Decomposing Bilexical Dependencies into Semantic and Syntactic Vectors

Jeff Mitchell
mitchelljeff@hotmail.com

Abstract

Bilexical dependencies have been commonly used to help identify the most likely parses of a sentence. The probability of a word occurring as the dependent of a given head within a particular structure provides a measure of semantic plausibility that complements the purely syntactic part of the parsing model.

Here, we attempt to use the distributional information within these bilexical dependencies to construct representations that decompose into semantic and syntactic components. In particular, we compare two different approaches to composing vectors to explore how syntactic and semantic representations should interact within such a model.

Our results suggest a tensor product approach has advantages, which we believe could be exploited in making more effective use of the information captured in these bilexical dependencies.

1 Introduction

Using points within the geometry of a vector space to represent the way words are distributed across contexts has proven to be a fruitful tactic for many language processing tasks. For example, Landauer and Dumais (1997) projected raw tf-idf scores of occurrence across a set of documents down into lower dimensional vectors using a technique called singular value decomposition. The resulting semantic representations were then applied to semantic dismbiguation and to predict synonyms in a TOEFL test. Working instead with the linear structure of raw text, Collobert et al. (2011) trained a neural language model to induce word vectors in the hidden layer of their network. These versatile representations were then applied to a wide range of tasks including part-of-speech tagging, chunking, named entity recognition, and semantic role labeling.

Two key elements within any such approach to constructing representations are the contexts across which the distribution of a word is tracked and how vectors are constructed from these occurrences. Here, we investigate the construction of distributional representations from bilexical dependencies found in a parser and explore how such vectors can be decomposed into semantic and syntactic components.

Although, distributional approaches have commonly become most strongly associated with semantic representations and tasks, they have also seen applications to syntax. In fact, distributional analysis was first applied by linguists to syntactic categories rather than the representation of meaning and Ross (1972) presented a continuous, or at least graded, conception of syntax long before the recent surge of interest in vectorial approaches to semantics.

Practical applications of these distributional techniques to syntactic problems have included work on the induction (Brown et al., 1992) or learning (Mintz, 2003) of categories and the computational problems of tagging (Tsuboi, 2014) and parsing (Socher et al., 2013). The latter problem of parsing brings to the foreground the question of how syntactic and semantic representations relate to and interact with each other, as the optimal parse must maximise both syntactic and semantic plausibility, in an integrated structure.

An unmodified PCFG, modelling just the dependencies between syntactic categories, is generally inadequate to derive robust parses, and lexicalisation is commonly used to enhance such models. In particular, bilexical dependencies introduce

Proceedings of the 1st Workshop on Representation Learning for NLP, pages 127–136,
Berlin, Germany, August 11th, 2016.

a measure of the plausibility of combining specific heads and dependents within the possible syntactic structures.

These dependencies contain much the same information used by Lin (1998) and Padó and Lapata (2007) to construct semantic representations, and we can easily see that the plausibility of *cake* as an object of *bake*, *eat* and *regret* tells us something about the semantic properties of *cake*. Nonetheless, these dependencies contain substantial quantities of syntactic information, too. The dependencies observed for *cake* and *eat*, for example, are substantially different because the former is a noun while the latter is a verb.

However, the sparsity of the resulting counts can mean these dependencies may contribute little to parser performance, particulaly on out of domain data. One solution, proposed by Rei and Briscoe (2013), is to smooth the bilexical dependencies using a similarity measure. For example, if counts for *publication* as an object of *read* are lacking we might instead leverage the similarity of *publication* to *book* to use the counts for *book* as an object of *read* to make a reasonable inference about the unseen dependency. Alternatively, we might try to use some form of dimensionality reduction to smooth out the sparsity.

Levy and Goldberg (2014) use a modified version of word2vec (Mikolov et al., 2013) to induce 300 dimensional representations from word distributions across 900,000 dependency contexts. They find that these word vectors capture a form of functional similarity, with the closest words in the space typically being cohyponyms within the same syntactic class. This syntactic specificity is not particularly surprising, as we would expect the strongest effects within these dependencies to relate to the syntactic class of a word - e.g. only a noun can be the subject of a verb - with semantic factors having a weaker influence merely on word choice within the correct syntactic class.

In this paper, we will consider a couple of approaches that attempt to separate out semantic and syntactic components of the dependencies, boosting performance on both types of task. One popular method of boosting semantic performance has been to ignore or average over syntactic structure. By treating the context a target occurs in as a bag-of-words (Landauer and Dumais, 1997; Blei et al., 2003; Mikolov et al., 2013), syntactic information is washed out and semantic information is retained. Conversely, distributional approaches to syntactic tasks typically make use of the sequential structure contained in bigrams (Brown et al., 1992; Clark, 2003) or longer n-grams (Mintz, 2003; Redington et al., 1998).

