Learning Embeddings for Transitive Verb Disambiguation by Implicit Tensor Factorization

Kazuma Hashimoto and Yoshimasa Tsuruoka
The University of Tokyo, 3-7-1 Hongo, Bunkyo-ku, Tokyo, Japan
{hassy,tsuruoka}@logos.t.u-tokyo.ac.jp

Abstract

We present an implicit tensor factorization method for learning the embeddings of transitive verb phrases. Unlike the implicit matrix factorization methods recently proposed for learning word embeddings, our method directly models the interaction between predicates and their two arguments, and learns verb phrase embeddings. By representing transitive verbs as matrices, our method captures multiple meanings of transitive verbs and disambiguates them taking their arguments into account. We evaluate our method on a widely-used verb disambiguation task and three phrase similarity tasks. On the disambiguation task, our method outperforms previous state-of-the-art methods. Our experimental results also show that adjuncts provide useful information in learning the meanings of verb phrases.

1 Introduction

There is a growing interest in learning vector-space representations of words and phrases using large training corpora in the field of Natural Language Processing (NLP) (Mikolov et al., 2013; Mitchell and Lapata, 2010). The phrase representations are usually computed by composition models that combine the meanings of words into the meanings of phrases. While some studies focus on representing entire phrases or sentences using syntactic structures (Hermann and Blunsom, 2013; Socher et al., 2011), others focus on representing the meaning of transitive verb phrases (Grefenstette and Sadrzadeh, 2011; Grefenstette et al., 2013; Kartsaklis et al., 2012).

In this paper, we investigate vector-space representations of transitive verb phrases. The meaning of a transitive verb is often ambiguous and disambiguated by its arguments, i.e., subjects and objects. Investigation of transitive verb phrases should therefore provide insights into how composition models can capture such semantic interactions between words. Moreover, in practice, capturing the meanings of transitive verb phrases should be useful in many real-world NLP applications such as semantic retrieval (Miyao et al., 2006) and question answering (*Who did What to Whom?*) (Srihari and Li, 2000).

There are several approaches to representing transitive verb phrases in a vector space using large unannotated corpora. One is based on tensor calculus (Grefenstette and Sadrzadeh, 2011; Kartsaklis et al., 2012; Van de Cruys et al., 2013) and another is based on neural networks (Hashimoto et al., 2014; Muraoka et al., 2014; Tsubaki et al., 2013). In the tensor-based methods, transitive verbs are represented as matrices, and they are constructed by using the pre-trained word embeddings of their subjects and objects. One limitation of this approach is that the embeddings of subject-verb-object phrases are computed statically, i.e., the composition process and the embedding (or matrix) construction process are conducted separately. In the neural network-based methods, the embeddings of words and phrases can be learned jointly (Hashimoto et al., 2014). However, the strong interaction between verbs and their arguments is not fully captured in their method because it relies on shallow neural networks using diagonal weight matrices which are designed to work on large training corpora.

To bridge the gap between the two approaches, we present an implicit tensor factorization method for learning the embeddings of transitive verb phrases. We assume a three-mode tensor in which the value of each element represents the level of plausibility of a tuple of a predicate and its two arguments (Van de Cruys et al., 2013). We then implicitly factorize the tensor into three latent factors, namely one predicate tensor and two argument matrices. This is motivated by the recently proposed implicit matrix factorization methods for learning word embeddings (Levy and Goldberg, 2014; Mikolov et al., 2013). Our method trains matrices representing predicates and embeddings of their arguments so that they maximize the accuracy of predicting the plausibility of the predicate-argument tuples in the training corpus. The transitive verb matrices and the embeddings of their subject and object are thus jointly learned. Furthermore, this method allows us to exploit the role of prepositional adjuncts when learning the meaning of verb phrases by modeling the relationship between prepositions and verb phrases.

Our experimental results show that our method enables predicates and their arguments to strongly interact with each other and that adjuncts are useful in learning the meaning of verb phrases. We evaluate our method using a widely-used verb disambiguation task and three phrase similarity tasks. On the disambiguation dataset provided by Grefenstette and Sadrzadeh (2011), we have achieved a Spearman's rank correlation score of 0.614, which is significantly higher than the state of the art (0.456). This result demonstrates that the direct interaction between verbs and their arguments is important in tackling verb disambiguation tasks. Qualitative evaluation further shows that the meanings of ambiguous verbs can be disambiguated according to their arguments and the learned verb matrices capture multiple meanings of transitive verbs.

2 Method

To learn the embeddings of transitive verb phrases, we focus on the role of adjuncts, which optionally complement the meaning of the verb phrases. For example, in the following sentence, the prepositional phrase starting from the preposition "in" is an adjunct of the verb "make":

> An importer might be able to make payment in his own domestic currency.

