Jointly Learning Word Representations and Composition Functions Using Predicate-Argument Structures

Kazuma Hashimoto†**, Pontus Stenetorp**†**, Makoto Miwa**‡**, and Yoshimasa Tsuruoka**†
†The University of Tokyo, 3-7-1 Hongo, Bunkyo-ku, Tokyo, Japan
{hassy,pontus,tsuruoka}@logos.t.u-tokyo.ac.jp
‡Toyota Technological Institute, 2-12-1 Hisakata, Tempaku-ku, Nagoya, Japan
makoto-miwa@toyota-ti.ac.jp

Abstract

We introduce a novel compositional language model that works on Predicate-Argument Structures (PASs). Our model jointly learns word representations and their composition functions using bag-of-words and dependency-based contexts. Unlike previous word-sequence-based models, our PAS-based model composes arguments into predicates by using the category information from the PAS. This enables our model to capture long-range dependencies between words and to better handle constructs such as verb-object and subject-verb-object relations. We verify this experimentally using two phrase similarity datasets and achieve results comparable to or higher than the previous best results. Our system achieves these results without the need for pre-trained word vectors and using a much smaller training corpus; despite this, for the subject-verb-object dataset our model improves upon the state of the art by as much as $\sim$10% in relative performance.

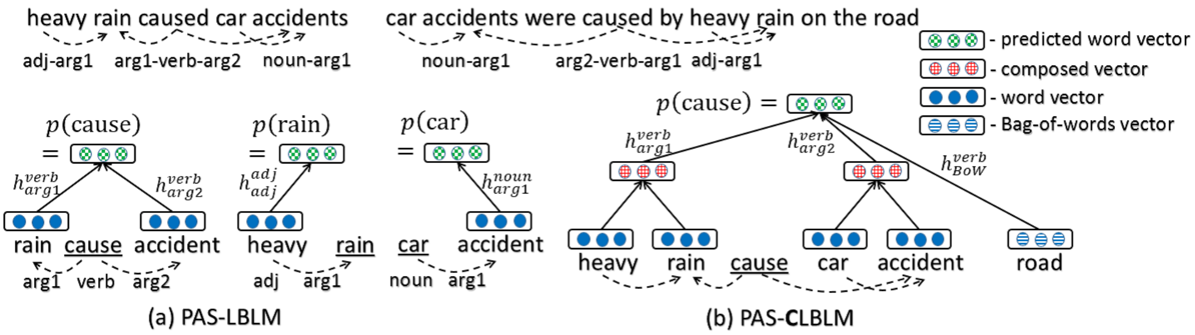
1 Introduction

Studies on embedding single words in a vector space have made notable successes in capturing their syntactic and semantic properties (Turney and Pantel, 2010). These embeddings have also been found to be a useful component for Natural Language Processing (NLP) systems; for example, Turian et al. (2010) and Collobert et al. (2011) demonstrated how low-dimensional word vectors learned by Neural Network Language Models (NNLMs) are beneficial for a wide range of NLP tasks.

Recently, the main focus of research on vector space representation is shifting from word representations to phrase representations (Baroni and Zamparelli, 2010; Grefenstette and Sadrzadeh, 2011; Mitchell and Lapata, 2010; Socher et al., 2012). Combining the ideas of NNLMs and semantic composition, Tsubaki et al. (2013) introduced a novel NNLM incorporating verb-object dependencies. More recently, Levy and Goldberg (2014) presented a NNLM that integrated syntactic dependencies. However, to the best of our knowledge, there is no previous work on integrating a variety of syntactic and semantic dependencies into NNLMs in order to learn *composition functions* as well as word representations. The following question thus arises naturally:

> Can a variety of dependencies be used to jointly learn both stand-alone word vectors and their compositions, embedding them in the same vector space?

In this work, we bridge the gap between purely context-based (Levy and Goldberg, 2014; Mikolov et al., 2013b; Mnih and Kavukcuoglu, 2013) and compositional (Tsubaki et al., 2013) NNLMs by using the flexible set of categories from Predicate-Argument-Structures (PASs). More specifically, we propose a Compositional Log-Bilinear Language Model using PASs (PAS-CLBLM), an overview of which is shown in Figure 1. The model is trained by maximizing the accuracy of predicting target words from their bag-of-words *and* dependency-based context, which provides information about selectional preference. As shown in Figure 1 (b), one of the advantages of the PAS-CLBLM is that the model can treat not only word vectors but also composed vectors as contexts. Since the composed vectors



(a) PAS-LBLM

(b) PAS-CLBLM

Figure 1: An overview of the proposed model: PAS-CLBLM. (a) The PAS-LBLM predicts target words from their bag-of-words and dependency-based context words. (b) The PAS-CLBLM predicts target words using not only context words but also composed vector representations derived from another level of predicate-argument structures. Underlined words are target words and we only depict the bag-of-words vector for the PAS-CLBLM.

are treated as input to the language model in the same way as word vectors, these composed vectors are expected to become similar to word vectors for words with similar meanings.

Our empirical results demonstrate that the proposed model has the ability to learn meaningful representations for adjective-noun, noun-noun, and (subject-) verb-object dependencies. On three tasks of measuring the semantic similarity between short phrases (adjective-noun, noun-noun, and verb-object), the learned composed vectors achieve scores (Spearman's rank correlation ρ) comparable to or higher than those of previous models. On a task involving more complex phrases (subject-verb-object), our learned composed vectors achieve state-of-the-art performance ($\rho = 0.50$) with a training corpus that is an order of magnitude smaller than that used by previous work (Tsubaki et al., 2013; Van de Cruys et al., 2013). Moreover, the proposed model does not require any pre-trained word vectors produced by external models, but rather induces word vectors jointly while training.