Recent work, (Mitchell, 2013; Mitchell and Steedman, 2015), has attempted to use both types of information in a single model that decomposes representations into syntactic and semantic components. An open question, however, is the most effective way of forming these combined representations. Mitchell and Steedman (2015) explicitly employ a direct sum - i.e. concatenation - of semantic and syntactic vectors. On the other hand, the multiplicative combination used by Mitchell (2013) is much closer to a tensor product formulation.

Griffiths et al. (2005) also pursue the representation of semantics and syntax in a single distributional model. They integrate a topic model and HMM to produce a model of the sequential structure of raw text in which each word is either semantic - chosen by the topic model to fit the long range semantic context - or syntactic - chosen to fit the short range dependencies of the HMM. This either/or assumption is rejected by Boyd-graber and Blei (2009) who moreover work with parsed sentences, rather than raw text. In this model, each word is chosen based on a product of a document topic distribution and a set of syntactic transition probabilities, determined top-down within the parse tree. Socher et al. (2010) are also concerned with inducing distributional representations within the stucture of parse trees. Their neural network model composes vectors recursively from the bottom up to represent possible phrases and from those representations computes how likely each is to be a valid constituent.

Although, parsing may seem, initially, to be the ideal task in which to explore the relationship between semantic and syntactic representations, the complexity of a working system - which Bikel (2004a) describes as *an intractable behemoth* - makes it difficult to isolate and investigate just this question on its own. Parser performance depends on a multitiude of interacting components, and could only obliquely produce insights into the merit of the approaches to representation we want to consider here.

Instead, we follow the advice of Bikel (2004b) to *treat the model as data*, and make direct eval-

uations of distributional representations induced from the parameters of a wide coverage model. We focus in on just the bilexical dependencies within the BLLIP parser (Charniak and Johnson, 2005; McClosky et al., 2006) and explore models of these parameters in which the representation for each word decomposes into a semantic and a syntactic vector. We evaluate both a direct sum and a tensor product approach to this decomposition of the representation space and find that the latter has advantages.

In the next section, we describe the BLLIP parser and the data we extract from the wide coverage model of McClosky et al. (2006). Then in Sections 3 and 4 we describe the models applied to this data and their evaluation. Finally, we present our results and conclusions in Sections 5 and 6.

2 BLLIP Parser

The BLLIP parser (Charniak and Johnson, 2005) uses a two stage approach, based on discriminative reranking applied to candidate parses produced by a generative lexicalised PCFG. That first stage of the model takes inspiration from loglinear models to express the overall parse probability in terms of a product of multiplicative factors.

Here we are specifically interested in the bilexical dependencies, which are stored in the model as a probability, $p(d|h,t)$, of a dependent, d, given a head, h within some tree structure, t, along with a count for the occurence of that head-tree combination. The tree structure, t, is only specified in terms of the tags on the head and dependent leaves, the node from which they branch, and the category of the dependent branch below that point. Thus, many distinct trees are collapsed into a single class. For example, the model fails to distinguish between subjects and objects of a verb.

McClosky et al. (2006) expanded the domain of the standard Penn Treebank (Marcus et al., 1993) trained BLLIP model, applying self-training to 2.5M sentences from the NANC corpus (Graff, 1995). The resulting model has a large vocabulary, with reliable estimates of probabilities for many words, which provides a useful basis for our investigations.

We extract the bilexical dependencies and head-tree counts from the model file, replacing words that occur less than 5 times with an $\langle UNK \rangle$ tag, and also excluding any word that does not occur in both the head and dependent positions. The head-tree contexts are similarly filtered, with items that occur less then 5 times replaced with a dummy catch-all context.

3 Models

The models we discuss here derive a probability of a dependent word, d, within the context of a tree, t, with a head, h, in terms of latent variables, e.g. i, j, k. So, the simplest model we will consider has the form:

$$p(d|h,t) = \sum_i p(d|i)p(i|h,t) \tag{1}$$

It will be useful, notationally and conceptually, to think of these models in terms of vectors. The equation above already has a superficial similarity to a dot product, being a sum over a series of products.

