# Syntax-Aware Multi-Sense Word Embeddings for Deep Compositional Models of Meaning

**Jianpeng Cheng**
University of Oxford
Department of
Computer Science
jianpeng.cheng@stcatz.oxon.org

**Dimitri Kartsaklis**
Queen Mary University of London
School of Electronic Engineering
and Computer Science
d.kartsaklis@qmul.ac.uk

## Abstract

Deep compositional models of meaning acting on distributional representations of words in order to produce vectors of larger text constituents are evolving to a popular area of NLP research. We detail a compositional distributional framework based on a rich form of word embeddings that aims at facilitating the interactions between words in the context of a sentence. Embeddings and composition layers are jointly learned against a generic objective that enhances the vectors with syntactic information from the surrounding context. Furthermore, each word is associated with a number of senses, the most plausible of which is selected dynamically during the composition process. We evaluate the produced vectors qualitatively and quantitatively with positive results. At the sentence level, the effectiveness of the framework is demonstrated on the MSRPar task, for which we report results within the state-of-the-art range.

## 1 Introduction

Representing the meaning of words by using their distributional behaviour in a large text corpus is a well-established technique in NLP research that has been proved useful in numerous tasks. In a distributional model of meaning, the semantic representation of a word is given as a vector in some high dimensional vector space, obtained either by explicitly collecting co-occurrence statistics of the target word with words belonging to a representative subset of the vocabulary, or by directly optimizing the word vectors against an objective function in some neural-network based architecture (Collobert and Weston, 2008; Mikolov et al., 2013).

Regardless their method of construction, distributional models of meaning do not scale up to larger text constituents such as phrases or sentences, since the uniqueness of multi-word expressions would inevitably lead to data sparsity problems, thus to unreliable vectorial representations. The problem is usually addressed by the provision of a compositional function, the purpose of which is to prepare a vectorial representation for a phrase or sentence by combining the vectors of the words therein. While the nature and complexity of these compositional models may vary, approaches based on deep-learning architectures have been shown to be especially successful in modelling the meaning of sentences for a variety of tasks (Socher et al., 2012; Kalchbrenner et al., 2014).

The mutual interaction of distributional word vectors by a means of a compositional model provides many opportunities for interesting research, the majority of which still remains to be explored. One such direction is to investigate in what way lexical ambiguity affects the compositional process. In fact, recent work has shown that shallow multi-linear compositional models that explicitly handle extreme cases of lexical ambiguity in a step prior to composition present consistently better performance than their "ambiguous" counterparts (Kartsaklis and Sadrzadeh, 2013; Kartsaklis et al., 2014). A first attempt to test these observations in a deep compositional setting has been presented by Cheng et al. (2014) with promising results.

Furthermore, a second important question relates to the very nature of the word embeddings used in the context of a compositional model. In a setting of this form, word vectors are not any more just a means for discriminating words based on their underlying semantic relationships; the main goal of a word vector is to contribute to a bigger whole—a task in which syntax, along with semantics, also plays a very important role. It is a central point of this paper, therefore, that in a compositional distributional model of meaning word vectors should be injected with information that reflects their syntactical roles in the training corpus.

The purpose of this work is to improve the current practice in deep compositional models of meaning in relation to both the compositional process itself and the quality of the word embeddings used therein. We propose an architecture for jointly training a compositional model and a set of word embeddings, in a way that imposes dynamic word sense induction for each word during the learning process. Note that this is in contrast with recent work in multi-sense neural word embeddings (Neelakantan et al., 2014), in which the word senses are learned without any compositional considerations in mind.

Furthermore, we make the word embeddings syntax-aware by introducing a variation of the hinge loss objective function of Collobert and Weston (2008), in which the goal is not only to predict the occurrence of a target word in a context, but to also predict the *position* of the word within that context. A qualitative analysis shows that our vectors reflect both semantic and syntactic features in a concise way.

In all current deep compositional distributional settings, the word embeddings are internal parameters of the model with no use for any other purpose than the task for which they were specifically trained. In this work, one of our main considerations is that the joint training step should be generic enough to not be tied in any particular task. In this way the word embeddings and the derived compositional model can be learned on data much more diverse than any task-specific dataset, reflecting a wider range of linguistic features. Indeed, experimental evaluation shows that the produced word embeddings can serve as a high quality general-purpose semantic word space, presenting performance on the Stanford Contextual Word Similarity (SCWS) dataset of Huang et al. (2012) competitive to and even better of the performance of well-established neural word embeddings sets.

Finally, we propose a dynamic disambiguation framework for a number of existing deep compositional models of meaning, in which the multisense word embeddings and the compositional model of the original training step are further refined according to the purposes of a specific task at hand. In the context of paraphrase detection, we achieve a result very close to the current state-of-the-art on the Microsoft Research Paraphrase Corpus (Dolan and Brockett, 2005). An interesting aspect at the sideline of the paraphrase detection experiment is that, in contrast to mainstream approaches that mainly rely on simple forms of classifiers, we approach the problem by following a siamese architecture (Bromley et al., 1993).

## 2 Background and related work

### 2.1 Distributional models of meaning

Distributional models of meaning follow the *distributional hypothesis* (Harris, 1954), which states that two words that occur in similar contexts have similar meanings. Traditional approaches for constructing a word space rely on simple counting: a word is represented by a vector of numbers (usually smoothed by the application of some function such as point-wise mutual information) which show how frequently this word co-occurs with other possible context words in a corpus of text.