The transitive verb "make" is inherently ambiguous, but this sentence tells us that the action ex-

Predicate	Argument 1	Argument 2
make	an importer	payment
in	make payment	his own domestic currency

Table 1: Example output from the Enju parser.

Predicate	Argument 1	Argument 2
make	importer	payment
in	importer make payment	currency

Table 2: Modified examples.

pressed by the verb phrase "make payment" is carried out by means of a currency. If we further observe the verb phrase "pay money" with a similar adjunct in another sentence, those two sentences tell us that the two phrases "make payment" and "pay money" are semantically similar to each other. We therefore expect such prepositional adjuncts to be useful in learning the meaning of verb phrases. In the disambiguation processes, strong interactions between transitive verbs and their arguments are desirable as with the method in Tsubaki et al. (2013). More specifically, the meaning of "make" changes according to its object "payment" and the meaning of "pay" changes according to its object "money".

We use the probabilistic HPSG parser *Enju*[1] (Miyao and Tsujii, 2008) to identify transitive verbs with their subjects and objects, and as adjuncts, we also extract prepositional phrases with transitive verbs. In the grammar of the Enju parser, each word in a sentence is a predicate with zero or more arguments, i.e., prepositions, too, are treated as predicates.

In the example sentence shown above, the transitive verb "make" and the preposition "in" are predicates which take two arguments. Table 1 shows the output from the Enju parser. In this example, the transitive verb "make" takes two arguments: the first argument (the subject) is the noun phrase "an importer" and the second argument (the object) is the noun "payment". The preposition "in" also takes two arguments: the first argument is the verb phrase "make payment" and the second argument is the noun phrase "his own domestic currency". For simplicity we use only the head word of each noun phrase, so the subjects and objects of the transitive verbs are nouns and the second arguments of the prepositions are also nouns. We further modify the output by incorporating the

[1]`http://kmcs.nii.ac.jp/enju/`.

subject for each verb phrase which is the first argument of prepositions when the subject exists [2]. Therefore, the words used in this paper are verbs, nouns, and prepositions. Table 2 shows the modified output of the examples in Table 1.

To model the co-occurrence statistics of predicate-argument structures, we follow Van de Cruys et al. (2013) and assume a three-mode tensor, which is just a three-dimensional array, $\mathcal{T} \in \mathbb{R}^{|\mathbb{P}|\times|\mathbb{A}_1|\times|\mathbb{A}_2|}$ in which plausibility scores are stored as real values. $\mathbb{P}$ is the set of predicates of a particular category in the training corpus, $\mathbb{A}_1$ is the set of the first argument of the predicates in $\mathbb{P}$, and $\mathbb{A}_2$ is the set of the second argument. When treating transitive verbs as predicates, $\mathbb{A}_1$ is the set of their subjects and $\mathbb{A}_2$ is the set of their objects. Table 1 shows an example, where "make", "an importer", and "payment" are a member of $\mathbb{P}$, $\mathbb{A}_1$, and $\mathbb{A}_2$, respectively. The plausibility score $\mathcal{T}(i,j,k)$ corresponds to the tuple of the i-th ($1 \leq i \leq |\mathbb{P}|$) predicate having the j-th ($1 \leq j \leq |\mathbb{A}_1|$) first-argument and the k-th ($1 \leq k \leq |\mathbb{A}_2|$) second-argument. The larger the value of $\mathcal{T}(i,j,k)$ is, the more plausible the tuple (i,j,k) is. In the above example, if the tuple (i,j,k) corresponds to "make", "an importer", and "payment" and i' corresponds to "eat", the value of $\mathcal{T}(i,j,k)$ is expected to be larger than that of $\mathcal{T}(i',j,k)$.

As with Van de Cruys et al. (2013), we factorize the large three-mode tensor $\mathcal{T}$ into three factors:

- three-mode tensor $\mathcal{P} \in \mathbb{R}^{|\mathbb{P}|\times d\times d}$,
- matrix $\mathbf{A}_1 \in \mathbb{R}^{d\times|\mathbb{A}_1|}$, and
- matrix $\mathbf{A}_2 \in \mathbb{R}^{d\times|\mathbb{A}_2|}$.

The dimensionality d is a hyperparameter that determines the size of the latent factors. Using these factors, we can compute a plausibility score:

$$\mathcal{T}(i,j,k) = \mathbf{a}_1(j)^{\mathrm{T}}\mathbf{P}(i)\mathbf{a}_2(k) \tag{1}$$

where $\mathbf{a}_1(j) \in \mathbb{R}^{d\times 1}$ and $\mathbf{a}_2(k) \in \mathbb{R}^{d\times 1}$ are the j-th and k-th column vectors of $\mathbf{A}_1$ and $\mathbf{A}_2$, respectively, and $\mathbf{P}(i) \in \mathbb{R}^{d\times d}$ is the i-th slice, which is just a matrix, of $\mathcal{P}$. $\mathbf{a}_1(j)^{\mathrm{T}}$ is the transpose of $\mathbf{a}_1(j)$. By this tensor factorization, each predicate in $\mathbb{P}$ is represented with a matrix, which we call a predicate matrix, and each argument in $\mathbb{A}_1$ and $\mathbb{A}_2$ is represented with a vector, which we call an argument embedding. As with Hashimoto et al. (2014), arguments are not restricted to words in our method, and thus we compute argument embeddings using a composition function when the arguments consist of more than two words.