We can rewrite this:

$$p(d|h,t) = p(d) \sum_i p(i) \frac{p(i|d)}{p(i)} \frac{p(i|h,t)}{p(i)} \tag{2}$$

We will think of $\frac{p(i|x)}{p(i)}$ as being the components, v_i^x, of a vector, $\mathbf{v}^x$, representing x and define an inner product in terms of a weighted [1] sum of component products as follows:

$$\mathbf{u} \cdot \mathbf{v} = \sum_i \lambda_i u_i v_i \tag{3}$$

Taking $\lambda_i = p(i)$, we can rewrite Eq. 2 as follows:

$$p(d|h,t) = p(d)\mathbf{v}^d \cdot \mathbf{v}^{ht} \tag{4}$$

More generally, this model form will need to include normalisation:

$$p(d|h,t) = \frac{p(d)\mathbf{v}^d \cdot \mathbf{v}^{ht}}{N(h,t)} \tag{5}$$

One of the benefits of this model form is that the normalising constant for each head-tree can be calculated fairly efficiently in terms of a single inner product.

$$N(h,t) = \mathbf{n} \cdot \mathbf{v}^{ht} \tag{6}$$

Here, $\mathbf{n}$ is a sum over all dependent probabilities and vectors.

[1] The use of such a weighting implies we are working with unnormalised basis vectors.

$$\mathbf{n} = \sum_d p(d)\mathbf{v}^d \quad (7)$$

Given this model form, we must then specify how the vectors $\mathbf{v}$ are constructed. In particular, if our representations are based on semantic vectors, $\mathbf{a}$, and syntactic vectors, $\mathbf{b}$, then we must decide how these are to be combined. One obvious choice is between a direct sum (Eq. 8) and a tensor product (Eq. 9).

$$\bar{\mathbf{v}} = \bar{\mathbf{a}} \oplus \bar{\mathbf{b}} \quad (8)$$

$$\tilde{\mathbf{v}} = \tilde{\mathbf{a}} \otimes \tilde{\mathbf{b}} \quad (9)$$

Although both these constructions consist of a combination of vectors, $\mathbf{a}$ and $\mathbf{b}$, the actual vectors induced during EM training will inevitably turn out to be substantially different for each approach. In fact, our purpose is precisely to investigate how this choice of combination affects the representations induced in our trained models. We therefore notationally distinguish the two approaches: using a bar, $\bar{\mathbf{v}}$, to indicate direct sum vectors and a tilde, $\tilde{\mathbf{v}}$, for tensor product vectors. However, we will also employ bare symbols without bar or tilde when discussing general properties across both types of structure.

So, if $\mathbf{a}$ and $\mathbf{b}$ are m and n dimensional vectors respectively, then Eq. 8 corresponds to forming the $n + m$ dimensional concatenation of those vectors, while Eq. 9 results in the $n \times m$ dimensional vector of all products of their components. From a probabilistic perspective, a reasonable interpretation would be that our models using the direct sum representations in Eq. 8 assume that the dependencies between head and dependent are either syntactic or semantic, whereas tensor product models, Eq. 9, assume that each word has both semantic and syntactic characteristics.

Given some method for combining vectors $\mathbf{a}$ and $\mathbf{b}$, we also need to specify the form of their components. In particular, we are interested here in separating semantic and syntactic dependencies.

Mitchell (2013) and Mitchell and Steedman (2015) both exploit word order to decompose representations into semantic and syntactic components, with semantic dependencies being modelled in terms of a similarity measure that is independent of word order, while the syntactic part of the model captures sequential information. However, the bilexical dependencies we are working with here do not explicitly relate to surface word order. Nonetheless, the relationship is still directed, distinguishing a head and a dependent, and we can exploit this directedness to define a symmetric semantic component and an asymmetric syntactic component.

In each of the models below, any word has a single semantic vector, $\mathbf{a}$, whether it occurs in head or dependent position, with components a_j given by:

$$a_j = \frac{p(j|w)}{p(j)} \quad (10)$$

Ignoring the syntactic component of the model for a moment, we can define a semantics only model (leaving out the normalising constant for brevity):

$$p(d|h,t) \propto p(d) \sum_j p(j) \frac{p(j|w^d)}{p(j)} \frac{p(j|w^h)}{p(j)} = p(d)\mathbf{a}^d \cdot \mathbf{a}^h \quad (11)$$

This employs an inner product defined by $\mathbf{a}^d \cdot \mathbf{a}^h = \sum_j \lambda_j a_j^d a_j^h$ with $\lambda_j = p(j)$. However, this semantic only model ignores the tree t the words occur in and gives words the same representations whether they occur in head or dependent position. It is therefore symmetric in relation to these roles, and we can think of this model as capturing what head and dependent have in common.