In contrast to these methods, a recent class of distributional models treat word representations as parameters directly optimized on a word prediction task (Bengio et al., 2003; Collobert and Weston, 2008; Mikolov et al., 2013; Pennington et al., 2014). Instead of relying on observed co-occurrence counts, these models aim to maximize the objective function of a neural net-based architecture; Mikolov et al. (2013), for example, compute the conditional probability of observing words in a context around a target word (an approach known as the *skip-gram model*). Recent studies have shown that, compared to their co-occurrence counterparts, neural word vectors reflect better the semantic relationships between words (Baroni et al., 2014) and are more effective in compositional settings (Milajevs et al., 2014).

### 2.2 Syntactic awareness

Since the main purpose of distributional models until now was to measure the semantic relatedness of words, relatively little effort has been put into making word vectors aware of information regarding the syntactic role under which a word occurs in a sentence. In some cases the vectors are POS-tag specific, so that 'book' as noun and 'book' as verb are represented by different vectors (Kartsaklis and Sadrzadeh, 2013). Furthermore, word spaces in which the context of a target word is determined by means of grammatical dependencies (Padó and Lapata, 2007) are more effective in capturing syntactic relations than approaches based on simple word proximity.

For word embeddings trained in neural settings, syntactic information is not usually taken explicitly into account, with some notable exceptions. At the lexical level, Levy and Goldberg (2014) propose an extension of the skip-gram model

based on grammatical dependencies. Following a different approach, Mnih and Kavukcuoglu (2013) weight the vector of each context word depending on its distance from the target word. With regard to compositional settings (discussed in the next section), Hashimoto et al. (2014) use dependency-based word embeddings by employing a hinge loss objective, while Hermann and Blunsom (2013) condition their objectives on the CCG types of the involved words.

As we will see in Section 3, the current paper offers an appealing alternative to those approaches that does not depend on grammatical relations or types of any form.

## 2.3 Compositionality in distributional models

The methods that aim to equip distributional models of meaning with compositional abilities come in many different levels of sophistication, from simple element-wise vector operators such as addition and multiplication (Mitchell and Lapata, 2008) to category theory (Coecke et al., 2010). In this latter work relational words (such as verbs or adjectives) are represented as multi-linear maps acting on vectors representing their arguments (nouns and noun phrases). In general, the above models are shallow in the sense that they do not have functional parameters and the output is produced by the direct interaction of the inputs; yet they have been shown to capture the compositional meaning of sentences to an adequate degree.

The idea of using neural networks for compositionality in language appeared 25 years ago in a seminal paper by Pollack (1990), and has been recently re-popularized by Socher and colleagues (Socher et al., 2011a; Socher et al., 2012). The compositional architecture used in these works is that of a *recursive neural network* (RecNN) (Socher et al., 2011b), where the words get composed by following a parse tree. A particular variant of the RecNN is the *recurrent neural network* (RNN), in which a sentence is assumed to be generated by aggregating words in sequence (Mikolov et al., 2010). Furthermore, some recent work (Kalchbrenner et al., 2014) models the meaning of sentences by utilizing the concept of a convolutional neural network (LeCun et al., 1998), the main characteristic of which is that it acts on small overlapping parts of the input vectors. In all the above models, the word embeddings and the weights of the compositional layers are optimized against a task-specific objective function.

In Section 3 we will show how to remove the restriction of a supervised setting, introduc-

ing a generic objective that can be trained on any general-purpose text corpus. While we focus on recursive and recurrent neural network architectures, the general ideas we will discuss are in principle model-independent.

## 2.4 Disambiguation in composition

Regardless of the way they address composition, all the models of Section 2.3 rely on ambiguous word spaces, in which every meaning of a polysemous word is merged into a single vector. Especially for cases of homonymy (such as 'bank', 'organ' and so on), where the same word is used to describe two or more completely unrelated concepts, this approach is problematic: the semantic representation of the word becomes the average of all senses, inadequate to express any of them in a reliable way.

To address this problem, a prior disambiguation step on the word vectors is often introduced, the purpose of which is to find the word representations that best fit to the given context, before composition takes place (Reddy et al., 2011; Kartsaklis et al., 2013; Kartsaklis and Sadrzadeh, 2013; Kartsaklis et al., 2014). This idea has been tested on algebraic and tensor-based compositional functions with very positive results. Furthermore, it has been also found to provide minimal benefits for a RecNN compositional architecture in a number of phrase and sentence similarity tasks (Cheng et al., 2014). This latter work clearly suggests that explicitly dealing with lexical ambiguity in a deep compositional setting is an idea that is worth to be further explored. While treating disambiguation as only a preprocessing step is a strategy less than optimal for a neural setting, one would expect that the benefits should be greater for an architecture in which the disambiguation takes place in a dynamic fashion during training.

We are now ready to start detailing a compositional model that takes into account the above considerations. The issue of lexical ambiguity is covered in Section 4; Section 3 below deals with generic training and syntactic awareness.

## 3 Syntax-based generic training

We propose a novel architecture for learning word embeddings and a compositional model to use them in a single step. The learning takes places in the context of a RecNN (or an RNN), and both word embeddings and parameters of the compositional layer are optimized against a generic objective function that uses a hinge loss function.
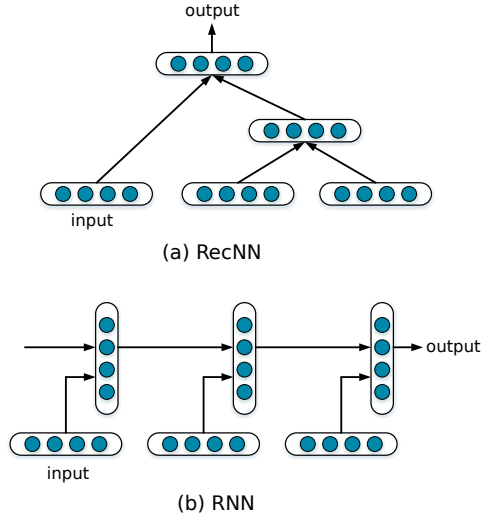
Figure 1: Recursive (a) and recurrent (b) neural networks.