[2]Subjects can be absent. For example, in the sentence "Learning word embedings is interesting" the subject of the transitive verb "learn" is absent.

To learn the predicate matrices and argument embeddings, we define a plausibility judgment task by using a cost function for each predicate-argument tuple observed in the training corpus. For each predicate-argument tuple (i,j,k), the cost function $E(i,j,k)$ is defined as follows:

$$\begin{aligned} -\log\sigma(\mathcal{T}(i,j,k)) &- \log(1-\sigma(\mathcal{T}(i',j,k))) \\ &- \log(1-\sigma(\mathcal{T}(i,j',k))) \\ &- \log(1-\sigma(\mathcal{T}(i,j,k'))) \end{aligned} \tag{2}$$

where $i' \in \mathbb{P}$ is a randomly drawn predicate, and $j' \in \mathbb{A}_1$ and $k' \in \mathbb{A}_2$ are randomly drawn arguments. $\sigma(x)$ is the logistic function, so the cost function $E(i,j,k)$ in Eq. (2) measures whether we can discriminate between the plausible tuple and other three implausible tuples by means of logistic regressions. We follow Mikolov et al. (2013) to draw the random predicates and arguments according to their frequencies weighted by an exponent of 0.75 and ensure that each of the randomly generated tuples is not observed in the corpus. The overall objective function is defined as the sum of the cost functions for all observed predicate-argument tuples and minimized by AdaGrad (Duchi et al., 2011) in a mini-batch setting.

The partial derivative $\frac{\partial E(i,j,k)}{\partial \mathbf{P}(i)}$ for updating the model parameters is computed as follows:

$$\begin{aligned} \frac{\partial E(i,j,k)}{\partial \mathbf{P}(i)} = (\sigma(\mathcal{T}(i,j,k)) - 1)\mathbf{a}_1(j) \otimes \mathbf{a}_2(k) + \\ \sigma(\mathcal{T}(i,j',k))\mathbf{a}_1(j') \otimes \mathbf{a}_2(k) + \\ \sigma(\mathcal{T}(i,j,k'))\mathbf{a}_1(j) \otimes \mathbf{a}_2(k') \end{aligned} \tag{3}$$

where $\otimes$ denotes the outer-product of two vectors. Similarly, the partial derivatives $\frac{\partial E(i,j,k)}{\partial \mathbf{a}_1(j)}$ and $\frac{\partial E(i,j,k)}{\partial \mathbf{a}_2(k)}$ are computed as follows:

$$\begin{aligned} \frac{\partial E(i,j,k)}{\partial \mathbf{a}_1(j)} = (\sigma(\mathcal{T}(i,j,k)) - 1)\mathbf{P}(i)\mathbf{a}_2(k) + \\ \sigma(\mathcal{T}(i',j,k))\mathbf{P}(i')\mathbf{a}_2(k) + \\ \sigma(\mathcal{T}(i,j,k'))\mathbf{P}(i)\mathbf{a}_2(k') \end{aligned} \tag{4}$$

$$\frac{\partial E(i,j,k)}{\partial \mathbf{a}_2(k)} = (\sigma(\mathcal{T}(i,j,k)) - 1)\mathbf{P}(i)^{\mathrm{T}}\mathbf{a}_1(j) + \sigma(\mathcal{T}(i',j,k))\mathbf{P}(i')^{\mathrm{T}}\mathbf{a}_1(j) + \sigma(\mathcal{T}(i,j',k))\mathbf{P}(i)^{\mathrm{T}}\mathbf{a}_1(j') \tag{5}$$

which can be used to learn the composition function using the backpropagation algorithm if the arguments are not words. When the arguments are words, we then use the partial derivatives to directly update the argument embeddings. $\mathbf{P}(i')$, $\mathbf{a}_1(j')$, and $\mathbf{a}_2(k')$ are also updated but for the sake of brevity the partial derivatives for them are not shown here. Equation (3) shows that a predicate matrix is updated to capture the information about which argument pairs are or are not relevant to the predicate. Argument embeddings are learned to capture similar information.