In contrast, there are two forms of syntactic vectors, $\mathbf{b}^d$ and $\mathbf{b}^{ht}$, distinguishing between dependents and the head-tree contexts they occur in, with components given by:

$$b_k^{ht} = \frac{p(k|h,t)}{p(k)} \quad (12)$$

$$b_k^d = \frac{p(k|d)}{p(k)} \quad (13)$$

Again we can ignore the other part of the model and consider this part on its own:

$$p(d|h,t) \propto p(d) \sum_k p(k) \frac{p(k|d)}{p(k)} \frac{p(k|h,t)}{p(k)} = p(d)\mathbf{b}^d \cdot \mathbf{b}^{ht} \quad (14)$$

This is exactly equivalent to the simple model Eq. 4 above.

Taking the direct sum approach first (Eq. 8), we concatenate the semantic vectors, $\bar{\mathbf{a}}$, and syntactic vectors, $\bar{\mathbf{b}}$, to form a combined vector, $\bar{\mathbf{v}}$, with indices ranging over both j and k.

$$\bar{v}_i = \begin{cases} \bar{a}_j & \text{if } i = j; \\ \bar{b}_k & \text{if } i = k. \end{cases} \quad (15)$$

Inner products of such vectors will consist of a sum over the j component products followed by a sum over the k component products, with some appropriate weighting. The simplest model having this structure is an interpolation of the two models above (Eq. 11 and Eq. 14) with proportions q_a and q_b.

$$p(d|ht) \propto q_a p(d) \bar{\mathbf{a}}^d \cdot \bar{\mathbf{a}}^h + q_b p(d) \bar{\mathbf{b}}^d \cdot \bar{\mathbf{d}}^{ht} \quad (16)$$

In terms of the model form of Eq. 5, this is equivalent to defining an inner product on the direct sum vectors, $\bar{\mathbf{v}}^d \cdot \bar{\mathbf{v}}^{ht}$, with weightings λ of $q_a p(j)$ and $q_b p(k)$ for the two sets of components respectively.

$$\bar{\mathbf{v}}^d \cdot \bar{\mathbf{v}}^{ht} = \begin{array}{c} q_a \sum_j p(j) \frac{p(j|w^d)}{p(j)} \frac{p(j|w^h)}{p(j)} \\ + \\ q_b \sum_k p(k) \frac{p(k|d)}{p(k)} \frac{p(k|h,t)}{p(k)} \end{array} \quad (17)$$

For the tensor product model (Eq. 9), the indices of the combined vector, $\tilde{\mathbf{v}}$, range over all combinations of j and k.

$$\tilde{v}_{jk} = \tilde{a}_j \tilde{b}_k \quad (18)$$

The components of $\tilde{\mathbf{v}}^{ht}$ are then given by products of terms, which suggests conditional independence of j and k on ht.

$$\tilde{v}_{jk} = \frac{p(k|h,t)}{p(k)} \frac{p(j|h)}{p(j)} = \frac{p(j,k|h,t)}{p(j)p(k)} \quad (19)$$

Making a similar assumption of conditional independence in relation to $\tilde{\mathbf{v}}^d$ is enough to derive a model for $p(d|h,t)$.

$$\begin{aligned} p(d|h,t) &= \sum_{jk} p(d|j,k) p(j,k|h,t) \\ &\propto p(d) \sum_{jk} \frac{p(j|d)p(k|d)}{p(j,k)} p(j|h) p(k|h,t) \end{aligned} \quad (20)$$

This can be put into the form of Eq. 5 by defining the inner product $\tilde{\mathbf{v}}^d \cdot \tilde{\mathbf{v}}^{ht}$ as follows:

$$\tilde{\mathbf{v}}^d \cdot \tilde{\mathbf{v}}^{ht} = \sum_{j,k} \lambda_{jk} \tilde{a}_j^d \tilde{b}_k^d \times \tilde{a}_j^h \tilde{b}_k^{ht} =$$

$$\sum_{j,k} \frac{p(j)^2 p(k)^2}{p(j,k)} \left(\begin{array}{c} \frac{p(j|w^d)}{p(j)} \frac{p(k|d)}{p(k)} \\ \times \\ \frac{p(j|w^h)}{p(j)} \frac{p(k|h,t)}{p(k)} \end{array} \right) \quad (21)$$

Here, the weighting $\lambda_{jk} = \frac{p(j)^2 p(k)^2}{p(j,k)}$ is based on the assumption that j and k are both conditionally independent of d and ht.