Figure 1 shows the general form of recursive and recurrent neural networks. In architectures of this form, a compositional layer is applied on each pair of inputs $\mathbf{x_1}$ and $\mathbf{x_2}$ in the following way:

$$\mathbf{p} = g(\mathbf{W}\mathbf{x}_{[1:2]} + \mathbf{b}) \tag{1}$$

where $\mathbf{x}_{[1:2]}$ denotes the concatenation of the two vectors, $g$ is a non-linear function, and $\mathbf{W}, \mathbf{b}$ are the parameters of the model. In the RecNN case, the compositional process continues recursively by following a parse tree until a vector for the whole sentence or phrase is produced; on the other hand, an RNN assumes that a sentence is generated in a left-to-right fashion, taking into consideration no dependencies other than word adjacency.

We amend the above setting by introducing a novel layer on the top of the compositional one, which scores the linguistic plausibility of the composed sentence or phrase vector with regard to *both* syntax and semantics. Following Collobert and Weston (2008), we convert the unsupervised learning problem to a supervised one by corrupting training sentences. Specifically, for each sentence $s$ we create two sets of negative examples. In the first set, $S'$, the target word within a given context is replaced by a random word; as in the original C&W paper, this set is used to enforce semantic coherence in the word vectors. Syntactic coherence is enforced by a second set of negative examples, $S''$, in which the words of the context have been randomly shuffled. The objective function is defined in terms of the following hinge losses:

$$\sum_{s \in S} \sum_{s' \in S'} \max(0, m - f(s) + f(s')) \tag{2}$$

$$\sum_{s \in S} \sum_{s'' \in S''} \max(0, m - f(s) + f(s'')) \tag{3}$$

where $S$ is the set of sentences, $f$ the compositional layer, and $m$ a margin we wish to retain between the scores of the positive training examples and the negative ones. During training, all parameters in the scoring layer, the compositional layers and word representations are jointly updated by error back-propagation. As output, we get both general-purpose syntax-aware word representations and weights for the corresponding compositional model.

## 4 From words to senses

We now extend our model to address lexical ambiguity. We achieve that by applying a gated architecture, similar to the one used in the multi-sense model of Neelakantan et al. (2014), but advancing the main idea to the compositional setting detailed in Section 3.

We assume a fixed number of $n$ senses per word.[1] Each word is associated with a main vector (obtained for example by using an existing vector set, or by simply applying the process of Section 3 in a separate step), as well as with $n$ vectors denoting cluster centroids and an equal number of sense vectors. Both cluster centroids and sense vectors are randomly initialized in the beginning of the process. For each word $w_t$ in a training sentence, we prepare a context vector by averaging the main vectors of all other words in the same context. This context vector is compared with the cluster centroids of $w_t$ by cosine similarity, and the sense corresponding to the closest cluster is selected as the most representative of $w_t$ in the current context. The selected cluster centroid is updated by the addition of the context vector, and the associated sense vector is passed as input to the compositional layer. The selected sense vectors for each word in the sentence are updated by back-propagation, based on the objectives of Equations 2 and 3. The overall architecture of our model, as described in this and the previous section, is illustrated in Figure 2.

## 5 Task-specific dynamic disambiguation

The model of Figure 2 decouples the training of word vectors and compositional parameters from

---

[1]Note that in principle the fixed number of senses assumption is not necessary; Neelakantan et al. (2014), for example, present a version of their model in which new senses are added dynamically when appropriate.
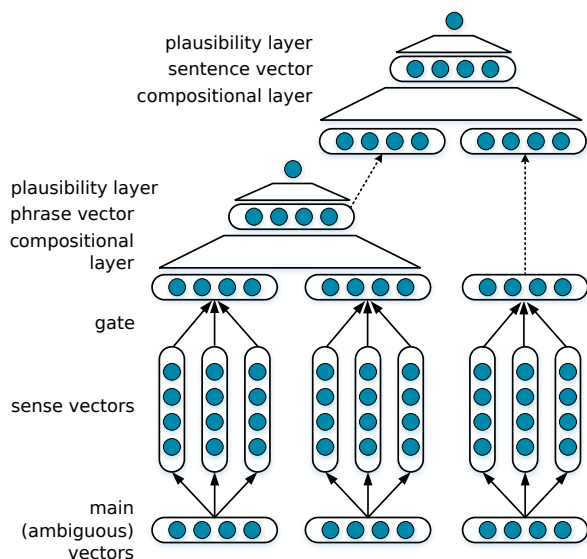
Figure 2: Training of syntax-aware multi-sense embeddings in the context of a RecNN.

a specific task, and as a consequence from any task-specific training dataset. However, note that by replacing the plausibility layer with a classifier trained for some task at hand, you get a task-specific network that transparently trains multi-sense word embeddings and applies dynamic disambiguation on the fly. While this idea of a single-step direct training seems appealing, one consideration is that the task-specific dataset used for the training will not probably reflect the linguistic variety that is required to exploit the expressiveness of the setting in its full. Additionally, in many cases the size of datasets tied to specific tasks is prohibiting for training a deep architecture.