2 Related Work

There is a large body of work on how to represent the meaning of a word in a vector space. Distributional approaches assume that the meaning of a word is determined by the contexts in which it appears (Firth, 1957). The context of a word is often defined as the words appearing in a window of fixed-length (bag-of-words) and a simple approach is to treat the co-occurrence statistics of a word w as a vector representation for w (Mitchell and Lapata, 2008; Mitchell and Lapata, 2010); alternatively, dependencies between words can be used to define contexts (Goyal et al., 2013; Erk and Padó, 2008; Thater et al., 2010).

In contrast to distributional representations, NNLMs represent words in a low-dimensional vector space (Bengio et al., 2003; Collobert et al., 2011). Recently, Mikolov et al. (2013b) and Mnih and Kavukcuoglu (2013) proposed highly scalable models to learn high-dimensional word vectors. Levy and Goldberg (2014) extended the model of Mikolov et al. (2013b) by treating syntactic dependencies as contexts.

Mitchell and Lapata (2008) investigated a variety of compositional operators to combine word vectors into phrasal representations. Among these operators, simple element-wise addition and multiplication are now widely used to represent short phrases (Mitchell and Lapata, 2010; Blacoe and Lapata, 2012). The obvious limitation with these simple approaches is that information about word order and syntactic relations is lost.

To incorporate syntactic information into composition functions, a variety of compositional models have been proposed. These include recursive neural networks using phrase-structure trees (Socher et al., 2012; Socher et al., 2013b) and models in which words have a specific form of parameters according to their syntactic roles and composition functions are syntactically dependent on the relations of input words (Baroni and Zamparelli, 2010; Grefenstette and Sadrzadeh, 2011; Hashimoto et al., 2013; Hermann and Blunsom, 2013; Socher et al., 2013a).

More recently, syntactic dependency-based compositional models have been proposed (Paperno et al., 2014; Socher et al., 2014; Tsubaki et al., 2013). One of the advantages of these models is that they are less restricted by word order. Among these, Tsubaki et al. (2013) introduced a novel compositional NNLM mainly focusing on verb-object dependencies and achieved state-of-the-art performance for the task of measuring the semantic similarity between subject-verb-object phrases.

3 PAS-CLBLM: A Compositional Log-Bilinear Language Model Using Predicate-Argument Structures

In some recent studies on representing words as vectors, word vectors are learned by solving word prediction tasks (Mikolov et al., 2013a; Mnih and Kavukcuoglu, 2013). More specifically, given target words and their context words, the basic idea is to train classifiers to discriminate between each target word and artificially induced negative target words. The feature vector of the classifiers are calculated using the context word vectors whose values are optimized during training. As a result, vectors of words in similar contexts become similar to each other.

Following these studies, we propose a novel model to jointly learn representations for words and their compositions by training word prediction classifiers using PASs. In this section, we first describe the predicate-argument structures as they serve as the basis of our model. We then introduce a Log-Bilinear Language Model using Predicate-Argument Structures (PAS-LBLM) to learn word representations using both bag-of-words and dependency-based contexts. Finally, we propose integrating compositions of words into the model. Figure 1 (b) shows the overview of the proposed model.

3.1 Predicate-Argument Structures

Due to advances in deep parsing technologies, syntactic parsers that can produce predicate-argument structures are becoming accurate and fast enough to be used for practical applications. In this work, we use the probabilistic HPSG parser *Enju* (Miyao and Tsujii, 2008) to obtain the predicate-argument structures of individual sentences. In its grammar, each word in a sentence is treated as a predicate of a certain category with zero or more arguments. Table 1 shows some examples of predicates of different categories.[1] For example, a predicate of the category *verb_arg12* expresses a verb with two arguments. A graph can be constructed by connecting words in predicate-argument structures in a sentence; in general, these graphs are acyclic.

Category	predicate	arg1	arg2
adj_arg1	heavy	rain	
noun_arg1	car	accident	
verb_arg12	cause	rain	accident
prep_arg12	at	eat	restaurant

Table 1: Examples of predicates of different categories from the grammar of the Enju parser. *arg1* and *arg2* denote the first and second arguments.

One of the merits of using predicate-argument structures is that they can capture dependencies between more than two words, while standard syntactic dependency structures are limited to dependencies between two words. For example, one of the predicates in the phrase "heavy rain caused car accidents" is the verb "cause", and it has two arguments ("rain" and "accident"). Furthermore, exactly the same predicate-argument structure (predicate: cause, first argument: rain, second argument: accident) is extracted from the passive form of the above phrase: "car accidents were caused by heavy rain". This is helpful when capturing semantic dependencies between predicates and arguments, and in extracting facts or relations described in a sentence, such as *who did what to whom*.