As described above, the tensor product of a pair of vectors (Eq. 9) of dimension m and n produces a vector of dimension $n \times m$. However, these vectors only form a subset of the full $n \times m$ dimensional space. Moreover, the form of the model in both the direct sum and tensor product cases assumes that the semantic relation of head and dependent is independent of the syntactic relation. That is, we employ the same semantic vectors to represent head and dependent irrespective of the tree they occur in. We could begin to address both these issues by considering representations that lie in the full $n \times m$ dimensional tensor product space. This would essentially allow us to represent the dependence of semantic content on syntactic context. However, for now we restrict ourselves to the models described above.

We calculate the cross-entropy between the model and the BLLIP bilexical dependencies for each head-tree context and our objective function is then an average of these values, weighted by the occurence of that context. Training maximises this measure over 200 iterations of the EM algorithm.

4 Evaluation

We evaluate our models in a number of ways. We assess the quality of the word representations in terms of two similarity tasks on the semantic vectors, **a**, and a POS induction task on the syntactic vectors, **b**. In addition, both these tasks are applied to the raw data and to the vectors induced by the undecomposed models, Eq. 2 and Eq. 11. We also investigate the ability of our models to differentiate semantically and syntactically implausible adjective-noun constructions. Finally, we list a sample of nearest neighbours to allow a qualitative insight into the best performing model.

Our semantic similarity tasks are based on the ratings in two datasets, on both of which we evaluate our models using Spearman correlation. The first is the WordSim353 dataset (Finkelstein et al., 2002) containing ratings from 16 participants between pairs of nouns. The second dataset contains similarity ratings for noun-verb pairs (Mitchell, 2013). The former measures the ability of the model to capture semantic similarity within a POS class, while the latter tells about its representation of similarity across classes. This cross-class measure is useful in determining how effective the model has been in separating semantic from syntactic information. A model that bundles both into a single representation may identify the similarity in *disappear-vanish* but will typically fail to make the same judgement about *disappearance-vanish*. Making that judgement requires ignoring the syntactic difference between nouns and verbs, which we achieve in our models by representing that information separately.

Our syntactic task is POS induction. We cluster the vocabulary into 45 classes using k-means, and evaluate in terms of the many-to-one measure using the PTB POS classes as a gold standard. Although POS class information is already present in the bilexical dependency data, we use this task as a means of determining the quality of syntactic information contained in the vectors, rather than as an example of a practical application.

We then examine how our models differentiate semantic and syntactic plausibility. Our semantic plausibility dataset is constructed by combining a set of food nouns (e.g. *milk*, *meat*, *bread*, etc.) with either food appropriate adjectives (e.g. *hot*, *bitter*, *sweet*, etc.) or implausible political adjectives (e.g. *bipartisan*, *legislative*, *constitutional*, etc.). To create an equivalent syntactic plausibility dataset we combine common singular and plural nouns (e.g. *year* - *years*, *player* - *players*, etc.) with the modifier *several*. In each case, we calculate a semantic plausibility ($\mathbf{a}^d \cdot \mathbf{a}^h = \sum p(j) a_j^d a_j^h$) and a syntactic plausibility ($\mathbf{b}^d \cdot \mathbf{b}^{ht} = \sum p(k) b_k^d b_k^{ht}$) for the resulting adjective-noun phrase. Comparing the distribution of these measures in the high and low plausibility cases allows us to investigate further the extent to which the model separates semantic and syntactic dependencies.

Finally, we evaluate the best performing model - based on a tensor product of vectors - qualitatively by examining the closest neighbours of set of nouns, adjectives and verbs.

5 Results

Table 1 gives the correlations and many-to-one measures for the raw data, the simple undecomposed model (Eq. 2), the symmetric undecomposed model (Eq. 11), the direct sum model (Eq. 8) and the tensor product model (Eq. 9). Looking at the first two rows of the table, to compare the raw data to the simple model, we can see that the latter outpeforms the fomer on the POS clustering task, but is worse on the semantic similarity tasks. The improvement in performance on the clustering task can probably be put down to the excessive dimensionality (= number of head-tree contexts) of the input space in the case of the raw data. Reduction of this space using a latent variable model appears to make the clustering more effective. On the other hand, achieving this dimensionality reduction requires preserving the strongest, typically syntactic, dependencies and discarding weaker, frequently semantic, dependencies with the result that performance on semantic similarity tasks degrades. The predicted noun-verb similarities for both approaches is only weakly correlated with the human ratings, which we ascribe to the fact that neither model has a mechanism for finding the commonalities between words found in distinct sets of syntactic contexts.

The undecomposed symmetric model, $\mathbf{a}$, produces a better performance on the semantic tasks, but is worse on the syntactic task. This is not surprising, as the form of this model ignores the difference between heads and dependents, essentially treating the dependencies as a bag-of-words.

