the similarity between two sentences is solely based on the distance between the two verbs, without applying any compositional step with subject and object.

The tensor-based models present much better performance than the vector mixture ones, with the disambiguated version of the copy-object model significantly higher than the relational model. By design, the copy-object model retains more information about the objects; so this result confirms previous findings, that in this certain dataset the role of objects is more important than that of subjects (Kartsaklis et al., 2012). In general, the disambiguation step improves the results of all the tensor-based models except Kronecker; the effect is reversed for the vector mixture models, where the disambiguated versions present much worse performance (these

[4]This dataset has been created by Mehrnoosh Sadrzadeh in collaboration with Edward Grefenstette, but remained unpublished until (Kartsaklis et al., 2013).

[5]For all tables in this section, $\ll$ and $\gg$ denote highly statistically significant differences with $p < 0.001$.

|  | Model | Ambig. |  | Disamb. |
|---|---|---|---|---|
| BL | Verbs only | 0.198 | $\gg$ | 0.132 |
| M1 | Multiplicative | 0.137 | $\gg$ | 0.044 |
| M2 | Additive | 0.127 | $\gg$ | 0.047 |
| T1 | Relational | 0.219 | $<$ | 0.223 |
| T2 | Kronecker | 0.207 | $\gg$ | 0.061 |
| T3 | Copy-subject | 0.070 | $\ll$ | 0.122 |
| **T4** | **Copy-object** | **0.241** | $\ll$ | **0.262** |
|  | Human agreement | 0.599 |  |  |

Difference between T4 and T1 is s.s. with $p < 0.001$

Table 1: Results for the G&S dataset.

|  | Model | Ambig. |  | Disamb. |
|---|---|---|---|---|
| **BL** | **Verbs only** | **0.151** | $\ll$ | **0.217** |
| M1 | Multiplicative | 0.131 | $<$ | 0.137 |
| M2 | Additive | 0.085 | $\ll$ | 0.193 |
| T1 | Relational | 0.036 | $\ll$ | 0.121 |
| T2 | Kronecker | 0.159 | $<$ | 0.166 |
| T3 | Copy-subject | 0.035 | $\ll$ | 0.117 |
| T4 | Copy-object | 0.033 | $\ll$ | 0.095 |
|  | Human agreement | 0.383 |  |  |

Difference between BL and M2 is s.s. with $p < 0.001$

Table 2: Results for the Kartsaklis et al. dataset.

findings are further discussed in Section 9).

The result of disambiguation is clearer for the dataset of Kartsaklis et al. (Table 2). The longer context in combination with genuinely ambiguous verbs produces two effects: first, disambiguation is now helpful for all models, either vector mixtures or tensor-based; second, the disambiguation of just the verb (verbs-only model), without any interaction with the context, is sufficient to provide the best score (0.22) with a difference statistically significant from the second model (0.19 for disambiguated additive). In fact, further composition of the verb with the context decreases performance, confirming the results reported by Kartsaklis et al. (2013) for vectors trained using BNC. Given the nature of the specific task, which is designed around the ambiguity of the verb, this result is not surprising: a direct disambiguation of the verb based on the rest of the context should naturally constitute the best method to achieve top performance—no composition is necessary for this task to be successful.

However, when one *does* use a task like this in order to evaluate compositional models (as we do here and as is commonly the case), they implicitly correlate the strength of the disambiguation effect that takes place during the composition with the quality of composition, essentially assuming that the stronger the disambiguation, the better the compositional model that produced this side-effect. Unfortunately, the extent to which this assumption is valid or not is still not quite clear; the subject is addressed in more detail in (Kartsaklis et al., 2013). Keeping a note of this observation, we now proceed to examine the performance of our models in a task that does not use disambiguation as a criterion of composition.

## 8.2 Phrase/sentence similarity task

Our second set of experiments is based on the phrase similarity task of Mitchell and Lapata (2010). On the contrary with the task of Section 8.1, this one does not involve any assumptions about disambiguation, and thus it seems like a more genuine test of models aiming to provide appropriate phrasal or sentential semantic representations; the only criterion is the degree to which these models correctly evaluate the *similarity* between pairs of sentences or phrases. We work on the verb-phrase part of the dataset, consisting of 72 short verb phrases (verb-object structures). These 72 phrases have been paired in three different ways to form groups exhibiting various degrees of similarity: the first group contains 36 pairs of highly similar phrases (e.g. *produce effect-achieve result*), the pairs of the second group are of medium similarity (e.g. *write book-hear word*), while a last group contains low-similarity pairs (*use knowledge-provide system*). The task is again to compare the similarity scores given by the various models for each phrase pair with those of human annotators. Additionally to the verb phrases task, we also perform a richer version of the experiment using transitive sentences.