3.2 A Log-Bilinear Language Model Using Predicate-Argument Structures

3.2.1 PAS-based Word Prediction

The PAS-LBLM predicts a target word given its PAS-based context. We assume that each word w in the vocabulary $\mathbb{V}$ is represented with a d-dimensional vector $v(w)$. When a predicate of category c is extracted from a sentence, the PAS-LBLM computes the predicted d-dimensional vector $p(w_t)$ for the target word w_t from its context words $w_1, w_2, \ldots, w_m$:

$$p(w_t) = f\left(\sum_{i=1}^{m} h_i^c \odot v(w_i)\right), \qquad (1)$$

[1] The categories of the predicates in the Enju parser are summarized at `http://kmcs.nii.ac.jp/~yusuke/enju/enju-manual/enju-output-spec.html`.

where $h_i^c \in \mathbb{R}^{d \times 1}$ are category-specific weight vectors and $\odot$ denotes element-wise multiplication. f is a non-linearity function; in this work we define f as $\tanh$.

As an example following Figure 1 (a), when the predicate "cause" is extracted with its first and second arguments "rain" and "accident", the PAS-LBLM computes $p(\text{cause}) \in \mathbb{R}^d$ following Eq. (1):

$$p(\text{cause}) = f(h_{\text{arg1}}^{\text{verb\_arg12}} \odot v(\text{rain}) + h_{\text{arg2}}^{\text{verb\_arg12}} \odot v(\text{accident})). \quad (2)$$

In Eq. (2), the predicate is treated as the target word, and its arguments are treated as the context words. In the same way, an argument can be treated as a target word:

$$p(\text{rain}) = f(h_{\text{verb}}^{\text{verb\_arg12}} \odot v(\text{cause}) + h_{\text{arg2}}^{\text{verb\_arg12}} \odot v(\text{accident})). \quad (3)$$

Relationship to previous work. If we omit the the category-specific weight vectors h_i^c in Eq. (1), our model is similar to the CBOW model in Mikolov et al. (2013a). CBOW predicts a target word given its surrounding bag-of-words context, while our model uses its PAS-based context. To incorporate the PAS information in our model more efficiently, we use category-specific weight vectors. Similarly, the vLBL model of Mnih and Kavukcuoglu (2013) uses different weight vectors depending on the position relative to the target word. As with previous neural network language models (Collobert et al., 2011; Huang et al., 2012), our model and vLBL can use weight matrices rather than weight vectors. However, as discussed by Mnih and Teh (2012), using weight vectors makes the training significantly faster than using weight matrices. Despite the simple formulation of the element-wise operations, the category-specific weight vectors efficiently propagate PAS-based context information as explained next.

3.2.2 Training Word Vectors

To train the PAS-LBLM, we use a scoring function to evaluate how well the target word w_t fits the given context:

$$s(w_t, p(w_t)) = \tilde{v}(w_t)^{\mathrm{T}} p(w_t), \quad (4)$$

where $\tilde{v}(w_t) \in \mathbb{R}^{d \times 1}$ is the scoring weight vector for w_t. Thus, the model parameters in the PAS-LBLM are $(V, \tilde{V}, H)$. V is the set of word vectors $v(w)$, and $\tilde{V}$ is the set of scoring weight vectors $\tilde{v}(w)$. H is the set of the predicate-category-specific weight vectors h_i^c.

Based on the objective in the model of Collobert et al. (2011), the model parameters are learned by minimizing the following hinge loss:

$$\sum_{n=1}^{N} \max(1 - s(w_t, p(w_t)) + s(w_n, p(w_t)), 0), \quad (5)$$

where the negative sample w_n is a randomly sampled word other than w_t, and N is the number of negative samples. In our experiments we set $N = 1$. Following Mikolov et al. (2013b), negative samples were drawn from the distribution over unigrams that we raise to the power 0.75 and then normalize to once again attain a probability distribution. We minimize the loss function in Eq. (5) using AdaGrad (Duchi et al., 2011). For further training details, see Section 4.5.

Relationship to softmax regression models. The model parameters can be learned by maximizing the log probability of the target word w_t based on the softmax function:

$$p(w_t|context) = \frac{\exp(s(w_t, p(w_t)))}{\sum_{i=1}^{|\mathbb{V}|} \exp(s(w_i, p(w_t)))}. \quad (6)$$

This is equivalent to a softmax regression model. However, when the vocabulary $\mathbb{V}$ is large, computing the softmax function in Eq. (6) is computationally expensive. If we do not need probability distributions over words, we are not necessarily restricted to using the probabilistic expressions. Recently, several methods have been proposed to efficiently learn word representations rather than accurate language models (Collobert et al., 2011; Mikolov et al., 2013b; Mnih and Kavukcuoglu, 2013), and our objective follows the work of Collobert et al. (2011). Mikolov et al. (2013b) and Mnih and Kavukcuoglu (2013) trained their models using word-dependent scoring weight vectors which are the arguments of our scoring function in Eq. (4). During development we also trained our model using the negative sampling technique of Mikolov et al. (2013b); however, we did not observe any significant performance difference.

Intuition behind the PAS-LBLM. Here we briefly explain how each class of the model parameters of the PAS-LBLM contributes to learning word representations at each stochastic gradient decent step. The category-specific weight vectors provide the PAS information for context word vectors which we would like to learn. During training, context word vectors having the same PAS-based syntactic roles are updated similarly. The word-dependent scoring weight vectors propagate the information on which words should, or should not, be predicted. In effect, context word vectors making similar contributions to word predictions are updated similarly. The non-linear function f provides context words with information on the other context words in the same PAS. In this way, word vectors are expected to be learned efficiently by the PAS-LBLM.

3.3 Learning Composition Functions

As explained in Section 3.1, predicate-argument structures inherently form graphs whose nodes are words in a sentence. Using the graphs, we can integrate relationships between multiple predicate-argument structures into our model.