2.1 Transitive Verb Phrases with Adjuncts

While our method is applicable to any categories of predicates which take two arguments, in this paper, we focus on learning the embeddings of transitive verb phrases by treating transitive verbs and prepositions as predicates. Thus, we factorize two tensors $\mathcal{T}_v$ and $\mathcal{T}_p$ for transitive verbs and prepositions, respectively. $\mathcal{T}_v$ is factorized into a verb tensor $\mathcal{V}$ (corresponding to $\mathcal{P}$), a subject matrix $\mathbf{S}$ (corresponding to $\mathbf{A}_1$), and an object matrix $\mathbf{O}$ (corresponding to $\mathbf{A}_2$). To compute argument embeddings composed by subject-verb-object tuples, we use the *copy-subject* function in Kartsaklis et al. (2012):

$$\mathbf{s}(m) \odot (\mathbf{V}(l)\mathbf{o}(n)) \tag{6}$$

where $\mathbf{V}(l)$ is a verb matrix, $\mathbf{s}(m)$ is a subject embedding, and $\mathbf{o}(n)$ is an object embedding. $\odot$ denotes the element-wise multiplication of two vectors. The composed verb phrase embeddings are taken as the first arguments of the prepositions. The copy-subject function is also used to compute verb-object phrase embeddings by omitting the subject embedding in Eq. (6):

$$\mathbf{V}(l)\mathbf{o}(n) \tag{7}$$

Compared with other composition functions defined in Kartsaklis et al. (2012), such as the copy-object function, the copy-subject function allows us to compute embeddings for both of subject-verb-object and verb-object phrases.

In the case of the copy-subject function, assuming that Eq. (4) is defined as $\boldsymbol{\delta}_1$, the subject embedding in Eq. (6) is updated using the following partial derivative:

$$\frac{\partial E(i,j,k)}{\partial \mathbf{s}(m)} = \boldsymbol{\delta}_1 \odot (\mathbf{V}(l)\mathbf{o}(n)) \tag{8}$$

We then define $\boldsymbol{\delta}_2$ as follows:

$$\boldsymbol{\delta}_2 = \boldsymbol{\delta}_1 \odot \mathbf{s}(m) \tag{9}$$

and update the verb matrix and object embedding using $\boldsymbol{\delta}_2$:

$$\frac{\partial E(i,j,k)}{\partial \mathbf{V}(l)} = \boldsymbol{\delta}_2 \mathbf{o}(n)^{\mathrm{T}} \tag{10}$$

$$\frac{\partial E(i,j,k)}{\partial \mathbf{o}(n)} = \mathbf{V}(l)^{\mathrm{T}} \boldsymbol{\delta}_2 \tag{11}$$

The model parameters used in the composition function are shared across the overall proposed method. That is, the verb matrices and subject/object embeddings are used for computing the composed embeddings and the plausibility scores in Eq. (1).

2.2 Relationship to Previous Work

Representing transitive verbs with matrices and computing transitive verb phrase embeddings have been proposed by Grefenstette and Sadrzadeh (2011) and others (Kartsaklis et al., 2012; Milajevs et al., 2014; Polajnar et al., 2014). All of them first construct word embeddings by using existing methods and then compute or learn transitive verb matrices. This kind of approach requires one to figure out which word embeddings are suitable for each method or task (Milajevs et al., 2014). By contrast, our method does not require any other word embedding methods and instead jointly learns word embeddings and matrices from scratch, which saves us from the time-consuming process to test which word representations learned by existing methods are suitable for which composition models. Moreover, our method *learns* the embeddings of transitive verb phrases by using adjuncts rather than statically computing them using learned word embeddings and matrices as done in the previous work.

The information stored in the verb matrices learned by Eq. (3) is similar to that in Grefenstette and Sadrzadeh (2011). In Grefenstette and Sadrzadeh (2011), a verb matrix is computed by the sum of the outer-products of the embeddings of its subject-object pairs observed in the corpus. By using such matrices, Kartsaklis et al. (2012) proposed the copy-subject function which has proven effective in representing transitive verb phrases. Using the copy-subject function is therefore a reasonable choice for our composition function.

The use of adjuncts constructed by prepositional phrases for learning verb phrase embeddings has been presented in Hashimoto et al. (2014). However, they used a variety of categories of predicates simultaneously, and thus it is not clear how adjuncts are useful in improving the embeddings of transitive verb phrases. In this paper, we use only transitive verbs and prepositions and clarify the effects of adjuncts. Moreover, the interactions between predicates and their arguments are weak in their method because their method relies on shallow neural networks using diagonal weight matrices. In contrast, our method allows the predicates to directly interact with their arguments.

The way of factorizing the three-mode tensors is based on Van de Cruys et al. (2013). The main difference between our method and theirs is that our method can treat phrases as the arguments. Their method is based on co-occurrence count statistics, and thus it is not straightforward to modify their method to treat phrases as well as words.