Both the direct sum and tensor product models also contain a similarity based component. However, only the tensor product model achieves an improvement in performance over the undecomposed models. In fact, this model outperforms the other models on all three measures, including achieving a reasonable level of correlation on the noun-verb dataset. This difference between the two decomposed models can be related to the fact that the form of the tensor product assumes that a dependent should be semantically *and* syntactically appropriate to its head-tree context, while the direct sum model uses an *or* condition between the two parts of the model.

Figures 1 and 2 present the results of exper-

Model	NV	WS353	MTO
raw	0.15	0.37	0.39
$\mathbf{v}$	-0.06	0.22	0.73
$\mathbf{a}$	0.24	0.42	0.42
$\bar{\mathbf{a}} \oplus \bar{\mathbf{b}}$	0.03	0.17	0.61
$\tilde{\mathbf{a}} \otimes \tilde{\mathbf{b}}$	0.38	0.49	0.74

Table 1: Correlations of model cosines with human similarity ratings on the noun-verb (NV) and WordSim353 (WS353) datasets, alongside many-to-one (MTO) measures of cluster quality on the POS clustering task, for the raw data (raw), the simple undecomposed model ($\mathbf{v}$), the symmetric undecomposed model ($\mathbf{a}$), the direct sum model ($\bar{\mathbf{a}} \oplus \bar{\mathbf{b}}$) and the tensor product model ($\tilde{\mathbf{a}} \otimes \tilde{\mathbf{b}}$).

iments on how these representations predict the plausibility of various adjective-noun phrases. In particular, these boxplots give an insight into the ability of these models to differentiate semantically from syntactically implausible constructions. Each plot contrasts the distribution of a logarithm of a dot product for high and low plausibility items. This dot product is either taken of semantic vectors, $\mathbf{a}$, or syntactic vectors, $\mathbf{b}$, providing a measure of, respectively, semantic and syntactic plausibility as predicted by the model. The results for each model are organised into a 2×2 array of plots, with the left hand column relating to the syntactic task and the right hand column to the semantic task.

Examining the results for the tensor product model in Figure 1 first, we find that of the syntactic plots, 1a and 1c, the contrast between high and low plausibility items is greatest for the $\ln(\bar{\mathbf{b}} \cdot \tilde{\mathbf{b}})$ measure. This indicates that these vectors have captured more of the syntactic information necessary to identify phrases such as *several year* as implausible. In contrast, the semantic plots, 1b and 1d, show the reverse pattern. There, it is the $\ln(\tilde{\mathbf{a}} \cdot \tilde{\mathbf{a}})$ measure which shows the largest contrast between high and low plausibility items. Thus, the difference in plausibility between *hot bread* and *bipartisan bread* is more effectively captured in the $\tilde{\mathbf{a}}$ vectors.

Turning to the results for the direct sum model in Figure 2, this differentiation between semantic and syntactic plausibility is no longer as clear, and the largest contrast between high and low plausibility items is always found in the $\ln(\bar{\mathbf{b}} \cdot \bar{\mathbf{b}})$ measure. Specifically, in the plots for the semantic task, 2b and 2d, the $\ln(\bar{\mathbf{a}} \cdot \bar{\mathbf{a}})$ measure does not produce a convincingly smaller prediction for the low plausibility items. In other words, the $\bar{\mathbf{a}}$ vectors do not contain the semantic information needed to identify the implausibility of phrases such as *bipartisan bread* or *constitutional milk*.

Word	$\tilde{\mathbf{a}} \otimes \tilde{\mathbf{b}}$	$\tilde{\mathbf{a}}$	$\tilde{\mathbf{b}}$
black	khaki	hispanic	female
	baggy	latino	decrepit
	cashmere	midterm	handmade
	plaid	D.C.	antique
	lace	Merle	year-old
political	tribal	cultural	biomedical
	tax-and-spend	favoritism	mechanical
	populist	sect	sole
	cultural	dissidents	mystical
	staunch	enlightenment	forensic
found	discovered	evidence	unveiled
	examined	takers	snared
	pleaded	fossils	rejected
	revealed	researchers	knocked
	flagged	conclusive	backfired
help	blame	strain	permit
	reprimand	psychological	resolve
	inflict	blisters	nudge
	relieve	suffering	laugh
	prevent	suffers	jump-start
company	firm	manufacturer	think-tank
	insurer	maker	nobody
	unit	pharmacuetical	everybody
	corporation	conglomerate	everyone
	consortium	subsidiary	foreman
game	opener	missed	speech
	finale	preseason	rotation
	rout	opener	shootout
	rematch	games	balloting
	tournament	NFC	opener

Table 2: Nearest neighbours within the full tensor product space ($\tilde{\mathbf{a}} \otimes \tilde{\mathbf{b}}$) and its semantic ($\tilde{\mathbf{a}}$) and syntactic ($\tilde{\mathbf{b}}$) components for a sample of adjectives, verbs and nouns.