When the context word w_i in Eq. (1), excluding predicate words, has another predicate-argument of category c' as a dependency, we replace $v(w_i)$ with the vector produced by the composition function for the predicate category c'. For example, as shown in Figure 1 (b), when the first argument "rain" of the predicate "cause" is also the argument of the predicate "heavy", we first compute the d-dimensional composed vector representation for "heavy" and "rain":

$$g_{c'}(v(\text{heavy}), v(\text{rain})), \tag{7}$$

where c' is the category *adj_arg1*, and $g_{c'}$ is a function to combine input vectors for the predicate-category c'. We can use any composition function that produces a representation of the same dimensionality as its inputs, such as element-wise addition/multiplication (Mitchell and Lapata, 2008) or neural networks (Socher et al., 2012). We then replace $v(\text{rain})$ in Eq. (2) with $g_{c'}(v(\text{heavy}), v(\text{rain}))$. When the second argument "accident" in Eq. (2) is also the argument of the predicate "car", $v(\text{accident})$ is replaced with $g_{c''}(v(\text{car}), v(\text{accident}))$. c'' is the predicate category *noun_arg1*. These multiple relationships of predicate-argument structures should provide richer context information. We refer to the PAS-LBLM with composition functions as PAS-CLBLM.

3.4 Bag-of-Words Sensitive PAS-CLBLM

Both the PAS-LBLM and PAS-CLBLM can take meaningful relationships between words into account. However, at times, the number of context words can be limited and the ability of other models to take ten or more words from a fixed context in a bag-of-words (BoW) fashion could compensate for this sparseness. Huang et al. (2012) combined local and global contexts in their neural network language models, and motivated by their work, we integrate bag-of-words vectors into our models. Concretely, we add an additional input term to Eq. (1):

$$p(w_t) = f\left(\sum_{i=1}^{m} h_i^c \odot v(w_i) + h_{\text{BoW}}^c \odot v(\text{BoW})\right), \tag{8}$$

where $h_{\text{BoW}}^c \in \mathbb{R}^{d\times 1}$ are additional weight vectors, and $v(\text{BoW}) \in \mathbb{R}^{d\times 1}$ is the average of the word vectors in the same sentence. To construct the $v(\text{BoW})$ for each sentence, we average the word vectors of nouns and verbs in the same sentence, excluding the target and context words.

4 Experimental Settings

4.1 Training Corpus

We used the British National Corpus (BNC) as our training corpus, extracted 6 million sentences from the original BNC files, and parsed them using the Enju parser described in Section 3.1.

4.2 Word Sense Disambiguation Using Part-of-Speech Tags

In general, words can have multiple syntactic usages. For example, the word *cause* can be a noun or a verb depending on its context. Most of the previous work on learning word vectors ignores this ambiguity since word sense disambiguation could potentially be performed after the word vectors have been trained (Huang et al., 2012; Kartsaklis and Sadrzadeh, 2013). Some recent work explicitly assigns an independent vector for each word usage according to its part-of-speech (POS) tag (Hashimoto et al., 2013; Kartsaklis and Sadrzadeh, 2013). Alternatively, Baroni and Zamparelli (2010) assigned different forms of parameters to adjectives and nouns.

In our experiments, we combined each word with its corresponding POS tags. We used the base-forms provided by the Enju parser rather than

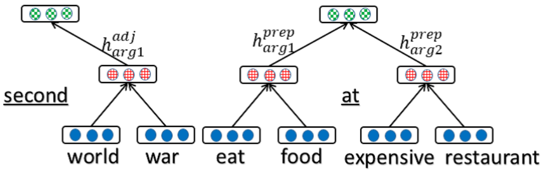


Figure 2: Two PAS-CLBLM training samples.

Model	Composition Function
Add$_\mathrm{l}$	$v(w_1) + v(w_2)$
Add$_\mathrm{nl}$	$\tanh(v(w_1) + v(w_2))$
Wadd$_\mathrm{l}$	$m^c_\mathrm{adj} \odot v(w_1) + m^c_\mathrm{arg1} \odot v(w_2)$
Wadd$_\mathrm{nl}$	$\tanh(m^c_\mathrm{adj} \odot v(w_1) + m^c_\mathrm{arg1} \odot v(w_2))$

Table 2: Composition functions used in this work. The examples are shown as the *adjective-noun* dependency between w_1 =“heavy” and w_2 =“rain”.

the surface-forms, and used the first two characters of the POS tags. For example, *VB*, *VBP*, *VBZ*, *VBG*, *VBD*, *VBN* were all mapped to *VB*. This resulted in two kinds of *cause*: *cause_NN* and *cause_VB* and we used the 100,000 most frequent lowercased word-POS pairs in the BNC.

4.3 Selection of Training Samples Based on Categories of Predicates

To train the PAS-LBLM and PAS-CLBLM, we could use all predicate categories. However, our preliminary experiments showed that these categories covered many training samples which are not directly relevant to our experimental setting, such as determiner-noun dependencies. We thus manually selected the categories used in our experiments. The selected predicates are listed in Table 1: *adj_arg1*, *noun_arg1*, *prep_arg12*, and *verb_arg12*. These categories should provide meaningful information on selectional preference. For example, the *prep_arg12* denotes prepositions with two arguments, such as “eat at restaurant” which means that the verb “eat” is related to the noun “restaurant” by the preposition “at”. Prepositions are one of the predicates whose arguments can be verbs, and thus prepositions are important in training the composition functions for (subject-) verb-object dependencies as described in the next paragraph.