Our implicit tensor factorization method is motivated by Levy and Goldberg (2014). They introduced a way to interpret the recently developed word embedding learning method (Mikolov et al., 2013) by using matrix factorization. While their method only produces embeddings of single tokens, our method jointly learns word and phrase embeddings by focusing on the relationship between predicates and their two arguments.

3 Experimental Settings

3.1 Training Corpora

We separately used two corpora as our training corpora. The first one is the British National Corpus (BNC), from which we extracted 6 million sentences. The second one is a snapshot of the English Wikipedia[3] (enWiki) from November 2013. We extracted 80 million sentences from the original Wikipedia file. We then used the Enju parser (Miyao and Tsujii, 2008) to parse all the extracted sentences.

[3]http://dumps.wikimedia.org/enwiki/

Using the parsing results, we constructed the vocabulary for each training corpus. To be more specific, we used the 100,000 most frequent base-form words paired with their corresponding part-of-speech tags in each corpus. Using verbs, nouns, and prepositions in the vocabulary, we extracted predicate-argument tuples whose predicate categories are *verb_arg12* or *prep_arg12* defined in the Enju parser. We then pre-processed the output as shown in Table 2. Consequently, BNC consists of about 1.38 million instances (1.23 million types) for the verb data and about 0.93 million instances (0.88 million types) for the preposition data, and enWiki consists of about 23.6 million instances (15.8 million types) for the verb data and about 17.3 million instances (13.5 million types) for the preposition data. We call the verb data **SVO** and the combination of the two data **SVOPN** [4].

For each corpus, we randomly split the data into the training data (80%), the development data (10%), and the test data (10%). We used the development data for tuning hyperparameters to be used in downstream NLP tasks. When splitting the data, we ensured that each type of predicate-argument tuples appeared in only one of the three parts. Hence, for example, instances in the test data do not appear in either of the training or the development data.

To evaluate our method on the plausibility judgment task, for each predicate-argument tuple type in the development and the test data, we randomly sampled implausible tuples N times in the same way as defining the cost function in Eq. (2). That is, we prepared N sets of the development and the test data. For each set of the development and the test data, we calculated the accuracy of the plausibility judgment task; concretely, for each type of predicate-argument tuples (i, j, k), we evaluated whether $\mathcal{T}(i, j, k)$ is larger than all of $\mathcal{T}(i', j, k)$, $\mathcal{T}(i, j', k)$, and $\mathcal{T}(i, j, k')$ and then calculated the ratio of the number of types counted as correct to the total number of the types in the development or the test data. Finally, we calculated the average accuracy of the N set. For BNC, we set N to 50 and for enWiki we set N to 10.

[4]The training data, the training code, and the learned model parameters used in this paper are publicly available at http://www.logos.t.u-tokyo.ac.jp/~hassy/publications/cvsc2015/

3.2 Initialization and Hyperparameters

We initialized the noun embeddings, the verb matrices, and the preposition matrices with zero-mean gaussian noise with a variance of $\frac{1}{d}$, $\frac{1}{d^2}$, and $\frac{1}{d^2}$, respectively. The hyperparameters for training the embeddings and the matrices are the embedding dimensionality d, the learning rate α for AdaGrad (Duchi et al., 2011), the mini-batch size, and the number of iterations n over the training data. In our preliminary experiments, we have found that varying the mini-batch size is not so influential in our experimental results. We thus fixed the mini-batch size to 100. For other hyperparameters, we set d to $\{25, 50, 100\}$, α to $\{0.01, 0.02, 0.04, 0.06, 0.08, 0.1\}$, and the maximum number of n to 20. We selected the values of the hyperparameters so that the accuracy of the plausibility task was maximized on the development data described in Section 3.1.

3.3 Baseline Method

We mainly compared our method with the method called *PAS-CLBLM* in Hashimoto et al. (2014) since PAS-CLBLM is designed to learn composed representations as well as word embeddings using a variety of predicate-argument structures. PAS-CLBLM is modeled as a word predication model using predicate-argument structures, which means that, as with our method, the training relies on the co-occurrence statistics of predicate-argument structures. PAS-CLBLM achieved state-of-the-art results on transitive verb phrase similarity tasks. To train PAS-CLBLM, we used the same data described in Section 3.1. We selected the Wadd$_{\text{nl}}$ function in PAS-CLBLM to compute the embedding of each subject-verb-object tuple (i, j, k):

$$\tanh(\mathbf{w}_{\text{s}} \odot \mathbf{s}(j) + \mathbf{w}_{\text{v}} \odot \mathbf{v}(i) + \mathbf{w}_{\text{o}} \odot \mathbf{o}(k)) \quad (12)$$

where $\mathbf{w}_{\text{s}}, \mathbf{w}_{\text{v}}, \mathbf{w}_{\text{o}} \in \mathbb{R}^{d\times 1}$ are the weight vectors (or the diagonal weight matrices) for composition and $\mathbf{s}(j), \mathbf{v}(i), \mathbf{o}(k) \in \mathbb{R}^{d\times 1}$ are the embeddings of the subjects, verbs, and objects, respectively. PAS-CLBLM has the same hyperparameters as our method described in Section 3.2. We used the development data for tuning the hyperparameters and added $d = 200$ to the candidate values for d since PAS-CLBLM is computationally less expensive than our method. We thus evaluated PAS-CLBLM also on the plausibility judgment task. Concretely, for each type of predicate-argument tuples (i, j, k) in the development data, the tuple is counted as correct when the predication scores for i, j, and k are larger than those for i', j', and k', respectively.