Thus, the tensor product space appears to be most effective in separating semantic and syntactic information and its structure can be understood more concretely in terms of the sample of nearest neighbours shown in Table 2. Taking the adjective *black* as an example, the first column, $\tilde{\mathbf{a}} \otimes \tilde{\mathbf{b}}$, lists its nearest neighbours within the full tensor prod-

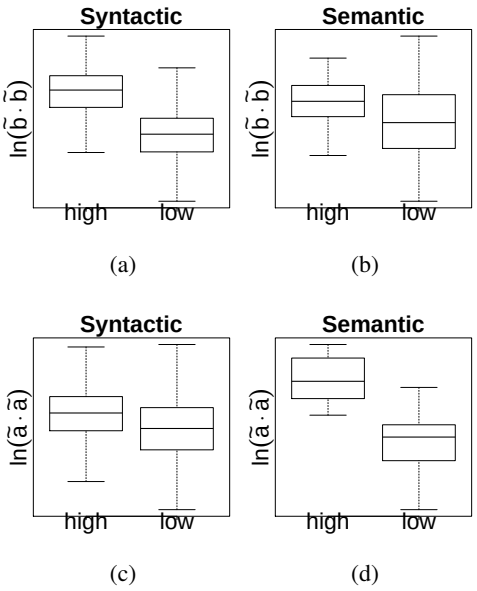

Figure 1: Boxplots of plausibility factors for tensor product representations on syntactic and semantic tasks.

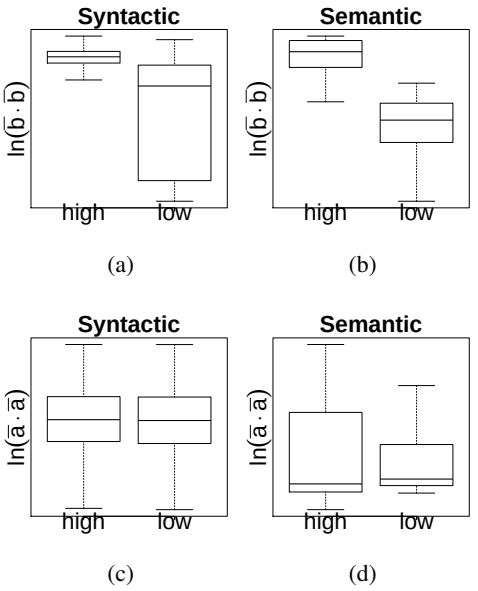

Figure 2: Boxplots of plausibility factors for direct sum representations on syntactic and semantic tasks.

uct space, which appear to be other descriptors of material appearance or structure. In contrast, the neighbours within the semantic space, $\tilde{\mathbf{a}}$, listed in the second column, seem to be words with cultural or political associations to *black*, while the syntactic neighbours in the third column, $\tilde{\mathbf{b}}$, are other adjectives drawn from a much wider domain.

6 Conclusions

We have shown that the bilexical dependencies within a parser capture useful semantic information, and also that it is possible to, at least partially, begin to separate out this semantic information from the syntactic information. Our experiments with vectors based on ratios of probabilities suggest that a tensor product approach to decomposing the space of representations has advantages over a direct sum approach. While the latter is conceptually simpler, being just a concatenation of the two vectors, the resulting model corresponds to an assumption that semantic and syntactic dependencies are disjoint, i.e. that the relationship between head and dependent is *either* semantic *or* syntactic. In contrast, the tensor product approach leads to a model in which a dependent must be syntactically *and* semantically appropriate to the context of tree and head word, and this seems to be more effective in practice.

These conclusions apply only to the ratio of probabilities type vectors that were investigated here. Log-linear vectors, as produced by neural network models, are likely to show substantially different behaviours. In fact, Mitchell and Steedman (2015) have shown that a direct sum approach can be effective for this type of model. Future work should investigate tensor product models in this setting.

Furthermore, there are theoretical reasons to pursue the tensor product approach further. While the models considered here are based on combining separate, independent semantic and syntactic vectors, the tensor product approach also allows us to consider the interaction of the two components. The direct sum approach, on the other hand, is less expressive.