Another point we had to consider was how to construct the training samples for the PAS-CLBLM. We constructed compositional training samples as explained in Section 3.3 when c' was *adj_arg1*, *noun_arg1*, or *verb_arg12*. Figure 2 shows two examples in addition to the example in Figure 1 (b). Using such training samples, the PAS-CLBLM could, for example, recognize from the two predicate-argument structures, “eat food” and “eat at restaurant”, that eating foods is an action that occurs at restaurants.

4.4 Selection of Composition Functions

As described in Section 3.3, we are free to select any composition functions in Eq. (7). To maintain the fast training speed of the PAS-LBLM, we avoid dense matrix-vector multiplication in our composition functions. In Table 2, we list the composition functions used for the PAS-CLBLM. **Add**$_\mathrm{l}$ is element-wise addition and **Add**$_\mathrm{nl}$ is element-wise addition with the non-linear function $\tanh$. The subscripts l and nl denote the words *linear* and *non-linear*. Similarly, **Wadd**$_\mathrm{l}$ is element-wise weighted addition and **Wadd**$_\mathrm{nl}$ is element-wise weighted addition with the non-linear function $\tanh$. The weight vectors $m^c_i \in \mathbb{R}^{d\times 1}$ in Table 2 are predicate-category-specific parameters which are learned during training. We investigate the effects of the non-linear function $\tanh$ for these composition functions. In the formulations of the backpropagation algorithm, non-linear functions allow the input vectors to weakly interact with each other.

4.5 Initialization and Optimization of Model Parameters

We assigned a 50-dimensional vector for each word-POS pair described in Section 4.2 and initialized the vectors and the scoring weight vectors using small random values. In part inspired by the initialization method of the weight matrices in Socher et al. (2013a), we initialized all values in the compositional weight vectors of the Wadd$_\mathrm{l}$ and Wadd$_\mathrm{nl}$ as 1.0. The context weight vectors were initialized using small random values.

We minimized the loss function in Eq. (5) using mini-batch SGD and AdaGrad (Duchi et al., 2011). Using AdaGrad, the SGD’s learning rate is adapted independently for each model parameter. This is helpful in training the PAS-LBLM and PAS-CLBLM, as they have conditionally dependent model parameters with varying frequencies.

The mini-batch size was 32 and the learning rate was 0.05 for each experiment, and no regularization was used. To verify the semantics captured by the proposed models during training and to tune the hyperparameters, we used the *WordSim-353*[2] word similarity data set (Finkelstein et al., 2001).

5 Evaluation on Phrase Similarity Tasks

5.1 Evaluation Settings

The learned models were evaluated on four tasks of measuring the semantic similarity between short phrases. We performed evaluation using the three tasks (AN, NN, and VO) in the dataset[3] provided by Mitchell and Lapata (2010), and the SVO task in the dataset[4] provided by Grefenstette and Sadrzadeh (2011).

The datasets include pairs of short phrases extracted from the BNC. AN, NN, and VO contain 108 phrase pairs of adjective-noun, noun-noun, and verb-object. SVO contains 200 pairs of subject-verb-object phrases. Each phrase pair has multiple human-ratings: the higher the rating is, the more semantically similar the phrases. For example, the subject-verb-object phrase pair of "student write name" and "student spell name" has a high rating. The pair "people try door" and "people judge door" has a low rating.

For evaluation we used the Spearman's rank correlation ρ between the human-ratings and the cosine similarity between the composed vector pairs. We mainly used *non-averaged* human-ratings for each pair, and as described in Section 5.3, we also used *averaged* human-ratings for the SVO task. Each phrase pair in the datasets was annotated by more than two annotators. In the case of averaged human ratings, we averaged multiple human-ratings for each phrase pair, and in the case of non-averaged human-ratings, we treated each human-rating as a separate annotation.

With the PAS-CLBLM, we represented each phrase using the composition functions listed in Table 2. When there was no composition present, we represented the phrase using element-wise addition. For example, when we trained the PAS-CLBLM with the composition function Wadd_{nl}, the composed vector for each phrase was computed using the Wadd_{nl} function, and when we trained the PAS-LBLM, we used the element-wise addition function. To compute the composed vectors using the Wadd_{l} and Wadd_{nl} functions, we used the categories of the predicates *adj_arg1*, *noun_arg1*, and *verb_arg12* listed in Table 1.

Model	AN	NN	VO
PAS-CLBLM (Add_{l})	**0.52**	0.44	0.35
PAS-CLBLM (Add_{nl})	**0.52**	0.46	**0.45**
PAS-CLBLM (Wadd_{l})	0.48	0.39	0.34
PAS-CLBLM (Wadd_{nl})	0.48	0.40	0.39
PAS-LBLM	0.41	0.44	0.39
word2vec	**0.52**	0.48	0.42
BL w/ BNC	0.48	**0.50**	0.35
HB w/ BNC	0.41	0.44	0.34
KS w/ ukWaC	n/a	n/a	**0.45**
K w/ BNC	n/a	n/a	0.41
Human agreement	0.52	0.49	0.55

Table 3: Spearman's rank correlation scores ρ for the three tasks: AN, NN, and VO.