	d	BNC Acc. (%)	enWiki Acc. (%)
Our method	25	57.44 (0.11)	64.77 (0.03)
	50	**57.80** (0.11)	66.98 (0.03)
	100	57.48 (0.10)	**68.18** (0.03)
PAS-CLBLM	25	54.44 (0.14)	60.40 (0.03)
	50	**55.69** (0.13)	63.42 (0.02)
	100	55.66 (0.12)	64.81 (0.02)
	200	55.48 (0.15)	**65.20** (0.03)

Table 3: Evaluation results on the plausibility judgment task on the SVO development data.

4 Results and Discussion

We first tuned the hyperparameters in both our method and the baseline method using the plausibility judgment task. Table 3 shows the average accuracy with the standard deviation for each dimensionality on the SVO development data[5]. As shown in the table, our method outperforms PAS-CLBLM on both BNC and enWiki. The number of the model parameters in PAS-CLBLM ($d = 200$) is larger than that of the model parameters in our method ($d = 50$). This result demonstrates that the model architecture itself is more important than the number of the model parameters. The results on the SVO test data were 57.76% (our method, $d = 50$) and 55.66% (PAS-CLBLM, $d = 50$) for BNC. For enWiki, the results were 68.18% (our method, $d = 100$) and 65.19% (PAS-CLBLM, $d = 200$). We observed a similar trend on the SVOPN data and in the next section, for each embedding dimensionality, we used the model parameters which performed best on the plausibility task.

4.1 Evaluation on Transitive Verb Tasks

4.1.1 Evaluation Settings

We evaluated the learned embeddings of transitive verbs using a transitive verb disambiguation task and three tasks for measuring the semantic similarity between transitive verb phrases. Each phrase

[5]Van de Cruys (2014) reported much higher accuracy in a similar evaluation setting with a neural network model, but as discussed in Chambers and Jurafsky (2010), this is because using the uniform distribution over words for producing implausible tuples leads to optimistic results.

[6]We replicated the results reported in their paper using the model parameters publicly provided at `http://www.logos.t.u-tokyo.ac.jp/~hassy/publications/emnlp2014/`.

			Dis.	Phrase similarity		
	Data	*d*	GS'11	ML'10	KS'13	KS'14
Our method	SVO	25	0.410	0.511	0.392	0.440
		50	0.374	0.550	0.164	0.290
		100	0.373	0.474	0.312	0.418
	SVOPN	25	**0.574**	0.543	0.439	0.432
		50	0.535	**0.586**	0.403	0.397
		100	0.508	0.545	**0.487**	**0.517**
PAS-CLBLM	SVO	25	0.270	0.601	0.592	0.722
		50	**0.412**	0.581	0.523	0.721
		100	0.390	0.463	0.465	0.699
		200	0.369	0.458	0.434	0.602
	SVOPN	25	0.241	0.562	0.550	0.715
		50	0.281	**0.605**	**0.590**	**0.760**
		100	0.337	0.593	0.585	0.758
		200	0.342	0.561	0.549	0.744
Milajevs et al. (2014)			0.456	n/a	n/a	0.732
Hashimoto et al. (2014)[6]			0.422	0.669	0.612	0.770
Polajnar et al. (2014)			0.35	n/a	0.58	n/a

Table 4: Results for the transitive verb tasks using the BNC data.

			Dis.	Phrase similarity		
	Data	*d*	GS'11	ML'10	KS'13	KS'14
Our method	SVO	25	0.438	0.403	0.255	0.406
		50	0.480	0.416	0.359	0.481
		100	0.433	0.392	0.239	0.409
	SVOPN	25	0.576	0.435	0.372	0.555
		50	**0.614**	0.495	**0.422**	**0.566**
		100	0.576	**0.558**	0.420	0.548
PAS-CLBLM	SVO	25	0.342	0.500	0.407	0.624
		50	0.313	0.527	0.502	0.710
		100	0.358	0.534	0.470	0.655
		200	0.361	0.535	0.459	0.653
	SVOPN	25	0.171	0.571	**0.583**	0.697
		50	0.320	0.501	0.518	0.729
		100	0.321	**0.606**	0.540	0.742
		200	**0.374**	0.588	0.515	**0.744**
Milajevs et al. (2014)			0.456	n/a	n/a	0.732
Hashimoto et al. (2014)			0.422	0.669	0.612	0.770
Polajnar et al. (2014)			0.35	n/a	0.58	n/a

Table 5: Results for the transitive verb tasks using the enWiki data.

pair in the four datasets is paired with multiple human ratings: the higher the rating is, the more semantically similar the phrases are. To evaluate the learned verb phrase embeddings on each dataset, we used the Spearman's rank correlation between the human ratings and the cosine similarity between the phrase embeddings. We calculated the correlation scores using averaged human ratings. Each phrase pair in the datasets was annotated by more than two annotators and we took the average of the multiple human ratings for each phrase pair.