In addition, implementation of parsers based on these representations may also be a fertile direction for future work. Our results suggest the techniques we investigated are effective in constructing semantic representations. We would also like to know whether capturing that semantic information effectively has benefits in modelling the overall probability of the whole dependency. However, our initial investigations suggest the syntactic part of the dependency needs a more sophisticated approach.

References

Daniel Bikel. 2004a. A distributional analysis of a lexicalized statistical parsing mode. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 182–189.

Daniel M. Bikel. 2004b. *On the parameter space of generative lexicalized statistical parsing models*. Ph.D. thesis, University of Pennsylvania, Philadelphia, PA, USA.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, March.

Jordan L. Boyd-graber and David M. Blei. 2009. Syntactic topic models. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems 21*, pages 185–192. Curran Associates, Inc.

Peter F. Brown, Peter V. deSouza, Robert L. Mercer, Vincent J. Della Pietra, and Jenifer C. Lai. 1992. Class-based n-gram models of natural language. *Comput. Linguist.*, 18(4):467–479, December.

Eugene Charniak and Mark Johnson. 2005. Coarse-to-fine n-best parsing and maxent discriminative reranking. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 173–180, Ann Arbor, Michigan. Association for Computational Linguistics.

Alexander Clark. 2003. Combining distributional and morphological information for part of speech induction. In *Proceedings of the tenth Annual Meeting of the European Association for Computational Linguistics (EACL)*, pages 59–66.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *J. Mach. Learn. Res.*, 12:2493–2537, November.

Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. 2002. Placing search in context: The concept revisited. *ACM Trans. Inf. Syst.*, 20(1):116–131, January.

David Graff. 1995. North american news text corpus. LDC95T21.

Thomas L. Griffiths, Mark Steyvers, David M. Blei, and Joshua B. Tenenbaum. 2005. Integrating topics and syntax. In *In Advances in Neural Information Processing Systems 17*, pages 537–544. MIT Press.

Thomas K Landauer and Susan T. Dumais. 1997. A solution to platos problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological Review*, 104(2):211–240.

Omer Levy and Yoav Goldberg. 2014. Dependency-based word embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL, Volume 2: Short Papers*, pages 302–308, Baltimore, MD, USA.

Dekang Lin. 1998. An information-theoretic definition of similarity. In *Proceedings of the Fifteenth International Conference on Machine Learning*, ICML '98, pages 296–304, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of english: The penn treebank. *Comput. Linguist.*, 19(2):313–330, June.

David McClosky, Eugene Charniak, and Mark Johnson. 2006. Effective self-training for parsing. In *Proceedings of the Main Conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, HLT-NAACL '06, pages 152–159, New York, New York. Association for Computational Linguistics.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 746–751, Atlanta, Georgia, June. Association for Computational Linguistics.

Toben H. Mintz. 2003. Frequent frames as a cue for grammatical categories in child directed speech. *Cognition.*, 90(1):91–117.

Jeff Mitchell and Mark Steedman. 2015. Orthogonality of syntax and semantics within distributional spaces. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1301–1310, Beijing, China, July. Association for Computational Linguistics.

Jeff Mitchell. 2013. Learning semantic representations in a bigram language model. In *Proceedings of the 10th International Conference on Computational Semantics (IWCS 2013) – Short Papers*, pages 362–368, Potsdam, Germany, March. Association for Computational Linguistics.

Sebastian Padó and Mirella Lapata. 2007. Dependency-based construction of semantic space models. *Comput. Linguist.*, 33(2):161–199, June.

Martin Redington, Nick Chater, and Steven Finch. 1998. Distributional information: A powerful cue for acquiring syntactic categories. *Cognitive Science*, 22(4):425–469.

Marek Rei and Ted Briscoe. 2013. Parser lexicalisation through self-learning. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 391–400, Atlanta, Georgia, June. Association for Computational Linguistics.

John R. Ross. 1972. The category squish: Endstation Hauptwort. In *Papers from the Eighth Regional Meeting*, pages 316–328, Chicago. Chicago Linguistic Society.

Richard Socher, Christopher D. Manning, and Andrew Y. Ng. 2010. Learning Continuous Phrase Representations and Syntactic Parsing with Recursive Neural Networks. In *Proceedings of the Deep Learning and Unsupervised Feature Learning Workshop of NIPS 2010*, pages 1–9.

Richard Socher, John Bauer, Christopher D. Manning, and Ng Andrew Y. 2013. Parsing with compositional vector grammars. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 455–465, Sofia, Bulgaria, August. Association for Computational Linguistics.

Yuta Tsuboi. 2014. Neural networks leverage corpus-wide information for part-of-speech tagging. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 938–950, Doha, Qatar, October. Association for Computational Linguistics.