**Verb phrases** It can be shown that for simple verb phrases the relational model reduces itself to the copy-subject model; for both of these methods, the meaning of the verb phrase is calculated according to Equation 6. Furthermore, according to the copy-object model the meaning of a verb phrase computed by a verb matrix $\sum_{ij} v_{ij}(\overrightarrow{n_i} \otimes \overrightarrow{n_j})$ and an object vector $\sum_j o_j \overrightarrow{n_j}$ becomes:

$$\overline{verb\ object}^2 = \sum_{ij} v_{ij} o_j (\overrightarrow{n_i} \otimes \overrightarrow{n_j}) \tag{14}$$

Finally, the Kronecker model has no meaning for verb phrases, since the vector of a verb phrase will become $(\overrightarrow{v_s} \otimes \overrightarrow{v_s}) \times \overrightarrow{obj}$, which is equal to $\langle \overrightarrow{v_s} | \overrightarrow{obj} \rangle \overrightarrow{v_s}$, where $\langle \overrightarrow{v_s} | \overrightarrow{obj} \rangle$ denotes the inner product between vectors of verb and object. Hence, the

| | Model | Ambig. | | Disamb. |
|---|---|---|---|---|
| BL | Verbs only | 0.310 | $\ll$ | 0.420 |
| **M1** | **Multiplicative** | **0.315** | $\ll$ | **0.448** |
| M2 | Additive | 0.291 | $\ll$ | 0.436 |
| T1 | Rel./Copy-sbj | 0.340 | $\ll$ | 0.367 |
| T2 | Copy-object | 0.290 | $\ll$ | 0.393 |
| | Human agreement | 0.550 | | |

Difference between M1 and M2 is not s.s.
Difference between M1 and BL is s.s. with $p < 0.001$

Table 3: Results for the original M&L task.

meaning of a verb phrase becomes a scalar multiplication of the meaning of its verb. As a result, the cosine distance (used for measuring similarity) between the meanings of two verb phrases is reduced to the distance between the vectors of their verbs, completely dismissing the role of their objects.

Hence our models are limited to those of Table 3. The effects of disambiguation for this task are quite impressive: the differences between the scores of all disambiguated models and those of the ambiguous versions are highly statistically significant (with $p < 0.001$), while 4 of the 5 models present an improvement greater than 10 units of correlation. The models that benefit the most from disambiguation are the vector mixtures; both of these approaches perform significantly better than the best tensor-based model (copy-object). In fact, the score of M1 (0.45) is quite high, given that the inter-annotator agreement is 0.55 (best score reported by Mitchell and Lapata was 0.41 for their LDA-dilation model).

**Transitive sentences** The second part of this experiment aims to examine the extent to which the above picture can change for the case of text structures longer than verb phrases. In order to achieve this, we extend each one of the 72 verb phrases to a full transitive sentence by adding an appropriate subject such that the similarity relationships of the original dataset are retained as much as possible, so the human judgements for the verb phrase pairs could as well be used for the transitive cases. We worked pair-wise: for each pair of verb phrases, we first selected one of the 5 most frequent subjects for the first phrase; then, the subject of the other phrase was selected by a list of synonyms of the first subject in a way that the new pair of transitive sentences constitutes the least more specific version of the given verb-phrase pair. So, for example, the pair *produce effect/achieve result* became *drug produce effect/medication achieve result*, while the pair *pose problem/address question* became *study pose problem/paper address question*.[6]

The restrictions of the verb-phrase version do not hold here, so we evaluate on the full set of models (Table 4). Once more disambiguation produces better results in all cases, with highly statistically significant differences for all but one model. Furthermore, now the best score is delivered by one of the tensor-based models (Kronecker), with a difference *not* statistically significant from disambiguated additive. In any case, the result suggests that as the length of the text segments increases, the performance of vector mixtures and tensor-based models converges. Indeed, note how the performance of the vector mixture models are significantly decreased compared to the verb phrase task.