It is a merit of this proposal that, in cases like these, it is possible for one to train the generic model of Figure 2 on any large corpus of text, and then use the produced word vectors and compositional weights to initialize the parameters of a more specific version of the architecture. As a result, the trained parameters will be further refined according to the task-specific objective. Figure 3 illustrates the generic case of a compositional framework applying dynamic disambiguation. Note that here sense selection takes place by a soft-max layer, which can be directly optimized on the task objective.

## 6  A siamese network for paraphrase detection

We will test the dynamic disambiguation framework of Section 5 in a paraphrase detection task. A *paraphrase* is a restatement of the meaning of a

sentence using different words and/or syntax. The goal of a paraphrase detection model, thus, is to examine two sentences and decide if they express the same meaning.

While the usual way to approach this problem is to utilize a classifier that acts (for example) on the concatenation of the two sentence vectors, in this work we follow a novel perspective: specifically, we apply a *siamese architecture* (Bromley et al., 1993), a concept that has been extensively used in computer vision (Hadsell et al., 2006; Sun et al., 2014). While siamese networks have been also used in the past for NLP purposes (for example, by Yih et al. (2011)), to the best of our knowledge this is the first time that such a setting is applied for paraphrase detection.

In our model, two networks sharing the same parameters are used to compute the vectorial representations of two sentences, the paraphrase relation of which we wish to detect; this is achieved by employing a cost function that compares the two vectors. There are two commonly used cost functions: the first is based on the $L_2$ norm (Hadsell et al., 2006; Sun et al., 2014), while the second on the cosine similarity (Nair and Hinton, 2010; Sun et al., 2014). The $L_2$ norm variation is capable of handling differences in the magnitude of the vectors. Formally, the cost function is defined as:

$$E_f = \begin{cases} \frac{1}{2} \left\| f(s_1) - f(s_2) \right\|_2^2, & \text{if } y = 1 \\ \frac{1}{2} \max(0, m - \left\| f(s_1) - f(s_2) \right\|_2)^2, & o.w. \end{cases}$$

where $s_1, s_2$ are the input sentences, $f$ the compositional layer (so $f(s_1)$ and $f(s_2)$ refer to sentence vectors), and $y = 1$ denotes a paraphrase relationship between the sentences; $m$ stands for the margin, a hyper-parameter chosen in advance. On
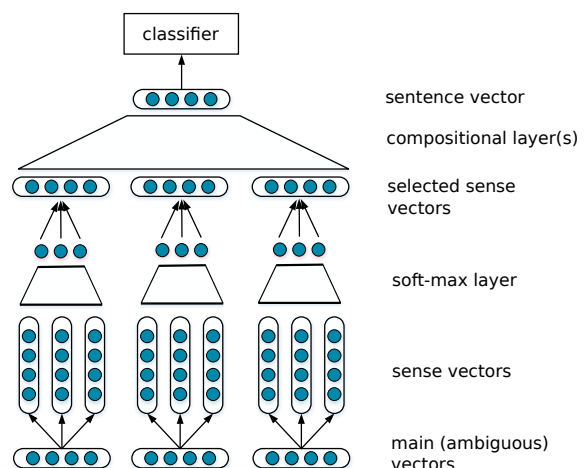


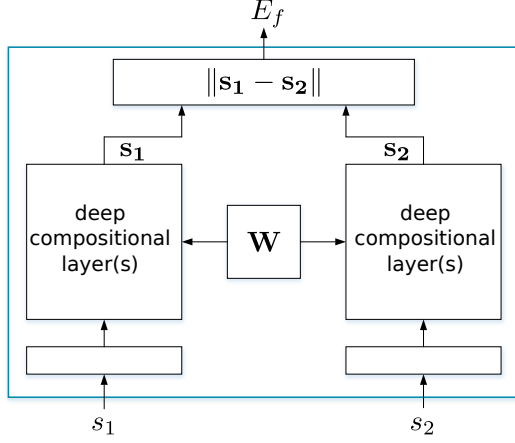Figure 3: Dynamic disambiguation in a generic compositional deep net.

Figure 4: A siamese network for paraphrase detection.

the other hand, the cost function based on cosine similarity handles only directional differences, as follows:

$$E_f = \frac{1}{2}(y - \sigma(wd + b))^2 \qquad (4)$$

where $d = \frac{f(s_1) \cdot f(s_2)}{\|f(s_1)\|_2 \|f(s_2)\|_2}$ is the cosine similarity of the two sentence vectors, $w$ and $b$ are the scaling and shifting parameters to be optimized, $\sigma$ is the sigmoid function and $y$ is the label. In the experiments that will follow in Section 7.4, both of these cost functions are evaluated. The overall architecture is shown in Figure 4.

In Section 7.4 we will use the pre-trained vectors and compositional weights for deriving sentence representations that will be subsequently fed to the siamese network. When the dynamic disambiguation framework is used, the sense vectors of the words are updated during training so that the sense selection process is gradually refined.

## 7 Experiments

We evaluate the quality of the compositional word vectors and the proposed deep compositional framework in the tasks of word similarity and paraphrase detection, respectively.

### 7.1 Model pre-training

In all experiments the word representations and compositional models are pre-trained on the British National Corpus (BNC), a general-purpose text corpus that contains 6 million sentences of written and spoken English. For comparison we train two sets of word vectors and compositional models, one ambiguous and one multi-sense (fix-

ing 3 senses per word). The dimension of the embeddings is set to 300.