Transitive verb disambiguation. The first dataset **GS'11** is provided by Grefenstette and Sadrzadeh (2011). GS'11 consists of pairs of transitive verbs and each verb pair takes the same subject and object. As discussed in previous work (Kartsaklis and Sadrzadeh, 2013; Milajevs et al., 2014; Polajnar et al., 2014), GS'11 has an aspect of a verb sense disambiguation task. For example, the transitive verb "run" is known as a polysemous word and this task requires one to identify the meanings of "run" and "operate" are similar to each other when taking "people" as their subject and "company" as their object. In the same setting, however, the meanings of "run" and "move" are not similar to each other. The task is suitable for evaluating our method since our method allows verbs and their subjects and objects to multiplicatively interact with each other.

Transitive verb phrase similarity. The other datasets are **ML'10** provided by Mitchell and Lapata (2010), **KS'13** provided by Kartsaklis and Sadrzadeh (2013), and **KS'14** provided by Kartsaklis and Sadrzadeh (2014). ML'10 consists of pairs of verb-object phrases and KS'13 complements ML'10 by incorporating an appropriate subject for each verb-object phrase. KS'14 is the reannotated version of KS'13 using a cloud sourcing service. Unlike GS'11, these three datasets require one to capture the topical similarity rather than the disambiguation aspect (Polajnar et al., 2014).

4.1.2 Result Overview

Table 4 and 5 show the evaluation results using BNC and enWiki, respectively. The results are shown for each method, data type, and embedding dimensionality. These tables also show the results from other work (Hashimoto et al., 2014; Milajevs et al., 2014; Polajnar et al., 2014) on the same tasks while the training settings, such as the corpus and information used in the training, are different from those in this work. However, the evaluation settings are the same with those in the previous work. That is, in the previous work, averaged human ratings were used to evaluate the Spearman's rank correlation scores, similarity scores between subject-verb-object phrases were used for GS'11, KS'13, and KS'14, and similarity scores between verb-object phrases were used for ML'10.

Effects of using adjuncts. Except for the results for GS'11 using PAS-CLBLM, the correla-

	Our method				PAS-CLBLM	
	SVO		SVOPN		SVOPN	
make money	make dollar	make saving	make cash	earn billion	make cash	make penny
	make pay	use money	make dollar	earn million	make dollar	make baht
	make profit	make cent	make profit	make gamble	make yen	make salary
	make cash	do business	earn baht	make pound	make pay	make profit
	earn profit	sell coin	earn pound	earn earning	make fund	make rupee
make payment	make repayment	make expenditure	make loan	pay reimbursement	make loan	make cost
	make loan	pay subsidy	make repayment	pay remuneration	make repayment	make receipt
	pay amount	pay deposit	pay fine	make raise	make compensation	make guarantee
	make offer	make transaction	pay amount	pay cost	make expense	make rebate
	pay compensation	pay donor	pay surcharge	pay fee	make debt	make purchase
make use	use material	use approach	use number	use one	make usage	make sort
	use type	use method	use concept	use element	make placement	make size
	use concept	use technique	use approach	use set	make kind	make utilization
	use form	use instrument	use method	use system	make quality	make redundancy
	use one	use system	use model	use type	make alternative	make handling

Table 6: Nearest neighbor verb-object phrases.

tion scores consistently improve when using the SVOPN data compared with using the SVO data, which shows using adjuncts is helpful in learning the meanings of verb phrases. Using the SVO data alone, verb phrase embeddings themselves are not directly learned but computed separately. By contrast, the SVOPN data provides the opportunity for learning verb phrase embeddings.

Effects of the training corpora. In previous work on learning and evaluating word embeddings, it is generally observed that increasing the training data results in better results. However, as opposed to our expectation, Table 4 and 5 show that using enWiki does not necessarily lead to better results. A possible explanation is that the nature of the training corpus matters the most. The usage of each word depends on the training corpora, and at least for these verb sense tasks, the size of BNC is sufficient and the nature of BNC fits these tasks.