## 9 Discussion

The purpose of this work was twofold: our main objective was to investigate how disambiguation can affect the compositional models which are based on higher order vector spaces; a second, but not less important goal, was to compare this more linguistically motivated approach to the simpler vector mixture methods. Based on the experimental work presented here, we can say with enough confidence that disambiguation as an additional step prior to composition is indeed very beneficial for tensor-based models. Furthermore, our experiments confirm and strengthen previous work (Reddy et al., 2011; Kartsaklis et al., 2013) that showed better performance of disambiguated vector mixture models compared to their ambiguous versions. The positive effect of disambiguation is more evident for the vector mixture models (especially for the additive model) than for

[6]The dataset will be available at `http://www.cs.ox.ac.uk/activities/compdistmeaning/`.

| | Model | Ambig. | | Disamb. |
|---|---|---|---|---|
| BL | Verbs only | 0.310 | $\ll$ | 0.341 |
| M1 | Multiplicative | 0.325 | $\ll$ | 0.404 |
| M2 | Additive | 0.368 | $\ll$ | 0.410 |
| T1 | Relational | 0.368 | $\ll$ | 0.397 |
| **T2** | **Kronecker** | **0.404** | $<$ | **0.412** |
| T3 | Copy-subject | 0.310 | $\ll$ | 0.337 |
| T4 | Copy-object | 0.321 | $\ll$ | 0.368 |
| | Human agreement | 0.550 | | |

Difference between T2 and M2 is not s.s.

Table 4: Transitive version of M&L task.

the tensor-based ones. This is expected: composite representations created by element-wise operations are averages, and a prior step of disambiguation can make a great difference.

From a task perspective, the effect of disambiguation was much more definite in the phrase/sentence similarity task. This observation is really interesting, since the words of that dataset were *not* selected in order to be ambiguous in any way. The superior performance of the disambiguated models, therefore, implies that the proposed methodology can improve tasks based on phrase or sentence similarity *regardless* of the level of ambiguity in the vocabulary. For these cases, the proposed disambiguation algorithm acts as a fine-tuning process, the outcome of which seems to be always positive; it can only produce better composite representations, not worse. In general, the positive effect of disambiguation in the phrase/sentence similarity task is quite encouraging, especially given the fact that this task constitutes a more appropriate test for evaluating compositional models, avoiding the pitfalls of disambiguation-based experiments (as shortly discussed in Section 8.1).

For disambiguation-based tasks similar to those of Section 8.1, the form of dataset is very important; hence the inferior performance of disambiguated models in the G&S dataset, compared to the dataset of Kartsaklis et al.. In fact, the G&S dataset was the only one where disambiguation was not helpful for some cases (specifically, for vector mixtures and the Kronecker model). We believe the reason behind this lies in the fact that the automatic selection of landmark verbs using the JCN metric (as done with the G&S dataset) was not very efficient for certain cases. Note, for example, that the bare baseline of comparing just ambiguous versions of verbs (without any composition) in that dataset already achieves a very high correlation of 0.198 with human judgements (Table 1).[7]. This number is only 0.15 for the Kartsaklis et al. dataset, due to the more efficient verb selection procedure. In general, we consider the results gained by this latter experiment more reliable for the specific task, the successful evaluation of which requires genuinely ambiguous verbs.

The results are less conclusive for the second question we posed in the beginning of this section, regarding the comparison of the two classes of models. Despite the obvious benefits of the tensor-based approaches, this work suggests for one more time that vector mixture models might constitute a hard-to-beat baseline; similar observations have been made, for example, in the comparative study of Blacoe and Lapata (2012). However, when trying to interpret the mixing results regarding the effectiveness of the tensor-based models compared to vector mixtures, we need to take into account that the tensor-based models tested in this work were all "hybrid", in the sense that they all involved some element of point-wise operation; in other words, they constituted a trade-off between transformational power and complexity.

Even with this compromise, though, the study presented in Section 8.2 implies that the effectiveness of each method depends to some extent on the length of the text segment: when more words are involved, vector mixture models tend to be less effective; on the contrary, the performance of tensor-based models seems to be proportional to the length of the phrase or sentence—the more, the better. These observations comply with the nature of the approaches: "averaging" larger numbers of points results in more general (hence less accurate) representations; on the other hand, a larger number of arguments makes a function (such as a verb) more accurate.

[7]The reported number for this baseline by Grefenstette and Sadrzadeh (2011a) was 0.16 using vectors trained from BNC.

## 10 Conclusion and future work

In the present paper we showed how to improve a number of tensor-based compositional distributional models of meaning by introducing a step of disambiguation prior to composition. Our simple algorithm (based on the procedure of Schütze (1998)) creates unambiguous versions of tensors before these are composed with vectors of nouns in order to construct vectors for sentences and phrases. This algorithm is quite generic, and can be applied to any model that follows the tensor contraction process described in Section 4. As for future work, we aim to investigate the application of this procedure to the regression model of Grefenstette et al. (2013).