As our compositional architectures we use a RecNN and an RNN. In the RecNN case, the words are composed by following the result of an external parser, while for the RNN the composition takes place in sequence from left to right. To avoid the exploding or vanishing gradient problem (Bengio et al., 1994) for long sentences, we employ a *long short-term memory* (LSTM) network (Hochreiter and Schmidhuber, 1997). During the training of each model, we minimize the hinge loss in Equations 2 and 3. The plausibility layer is implemented as a 2-layer network, with 150 units at the hidden layer, and is applied at each individual node (as opposed to a single application at the sentence level). All parameters are updated with mini-batches by AdaDelta (Zeiler, 2012) gradient descent method ($\lambda = 0.03$, initial $\alpha = 0.05$).

### 7.2 Qualitative evaluation of the word vectors

As a first step, we qualitatively evaluate the trained word embeddings by examining the nearest neighbours lists of a few selected words. We compare the results with those produced by the skip-gram model (SG) of Mikolov et al. (2013) and the language model (CW) of Collobert and Weston (2008). We refer to our model as SAMS (Syntax-Aware Multi-Sense). The results in Table 1 show clearly that our model tends to group words that are both semantically and syntactically related; for example, and in contrast with the compared models which group words only at the semantic level, our model is able to retain tenses, numbers (singulars and plurals), and gerunds.

The observed behaviour is comparable to that of embedding models with objective functions conditioned on grammatical relations between words; Levy and Goldberg (2014), for example, present a similar table for their dependency-based extension of the skip-gram model. The advantage of our approach against such models is twofold: firstly, the word embeddings are accompanied by a generic compositional model that can be used for creating sentence representations independently of any specific task; and secondly, the training is quite forgiving to data sparsity problems that in general a dependency-based approach would intensify (since context words are paired with the grammatical relations they occur with the target word). As a result, a small corpus such as the BNC is sufficient for producing high quality syntax-aware word embeddings.

1559

| | SG | CW | SAMS |
|---|---|---|---|
| begged | beg, begging, cried | begging, pretended, beg | persuaded, asked, cried |
| refused | refusing, refuses, refusal | refusing , declined, refuse | declined, rejected, denied |
| interrupted | interrupting, punctuated, interrupt | interrupts, interrupt, interrupting | punctuated, preceded, disrupted |
| themes | thematic, theme, notions | theme, concepts, subtext | meanings, concepts, ideas |
| patiently | impatiently, waited, waits | impatiently, queue, expectantly | impatiently, siliently, anxiously |
| player | players, football, league | game, club, team | athlete, sportsman, team |
| prompting | prompted, prompt, sparking | prompt, amid, triggered | sparking, triggering, forcing |
| reproduce | reproducing, replicate, humans | reproducing, thrive, survive | replicate, produce, repopulate |
| predictions | prediction, predict, forecasts | predicting, assumption, predicted | expectations, projections, forecasts |

Table 1: Nearest neighbours for a number of words with various embedding models.

## 7.3 Word similarity

We now proceed to a quantitative evaluation of our embeddings on the Stanford Contextual Word Similarity (SCWS) dataset of Huang et al. (2012). The dataset contains 2,003 pairs of words and the contexts they occur in. We can therefore make use of the contextual information in order to select the most appropriate sense for each ambiguous word. Similarly to Neelakantan et al. (2014), we use three different metrics: *globalSim* measures the similarity between two ambiguous word vectors; *localSim* selects a single sense for each word based on the context and computes the similarity between the two sense vectors; *avgSim* represents each word as a weighted average of all senses in the given context and computes the similarity between the two weighted sense vectors.

We compute and report the Spearman's correlation between the embedding similarities and human judgments (Table 2). In addition to the skip-gram and Collobert and Weston models, we also compare against the CBOW model (Mikolov et al., 2013) and the multi-sense skip-gram (MSSG) model of Neelakantan et al. (2014).

| Model | globalSim | localSim | avgSim |
|---|---|---|---|
| CBOW | 59.5 | – | – |
| SG | **61.8** | – | – |
| CW | 55.3 | – | – |
| MSSG | 61.3 | 56.7 | 62.1 |
| SAMS | 59.9 | **58.5** | **62.5** |

Table 2: Results for the word similarity task (Spearman's $\rho \times 100$).

Among all methods, only the MSSG model and ours are capable of learning multi-prototype word representations. Our embeddings show top performance for *localSim* and *avgSim* measures, and performance competitive to that of MSSG and SG for *globalSim*, both of which use a hierarchical

soft-max as their objective function. Compared to the original C&W model, our version presents an improvement of 4.6%—a clear indication for the effectiveness of the proposed learning method and the enhanced objective.

## 7.4 Paraphrase detection

In the last set of experiments, the proposed compositional distributional framework is evaluated on the Microsoft Research Paraphrase Corpus (MSRPC) (Dolan and Brockett, 2005), which contains 5,800 pairs of sentences. This is a binary classification task, with labels provided by human annotators. We apply the siamese network detailed in Section 6.

While MSRPC is one of the most used datasets for evaluating paraphrase detection models, its size is prohibitory for any attempt of training a deep architecture. Therefore, for our training we rely on a much larger external dataset, the Paraphrase Database (PPDB) (Ganitkevitch et al., 2013). The PPDB contains more than 220 million paraphrase pairs, of which 73 million are phrasal paraphrases and 140 million are paraphrase patterns that capture syntactic transformations of sentences. We use these phrase- and sentence-level paraphrase pairs as additional training contexts to fine-tune the generic compositional model parameters and word embeddings and to train the baseline models. The original training set of the MSRPC is used as validation set for deciding hyperparameters, such as the margin of the error function and the number of training epochs.