As a strong baseline, we trained the *Skip-gram* model of Mikolov et al. (2013b) using the publicly available *word2vec*[5] software. We fed the POS-tagged BNC into word2vec since our models utilize POS tags and trained 50-dimensional word vectors using word2vec. For each phrase we then computed the representation using vector addition.

5.2 AN, NN, and VO Tasks

Table 3 shows the correlation scores ρ for the AN, NN, and VO tasks. *Human agreement* denotes the inter-annotator agreement. The word2vec baseline achieves unexpectedly high scores for these three tasks. Previously these kinds of models (Mikolov et al., 2013b; Mnih and Kavukcuoglu, 2013) have mainly been evaluated for word analogy tasks and, to date, there has been no work using these word vectors for the task of measuring the semantic similarity between phrases. However, this experimental result suggests that word2vec can serve as a strong baseline for these kinds of tasks, in addition to word analogy tasks.

In Table 3, **BL**, **HB**, **KS**, and **K** denote the work of Blacoe and Lapata (2012), Hermann and Blunsom (2013), Kartsaklis and Sadrzadeh (2013), and Kartsaklis et al. (2013) respectively. Among these,

[2] http://www.cs.technion.ac.il/~gabr/resources/data/wordsim353/

[3] http://homepages.inf.ed.ac.uk/s0453356/share

[4] http://www.cs.ox.ac.uk/activities/compdistmeaning/GS2011data.txt

[5] https://code.google.com/p/word2vec/

		Averaged		Non-averaged	
Model	Corpus	SVO-SVO	SVO-V	SVO-SVO	SVO-V
PAS-CLBLM (Add$_{\text{l}}$)	BNC	0.29	0.34	0.24	0.28
PAS-CLBLM (Add$_{\text{nl}}$)		0.27	0.32	0.24	0.28
PAS-CLBLM (Wadd$_{\text{l}}$)		0.25	0.26	0.21	0.23
PAS-CLBLM (Wadd$_{\text{nl}}$)		**0.42**	**0.50**	**0.34**	**0.41**
PAS-LBLM		0.21	0.06	0.18	0.08
word2vec	BNC	0.12	0.32	0.12	0.28
Grefenstette and Sadrzadeh (2011)	BNC	n/a	n/a	0.21	n/a
Tsubaki et al. (2013)	ukWaC	n/a	0.47	n/a	n/a
Van de Cruys et al. (2013)	ukWaC	n/a	n/a	0.32	0.37
Human agreement		0.75		0.62	

Table 4: Spearman's rank correlation scores ρ for the SVO task. *Averaged* denotes the ρ calculated by averaged human ratings, and *Non-averaged* denotes the ρ calculated by non-averaged human ratings.

only Kartsaklis and Sadrzadeh (2013) used the ukWaC corpus (Baroni et al., 2009) which is an order of magnitude larger than the BNC. As we can see in Table 3, the PAS-CLBLM (Add$_{\text{nl}}$) achieves scores comparable to and higher than those of the baseline and the previous state-of-the-art results. In relation to these results, the Wadd$_{\text{l}}$ and Wadd$_{\text{nl}}$ variants of the PAS-CLBLM do not achieve great improvements in performance. This indicates that simple word vector addition can be sufficient to compose representations for phrases consisting of word pairs.

5.3 SVO Task

Table 4 shows the correlation scores ρ for the SVO task. The scores ρ for this task are reported for both *averaged* and *non-averaged* human ratings. This is due to a disagreement in previous work regarding which metric to use when reporting results. Hence, we report the scores for both settings in Table 4. Another point we should consider is that some previous work reported scores based on the similarity between composed representations (Grefenstette and Sadrzadeh, 2011; Van de Cruys et al., 2013), and others reported scores based on the similarity between composed representations and word representations of landmark verbs from the dataset (Tsubaki et al., 2013; Van de Cruys et al., 2013). For completeness, we report the scores for both settings: *SVO-SVO* and *SVO-V* in Table 4.

The results show that the weighted addition model with the non-linear function tanh (PAS-CLBLM (Wadd$_{\text{nl}}$)) is effective for the more complex phrase task. While simple vector addition is sufficient for phrases consisting of word pairs, it is clear from our experimental results that they fall short for more complex structures such as those involved in the SVO task.

Our PAS-CLBLM (Wadd$_{\text{nl}}$) model outperforms the previous state-of-the-art scores for the SVO task as reported by Tsubaki et al. (2013) and Van de Cruys et al. (2013). As such, there are three key points that we would like to emphasize:

(1) the difference of the training corpus size,

(2) the necessity of the pre-trained word vectors,

(3) the modularity of deep learning models.

Tsubaki et al. (2013) and Van de Cruys et al. (2013) used the ukWaC corpus. This means our model works better, despite using a considerably smaller corpora. It should also be noted that, like us, Grefenstette and Sadrzadeh (2011) used the BNC corpus.

The model of Tsubaki et al. (2013) is based on neural network language models which use syntactic dependencies between verbs and their objects. While their novel model, which incorporates the idea of *co-compositionality*, works well with pre-trained word vectors produced by external models, it is not clear whether the pre-trained vectors are required to achieve high scores. In contrast, we have achieved state-of-the-art results without the use of pre-trained word vectors.