## Acknowledgements

We would like to thank Edward Grefenstette for his comments on the first draft of this paper, as well as the three anonymous reviewers for their fruitful suggestions. Support by EPSRC grant EP/F042728/1 is gratefully acknowledged by the authors.

## References

Baroni, M. and Zamparelli, R. (2010). Nouns are Vectors, Adjectives are Matrices. In *Proceedings of Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Blacoe, W. and Lapata, M. (2012). A comparison of vector-based representations for semantic composition. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 546–556, Jeju Island, Korea. Association for Computational Linguistics.

Bourbaki, N. (1989). *Commutative Algebra: Chapters 1-7*. Srpinger Verlag, Berlin/New York.

Caliński, T. and Harabasz, J. (1974). A Dendrite Method for Cluster Analysis. *Communications in Statistics-Theory and Methods*, 3(1):1–27.

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

Curran, J. (2004). *From Distributional to Semantic Similarity*. PhD thesis, School of Informatics, University of Edinburgh.

Ferraresi, A., Zanchetta, E., Baroni, M., and Bernardini, S. (2008). Introducing and evaluating ukWaC, a very large web-derived corpus of English. In *Proceedings of the 4th Web as Corpus Workshop (WAC-4) Can we beat Google*, pages 47–54.

Grefenstette, E., Dinu, G., Zhang, Y.-Z., Sadrzadeh, M., and Baroni, M. (2013). Multi-step regression learning for compositional distributional semantics. In *Proceedings of the 10th International Conference on Computational Semantics (IWCS 2013)*.

Grefenstette, E. and Sadrzadeh, M. (2011a). Experimental Support for a Categorical Compositional Distributional Model of Meaning. In *Proceedings of Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Grefenstette, E. and Sadrzadeh, M. (2011b). Experimenting with Transitive Verbs in a DisCoCat. In *Proceedings of Workshop on Geometrical Models of Natural Language Semantics (GEMS)*.

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

Jiang, J. and Conrath, D. (1997). Semantic similarity based on corpus statistics and lexical taxonomy. In *Proceedings on International Conference on Research in Computational Linguistics*, pages 19–33, Taiwan.

Kartsaklis, D., Sadrzadeh, M., and Pulman, S. (2012). A unified sentence space for categorical distributional-compositional semantics: Theory and experiments. In *Proceedings of 24th International Conference on Computational Linguistics (COLING 2012): Posters*, pages 549–558, Mumbai, India. The COLING 2012 Organizing Committee.

Kartsaklis, D., Sadrzadeh, M., and Pulman, S. (2013). Separating Disambiguation from Composition in Distributional Semantics. In *Proceedings of 17th Conference on Computational Natural Language Learning (CoNLL-2013)*, Sofia, Bulgaria.

Kintsch, W. (2001). Predication. *Cognitive Science*, 25(2):173–202.

Landauer, T. and Dumais, S. (1997). A Solution to Plato's Problem: The Latent Semantic Analysis Theory of Acquision, Induction, and Representation of Knowledge. *Psychological Review*.

Manandhar, S., Klapaftis, I., Dligach, D., and Pradhan, S. (2010). Semeval-2010 task 14: Word sense induction & disambiguation. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 63–68. Association for Computational Linguistics.

Manning, C., Raghavan, P., and Schütze, H. (2008). *Introduction to Information Retrieval*. Cambridge University Press.

Mitchell, J. and Lapata, M. (2008). Vector-based Models of Semantic Composition. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics*, pages 236–244.

Mitchell, J. and Lapata, M. (2010). Composition in distributional models of semantics. *Cognitive Science*, 34(8):1388–1439.

Müllner, D. (2013). fastcluster: Fast Hierarchical Clustering Routines for R and Python. *Journal of Statistical Software*, 9(53):1–18.

Pickering, M. and Frisson, S. (2001). Processing ambiguous verbs: Evidence from eye movements. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 27(2):556.

Pulman, S. (2013). Combining Compositional and Distributional Models of Semantics. In Heunen, C., Sadrzadeh, M., and Grefenstette, E., editors, *Quantum Physics and Linguistics: A Compositional, Diagrammatic Discourse*. Oxford University Press.

Reddy, S., Klapaftis, I., McCarthy, D., and Manandhar, S. (2011). Dynamic and static prototype vectors for semantic composition. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 705–713.

Rosenberg, A. and Hirschberg, J. (2007). V-measure: A conditional entropy-based external cluster evaluation measure. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 410–420.

Schütze, H. (1998). Automatic Word Sense Discrimination. *Computational Linguistics*, 24:97–123.