The evaluations were conducted on various aspects, and the models are gradually refined to demonstrate performance within the state-of-the-art range.

**Comparison of the two error functions** In the first evaluation, we compare the two error functions of the siamese network using only ambigu-

ous vectors. As we can see in Table 3, the cosine error function consistently outperforms the $L_2$ norm-based one for both compositional models, providing a yet another confirmation of the already well-established fact that similarity in semantic vector spaces is better reflected by length-invariant measures.

| Model | $L_2$ | Cosine |
|-------|-------|--------|
| RecNN | 73.8 | **74.9** |
| RNN | 73.0 | **74.3** |

Table 3: Results with different error functions for the paraphrase detection task (accuracy $\times$ 100).

**Effectiveness of disambiguation** We now proceed to compare the effectiveness of the two compositional models when using ambiguous vectors and multi-sense vectors, respectively. Our error function is set to cosine similarity, following the results of the previous evaluation. When dynamic disambiguation is applied, we test two methods of selecting sense vectors: in the hard case the vector of the most plausible sense is selected, while in the soft case a new vector is prepared as the weighted average of all sense vectors according to probabilities returned by the soft-max layer (see Figure 3). As a baseline we use a simple compositional model based on vector addition.

The dynamic disambiguation models and the additive baseline are compared with variations that use a simple prior disambiguation step applied on the word vectors. This is achieved by first selecting for each word the sense vector that is the closest to the average of all other word vectors in the same sentence, and then composing the selected sense vectors without further considerations regarding ambiguity. The baseline model and the prior disambiguation variants are trained as separate logistic regression classifiers. The results are shown in Table 4.

| Model | Ambig. | Prior | Hard DD | Soft DD |
|-------|--------|-------|---------|---------|
| Addition | 69.9 | 71.3 | – | – |
| RecNN | 74.9 | 75.3 | 75.7 | **76.0** |
| RNN | 74.3 | 74.6 | 75.1 | **75.2** |

Table 4: Different disambiguation choices for the paraphrase detection task (accuracy $\times$ 100).

Overall, disambiguated vectors work better than the ambiguous ones, with the improvement to be more significant for the additive model; there, a simple prior disambiguation step produces 1.4% gains. For the deep compositional models, simple

prior disambiguation is still helpful with small improvements, a result which is consistent with the findings of Cheng et al. (2014). The small gains of the prior disambiguation models over the ambiguous models clearly show that deep architectures are quite capable of performing this elementary form of sense selection intrinsically, as part of the learning process itself. However, the situation changes when the dynamic disambiguation framework is used, where the gains over the ambiguous version become more significant. Comparing the two ways of dynamic disambiguation (hard method and soft method), the numbers that the soft method gives are slightly higher, producing a total gain of 1.1% over the ambiguous version for the RecNN case.[2]

Note that, at this stage, the advantage of using the dynamic disambiguation framework over simple prior disambiguation is still small (0.7% for the case of RecNN). We seek the reason behind this in the recursive nature of our architecture, which tends to progressively "hide" local features of word vectors, thus diminishing the effect of the fine-tuned sense vectors produced by the dynamic disambiguation mechanism. The next section discusses the problem and provides a solution.

**The role of pooling** One of the problems of the recursive and recurrent compositional architectures, especially in grammars with strict branching structure such as in English, is that any given composition is usually the product of a terminal and a non-terminal; i.e. a single word can contribute to the meaning of a sentence to the same extent as the rest of a sentence on its whole, as below:

$[[kids]_{NP} [play ball games in the park]_{VP}]_S$

In the above case, the contribution of the words within the verb phrase to the final sentence representation will be faded out due to the recursive composition mechanism. Inspired by related work in computer vision (Sun et al., 2014), we attempt to alleviate this problem by introducing an average pooling layer at the sense vector level and adding the resulting vector to the sentence representation. By doing this we expect that the new sentence vector will reflect local features from all words in the sentence that can help in the classification in a more direct way. The results for the new deep architectures are shown in Table 5, where we see substantial improvements for both deep nets. More importantly, the effect of dynamic

---

[2]For all subsequent experiments, the reported results are based on the soft selection method.

disambiguation now becomes more significant, as expected by our analysis.

Table 5 also includes results for two models trained in a single step, with word and sense vectors randomly initialized at the beginning of the process. We see that, despite the large size of the training set, the results are much lower than the ones obtained when using the pre-training step. This demonstrates the importance of the initial training on a general-purpose corpus: the resulting vectors reflect linguistic information that, although not obtainable from the task-specific training, can make great difference in the result of the classification.

| Model | Ambig. | Prior | Dynamic |
|---|---|---|---|
| RecNN+pooling | 75.5 | 76.3 | **77.6** |
| RNN+pooling | 74.8 | 75.9 | **76.6** |
| 1-step RecNN+pooling | 74.4 | – | 72.9 |
| 1-step RNN+pooling | 73.6 | – | 73.1 |

Table 5: Results with average pooling for the paraphrase detection task (accuracy × 100).

**Cross-model comparison**  In this section we propose a method to further improve the performance of our models, and we present an evaluation against some of the previously reported results.