Despite our model's scalability, we trained 50-dimensional vector representations for words and their composition functions and achieved high scores using this low dimensional vector space.

model	d	AN	NN	VO	SVO
Add_l	50	0.52	0.44	0.35	0.24
	1000	0.51	**0.51**	0.43	0.31
Add_nl	50	0.52	0.46	0.45	0.24
	1000	0.51	0.50	0.45	0.31
Wadd_l	50	0.48	0.39	0.34	0.21
	1000	0.50	0.49	0.43	0.32
Wadd_nl	50	0.48	0.40	0.39	0.34
	1000	0.51	0.48	**0.48**	0.34

Table 5: Comparison of the PAS-CLBLM between $d = 50$ and $d = 1000$.

model	BoW	AN	NN	VO	SVO
Add_l	w/	0.52	0.44	0.35	0.24
	w/o	0.48	0.46	0.38	0.23
Add_nl	w/	0.52	0.46	**0.45**	0.24
	w/o	0.50	0.47	0.41	0.15
Wadd_l	w/	0.48	0.39	0.34	0.21
	w/o	0.47	0.39	0.38	0.21
Wadd_nl	w/	0.48	0.40	0.39	**0.34**
	w/o	0.52	0.42	0.33	0.26

Table 6: Scores of the PAS-CLBLM with and without BoW contexts.

This maintains the possibility to incorporate recently developed deep learning composition functions into our models, such as recursive neural tensor networks (Socher et al., 2013b) and co-compositional neural networks (Tsubaki et al., 2013). While such complex composition functions slow down the training of compositional models, richer information could be captured during training.

5.4 Effects of the Dimensionality

To see how the dimensionality of the word vectors affects the scores, we trained the PAS-CLBLM for each setting using 1,000-dimensional word vectors and set the learning rate to 0.01. Table 5 shows the scores for all four tasks. Note that we only report the scores for the setting *non-averaged SVO-SVO* here. As shown in Table 5, the scores consistently improved with a few exceptions. The scores $\rho = 0.51$ for the NN task and $\rho = 0.48$ for the VO task are the best results to date. However, the score $\rho = 0.34$ for the SVO task did not improve by increasing the dimensionality. This means that simply increasing the dimensionality of the word vectors does not necessarily lead to better results for complex phrases.

5.5 Effects of Bag-of-Words Contexts

Lastly, we trained the PAS-CLBLM without the bag-of-words contexts described in Section 3.4 and used 50-dimensional word vectors. As can be seen in Table 6, large score improvements were observed only for the VO and SVO tasks by including the bag-of-words contexts and the non-linearity function. It is likely that the results depend on how the bag-of-words contexts are constructed. However, we leave this line of analysis as future work. Both adjective-noun and noun-noun phrase are noun phrases, and (subject-) verb-object phrases can be regarded as complete sentences. Therefore, different kinds of context information might be required for both groups.

6 Qualitative Analysis on Composed Vectors

An open question that remains is to what extent composition affects the representations produced by our PAS-CLBLM model. To evaluate this we assigned a vector for each composed representation. For example, the adjective-noun dependency “heavy rain” would be assigned an independent vector. We added the most frequent 100,000 adjective-noun, noun-noun, and (subject-) verb-object tuples to the vocabulary and the resulting vocabulary contained 400,000 tokens (100,000+3×100,000). A similar method for treating frequent neighboring words as single words was introduced by Mikolov et al. (2013b). However, some dependencies, such as (subject-) verb-object phrases, are not always captured when considering only neighboring words.

Table 7 (*No composition*) shows some examples of predicate-argument dependencies with their closest neighbors in the vector space according to the cosine similarity. The table shows that the learned vectors of multiple words capture semantic similarity. For example, the vector of “heavy rain” is close to the vectors of words which express the phenomena *heavily raining*. The vector of “new york” captures the concept of a *major city*. The vectors of (subject-) verb-object dependencies also capture the semantic similarity, which is the main difference to previous approaches, such as that of Mikolov et al. (2013b), which only consider neighboring words. These results suggest that the PAS-CLBLM can learn meaningful composition

Query	No composition	Composition
	rain	rain
(AN)	thunderstorm	sunshine
heavy	downpour	storm
rain	blizzard	drizzle
	much rain	chill
	general manager	executive
(AN)	vice president	director
chief	executive director	representative
executive	project manager	officer
	managing director	administrator
	second war	war
(NN)	plane crash	world
world	riot	race
war	last war	holocaust
	great war	warfare
	oslo	york
(NN)	paris	toronto
new	birmingham	paris
york	moscow	edinburgh
	madrid	glasgow
	make order	make
(VO)	carry survey	allow
make	pay tax	demand
payment	pay	produce
	impose tax	bring
	achieve objective	solve
(VO)	bridge gap	alleviate
solve	improve quality	overcome
problem	deliver information	resolve
	encourage development	circumvent
	hold meeting	take
(SVO)	event take place	get
meeting	end season	win
take	discussion take place	put
place	do work	gain

Table 7: Nearest neighbor vectors for multiple words. POS-tags are not shown for simplicity.

category	predicate	arg1	arg2
adj_arg1	2.38	**6.55**	-
noun_arg1	3.37	**5.60**	-
verb_arg12	**6.78**	2.57	2.18

Table 8: L2-norms of the 50-dimensional weight vectors of the composition function Wadd$_{\mathrm{nl}}$.

functions since the composition layers receive the same error signal via backpropagation.