We notice that using distributional properties alone cannot capture efficiently subtle aspects of a sentence, for example numbers or human names. However, even small differences on those aspects between two sentences can lead to a different classification result. Therefore, we train (using the MSPRC training data) an additional logistic regression classifier which is based not only on the embeddings similarity, but also on a few hand-engineered features. We then ensemble the new classifier (C1) with the original one. In terms of feature selection, we follow Socher et al. (2011a) and Blacoe and Lapata (2012) and add the following features: the difference in sentence length, the unigram overlap among the two sentences, features related to numbers (including the presence or absence of numbers from a sentence and whether or not the numbers in the two sentences are the same). In Table 6 we report results of the original model and the ensembled model, and we compare with the performance of other existing models.

In all of the implemented models (including the additive baseline), disambiguation is performed to guarantee the best performance. We see that by ensembling the original classifier with C1, we improve the result of the previous section by another 1%. This is the second best result reported so far

| | Model | Acc. | F1 |
|---|---|---|---|
| **BL** | All positive | 66.5 | 79.9 |
| | Addition (disamb.) | 71.3 | 81.1 |
| **Dynamic Dis.** | RecNN | 76.0 | 84.0 |
| | RecNN+Pooling | 77.6 | 84.7 |
| | RecNN+Pooling+C1 | **78.6** | **85.3** |
| | RNN | 75.2 | 83.6 |
| | RNN+Pooling | 76.6 | 84.3 |
| | RNN+Pooling+C1 | 77.5 | 84.6 |
| **Published results** | Mihalcea et al. (2006) | 70.3 | 81.3 |
| | Rus et al. (2008) | 70.6 | 80.5 |
| | Qiu et al. (2006) | 72.0 | 81.6 |
| | Islam and Inkpen (2009) | 72.6 | 81.3 |
| | Fernando and Stevenson (2008) | 74.1 | 82.4 |
| | Wan et al. (2006) | 75.6 | 83.0 |
| | Das and Smith (2009) | 76.1 | 82.7 |
| | Socher et al. (2011a) | 76.8 | 83.6 |
| | Madnani et al. (2012) | 77.4 | 84.1 |
| | Ji and Eisenstein (2013) | **80.4** | **85.9** |

Table 6: Cross-model comparison in the paraphrase detection task.

for the specific task, with a 0.6 difference in F-score from the first (Ji and Eisenstein, 2013).[3]

## 8 Conclusion and future work

The main contribution of this paper is a deep compositional distributional model acting on linguistically motivated word embeddings.[4] The effectiveness of the syntax-aware, multi-sense word vectors and the dynamic compositional disambiguation framework in which they are used was demonstrated by appropriate tasks at the lexical and sentence level, respectively, with very positive results. As an aside, we also demonstrated the benefits of a siamese architecture in the context of a paraphrase detection task. While the architectures tested in this work were limited to a RecNN and an RNN, the ideas we presented are in principle directly applicable to any kind of deep network. As a future step, we aim to test the proposed models on a convolutional compositional architecture, similar to that of Kalchbrenner et al. (2014).

## Acknowledgments

---

[3]Source: ACL Wiki (`http://www.aclweb.org/acl-wiki`), August 2015.

[4]Code in Python/Theano and the word embeddings can be found at `https://github.com/cheng6076`.

# References

Marco Baroni, Georgiana Dinu, and Germán Kruszewski. 2014. Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, volume 1.

Yoshua Bengio, Patrice Simard, and Paolo Frasconi. 1994. Learning long-term dependencies with gradient descent is difficult. *Neural Networks, IEEE Transactions on*, 5(2):157–166.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A neural probabilistic language model. *The Journal of Machine Learning Research*, 3:1137–1155.

William Blacoe and Mirella Lapata. 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. Association for Computational Linguistics.

Jane Bromley, James W Bentz, Léon Bottou, Isabelle Guyon, Yann LeCun, Cliff Moore, Eduard Säckinger, and Roopak Shah. 1993. Signature verification using a siamese time delay neural network. *International Journal of Pattern Recognition and Artificial Intelligence*, 7(04):669–688.

Jianpeng Cheng, Dimitri Kartsaklis, and Edward Grefenstette. 2014. Investigating the role of prior disambiguation in deep-learning compositional models of meaning. In *2nd Workshop of Learning Semantics, NIPS 2014*, Montreal, Canada, December.

B Coecke, M Sadrzadeh, and S Clark. 2010. Mathematical foundations for a distributional compositional model of meaning. lambek festschrift. *Linguistic Analysis*, 36:345–384.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. ACM.

Dipanjan Das and Noah A Smith. 2009. Paraphrase identification as probabilistic quasi-synchronous recognition. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1*, pages 468–476. Association for Computational Linguistics.

W.B. Dolan and C. Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Third International Workshop on Paraphrasing (IWP2005)*.

Samuel Fernando and Mark Stevenson. 2008. A semantic similarity approach to paraphrase detection. In *Proceedings of the 11th Annual Research Colloquium of the UK Special Interest Group for Computational Linguistics*, pages 45–52. Citeseer.

Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2013. Ppdb: The paraphrase database. In *HLT-NAACL*, pages 758–764.

Raia Hadsell, Sumit Chopra, and Yann LeCun. 2006. Dimensionality reduction by learning an invariant mapping. In *Computer vision and pattern recognition, 2006 IEEE computer society conference on*, volume 2, pages 1735–1742. IEEE.

Zellig S Harris. 1954. Distributional structure. *Word*.

Kazuma Hashimoto, Pontus Stenetorp, Makoto Miwa, and Yoshimasa Tsuruoka. 2014. Jointly learning word representations and composition functions using predicate-argument structures. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1544–1555, Doha, Qatar, October. Association for Computational Linguistics.

Karl Moritz Hermann and Phil Blunsom. 2013. The role of syntax in vector space models of compositional semantics. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 894–904, Sofia, Bulgaria, August. Association for Computational Linguistics.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Eric H Huang, Richard Socher, Christopher D Manning, and Andrew Y Ng. 2012. Improving word representations via global context and multiple word prototypes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 873–882. Association for Computational Linguistics.

Aminul Islam and Diana Inkpen. 2009. Semantic similarity of short texts. *Recent Advances in Natural Language Processing V*, 309:227–236.

Yangfeng Ji and Jacob Eisenstein. 2013. Discriminative improvements to distributional sentence similarity. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 891–896, Seattle, Washington, USA, October. Association for Computational Linguistics.

Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. 2014. A convolutional neural network for modelling sentences. *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, June.

Dimitri Kartsaklis and Mehrnoosh Sadrzadeh. 2013. Prior disambiguation of word tensors for constructing sentence vectors. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1590–1601, Seattle, Washington, USA, October. Association for Computational Linguistics.

Dimitri Kartsaklis, Mehrnoosh Sadrzadeh, and Stephen Pulman. 2013. Separating disambiguation from composition in distributional semantics. In *Proceedings of 17th Conference on Natural Language Learning (CoNLL)*, pages 114–123, Sofia, Bulgaria, August.

Dimitri Kartsaklis, Nal Kalchbrenner, and Mehrnoosh Sadrzadeh. 2014. Resolving lexical ambiguity in tensor regression models of meaning. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Vol. 2: Short Papers)*, pages 212–217, Baltimore, USA, June. Association for Computational Linguistics.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

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

Nitin Madnani, Joel Tetreault, and Martin Chodorow. 2012. Re-examining machine translation metrics for paraphrase identification. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 182–190. Association for Computational Linguistics.

Rada Mihalcea, Courtney Corley, and Carlo Strapparava. 2006. Corpus-based and knowledge-based measures of text semantic similarity. In *AAAI*, volume 6, pages 775–780.

Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernockỳ, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *INTERSPEECH 2010, 11th Annual Conference of the International Speech Communication Association, Makuhari, Chiba, Japan, September 26-30, 2010*, pages 1045–1048.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119.

Dmitrijs Milajevs, Dimitri Kartsaklis, Mehrnoosh Sadrzadeh, and Matthew Purver. 2014. Evaluating neural word representations in tensor-based compositional settings. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 708–719, Doha, Qatar, October. Association for Computational Linguistics.

Jeff Mitchell and Mirella Lapata. 2008. Vector-based models of semantic composition. In *Proceedings of ACL-08: HLT*, pages 236–244, Columbus, Ohio, June. Association for Computational Linguistics.

Andriy Mnih and Koray Kavukcuoglu. 2013. Learning word embeddings efficiently with noise-contrastive estimation. In *Advances in Neural Information Processing Systems*, pages 2265–2273.

Vinod Nair and Geoffrey E Hinton. 2010. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 807–814.

Arvind Neelakantan, Jeevan Shankar, Alexandre Passos, and Andrew McCallum. 2014. Efficient non-parametric estimation of multiple embeddings per word in vector space. In *Proceedings of EMNLP*.

S. Padó and M. Lapata. 2007. Dependency-based Construction of Semantic Space Models. *Computational Linguistics*, 33(2):161–199.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. *Proceedings of the Empiricial Methods in Natural Language Processing (EMNLP 2014)*, 12.

Jordan B Pollack. 1990. Recursive distributed representations. *Artificial Intelligence*, 46(1):77–105.

Long Qiu, Min-Yen Kan, and Tat-Seng Chua. 2006. Paraphrase recognition via dissimilarity significance classification. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 18–26. Association for Computational Linguistics.

Siva Reddy, Ioannis P Klapaftis, Diana McCarthy, and Suresh Manandhar. 2011. Dynamic and static prototype vectors for semantic composition. In *IJCNLP*, pages 705–713.

Vasile Rus, Philip M McCarthy, Mihai C Lintean, Danielle S McNamara, and Arthur C Graesser. 2008. Paraphrase identification with lexico-syntactic graph subsumption. In *FLAIRS conference*, pages 201–206.

Richard Socher, Eric H Huang, Jeffrey Pennin, Christopher D Manning, and Andrew Y Ng. 2011a. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In *Advances in Neural Information Processing Systems*, pages 801–809.

Richard Socher, Cliff C Lin, Chris Manning, and Andrew Y Ng. 2011b. Parsing natural scenes and natural language with recursive neural networks. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 129–136.

R. Socher, B. Huval, C. Manning, and Ng. A. 2012. Semantic compositionality through recursive matrix-vector spaces. In *Conference on Empirical Methods in Natural Language Processing 2012*.

Yi Sun, Yuheng Chen, Xiaogang Wang, and Xiaoou Tang. 2014. Deep learning face representation by joint identification-verification. In *Advances in Neural Information Processing Systems*, pages 1988–1996.

Stephen Wan, Mark Dras, Robert Dale, and Cécile Paris. 2006. Using dependency-based features to take the para-farce out of paraphrase. In *Proceedings of the Australasian Language Technology Workshop*, volume 2006.

Wen-tau Yih, Kristina Toutanova, John C. Platt, and Christopher Meek. 2011. Learning discriminative projections for text similarity measures. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning*, pages 247–256, Portland, Oregon, USA, June. Association for Computational Linguistics.

Matthew D Zeiler. 2012. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*.