We then trained the PAS-CLBLM using Wadd$_{\mathrm{nl}}$ to learn composition functions. Table 7 (*Composition*) shows the nearest neighbor words for each composed vector, and as we can see, the learned composition function emphasizes the head words and captures some sort of semantic similarity. We then inspected the L2-norms of the weight vectors of the composition function. As shown in Table 8, head words are strongly emphasized. Emphasizing head words is helpful in representing composed meanings, but in the case of verbs it may not always be sufficient. This can be observed in Table 3 and Table 4, which demonstrates that verb-related tasks are more difficult than noun-phrase tasks.

While *No composition* captures the semantic similarity well using independent parameters, there is the issue of data sparseness. As the size of the vocabulary increases, the number of tuples of word dependencies increases rapidly. In this experiment, we used only the 300,000 most frequent tuples. In contrast to this, the learned composition functions can capture similar information using only word vectors and a small set of predicate categories.

7 Conclusion and Future Work

We have presented a compositional log-bilinear language model using predicate-argument structures that incorporates both bag-of-words and dependency-based contexts. In our experiments the learned composed vectors achieve state-of-the-art scores for the task of measuring the semantic similarity between short phrases. For the subject-verb-object phrase task, the result is achieved without any pre-trained word vectors using a corpus an order of magnitude smaller than that of the previous state of the art. For future work, we will investigate how our models and the resulting vector representations can be helpful for other unsupervised and/or supervised tasks.

Acknowledgments

We thank the anonymous reviewers for their helpful comments and suggestions. This work was supported by JSPS KAKENHI Grant Number 13F03041.

References

Marco Baroni and Roberto Zamparelli. 2010. Nouns are vectors, adjectives are matrices: Representing adjective-noun constructions in semantic space. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1183–1193.

Marco Baroni, Silvia Bernardini, Adriano Ferraresi, and Eros Zanchetta. 2009. The wacky wide web: a collection of very large linguistically processed web-crawled corpora. *Language Resources and Evaluation*, 43(3):209–226.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. *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.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12:2493–2537.

John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12:2121–2159.

Katrin Erk and Sebastian Padó. 2008. A structured vector space model for word meaning in context. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 897–906.

Lev Finkelstein, Gabrilovich Evgenly, Matias Yossi, Rivlin Ehud, Solan Zach, Wolfman Gadi, and Ruppin Eytan. 2001. Placing search in context: the concept revisited. In *Proceedings of the Tenth International World Wide Web Conference*.

John Rupert Firth. 1957. A synopsis of linguistic theory 1930-55. In *Studies in Linguistic Analysis*, pages 1–32.

Kartik Goyal, Sujay Kumar Jauhar, Huiying Li, Mrinmaya Sachan, Shashank Srivastava, and Eduard Hovy. 2013. A structured distributional semantic model : Integrating structure with semantics. In *Proceedings of the Workshop on Continuous Vector Space Models and their Compositionality*, pages 20–29.

Edward Grefenstette and Mehrnoosh Sadrzadeh. 2011. Experimental support for a categorical compositional distributional model of meaning. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1394–1404.

Kazuma Hashimoto, Makoto Miwa, Yoshimasa Tsuruoka, and Takashi Chikayama. 2013. Simple customization of recursive neural networks for semantic relation classification. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1372–1376.

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.

Eric Huang, Richard Socher, Christopher Manning, and Andrew 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 (Volume 1: Long Papers)*, pages 873–882.

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.

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.

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.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. In *Proceedings of Workshop at the International Conference on Learning Representations*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26*, pages 3111–3119.

Jeff Mitchell and Mirella Lapata. 2008. Vector-based models of semantic composition. In *Proceedings of 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 236–244.

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

Yusuke Miyao and Jun'ichi Tsujii. 2008. Feature forest models for probabilistic hpsg parsing. *Computational Linguistics*, 34(1):35–80.

Andriy Mnih and Koray Kavukcuoglu. 2013. Learning word embeddings efficiently with noise-contrastive estimation. In *Advances in Neural Information Processing Systems 26*, pages 2265–2273.

Andriy Mnih and Yee Whye Teh. 2012. A fast and simple algorithm for training neural probabilistic language models. In John Langford and Joelle Pineau, editors, *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*, ICML '12, pages 1751–1758.

Denis Paperno, Nghia The Pham, and Marco Baroni. 2014. A practical and linguistically-motivated approach to compositional distributional semantics. In

Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 90–99.

Richard Socher, Brody Huval, Christopher D. Manning, and Andrew Y. Ng. 2012. Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1201–1211.

Richard Socher, John Bauer, Christopher D. Manning, and Ng Andrew Y. 2013a. 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.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013b. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642.

Richard Socher, Quoc V. Le, Christopher D. Manning, and Andrew Y. Ng. 2014. Grounded compositional semantics for finding and describing images with sentences. *Transactions of the Association for Computational Linguistics*, 2:207–218.

Stefan Thater, Hagen Fürstenau, and Manfred Pinkal. 2010. Contextualizing semantic representations using syntactically enriched vector models. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 948–957.

Masashi Tsubaki, Kevin Duh, Masashi Shimbo, and Yuji Matsumoto. 2013. Modeling and learning semantic co-compositionality through prototype projections and neural networks. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 130–140.

Joseph Turian, Lev-Arie Ratinov, and Yoshua Bengio. 2010. Word representations: A simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 384–394.

Peter D. Turney and Patrick Pantel. 2010. From frequency to meaning: Vector space models of semantics. *Journal of Artificial Intelligence Research*, 37(1):141–188.

Tim Van de Cruys, Thierry Poibeau, and Anna Korhonen. 2013. A tensor-based factorization model of semantic compositionality. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1142–1151.