4.1.3 Disambiguation Task

Our method outperforms both the baseline and the previous state of the art for GS'11, which demonstrates that our method better handles the disambiguation of transitive verbs. This result is somewhat expected since our method provides stronger interaction between predicates and their arguments than the baseline method.

Table 6 shows some examples[7] of verb-object phrases with their nearest neighbor ones in the embedding space according to the cosine similarity. For our method, we show the results of using the SVO and SVOPN data, and for PAS-CLBLM, we show the results of using the SVOPN data. In each setting, we used the enWiki data with $d = 50$.

Table 6 clearly shows the difference between our method and the baseline method. In our method, the meaning of "make" becomes close to those of "earn", "pay", and "use" when taking "money", "payment", and "use", respectively, as its object. By contrast, PAS-CLBLM simply emphasizes the head word "make". In previous work, it is also reported that the weighed addition composition functions put more weight on head words (Hashimoto et al., 2014; Muraoka et al., 2014; Socher et al., 2013). As opposed to these previous methods, our method has the ability of selecting the meaning of transitive verbs according to their objects.

Table 6 also shows that the phrase embeddings in our method are influenced by using the adjunct data (i.e., the SVOPN data). For example, in the example of "make money", the results for using the SVO data include "use money" as the nearest neighbors. When using the SVOPN data, the focus seems to shift to the true meaning of "make money".

4.1.4 Phrase Similarity Task

In the phrase similarity tasks, our method compares favorably to PAS-CLBLM for ML'10, but PAS-CLBLM outperforms our method for KS'13 and KS'14. These results are consistent with those in previous work. In Milajevs et al. (2014) and Polajnar et al. (2014), using the simplest composition function (the element-wise vector addition) achieves much better correlation scores than other tensor-based complex composition functions. These results indicate that our method is suitable for capturing the disambigua-

[7]The verb-object phrase "make use" is the part of the idiomatic expression "make use of".

Verb		Nearest neighbors
run	27th col.	operate, execute, insert, hold, grid, produce, add, assume, manage, render
	34th row	release, operate, create, override, govern, oversee, distribute, host, organize
	all	operate, start, manage, own, launch, continue, establish, open, maintain
encode	28th row	denature, transfect, phosphorylate, polymerize, subtend, acid
	39th row	format, store, decode, embed, concatenate, encrypt, memorize
	all	concatenate, permute, phosphorylate, quantize, composite, transfect, transduce

Table 7: Nearest neighbor verbs.

tion rather than capturing the topical similarity between phrases.

4.2 Qualitative Evaluation on Verb Matrices

Finally, we inspect the learned verb matrices using the SVO data of enWiki with $d = 50$. Compared with the word embeddings, the verb matrices have two-dimensional structure. According to Eq. (3), each row vector and each column vector in a verb matrix are updated to capture the information about what subject-object pairs are relevant (or irrelevant) to the verb.

Table 7 shows the nearest neighbor verbs using the cosine similarity between row (or column) vectors in the verb matrices. For reference, we also show the results using the vectorized representation of the verb matrices (denoted as "all" in the table). While the entire matrices capture the general similarity between verbs as with word embeddings, some specific rows (or columns) capture the multiple meanings of usages of the verbs.

5 Related Work

Based on the distributional hypothesis (Firth, 1957), various methods for word embeddings have been actively studied (Levy and Goldberg, 2014; Mikolov et al., 2013). Recent studies also investigate how to learn phrase and/or sentence embeddings using syntactic structures and word embeddings (Socher et al., 2011). Along the same line of research, there is a growing body of work on representing transitive verb phrases using word embeddings (Grefenstette and Sadrzadeh, 2011; Hashimoto et al., 2014; Kartsaklis et al., 2012; Tsubaki et al., 2013). Those studies can be split into two approaches: one is based on tensor calculus and the other is based on neural networks.

In contrast to the recent studies on word embeddings, the tensor-based methods represent words with tensors which are not limited to vectors. That is, higher order tensors such as matrices and three-mode tensors are also used. In the case of representing transitive verb phrases, for example, each transitive verb is represented as a matrix and each noun is represented as a vector in Grefenstette and Sadrzadeh (2011). Based on Coecke et al. (2010), Grefenstette and Sadrzadeh (2011) presented a method for calculating a verb matrix using word embeddings of its observed subjects and objects. The word embeddings were constructed by the method in Mitchell and Lapata (2008). Grefenstette and Sadrzadeh (2011) then introduced composition functions using the verb matrices and the noun embeddings. Their approach has been followed by some recent studies (Kartsaklis et al., 2012; Milajevs et al., 2014; Polajnar et al., 2014; Van de Cruys et al., 2013).

In the neural network-based methods each word is usually represented with a vector. Tsubaki et al. (2013) presented a neural network language model focusing on the binary relationship between verbs and their objects. Their co-compositionality method enables verb embeddings to be multiplicatively influenced by the objects, and vice versa. Subsequently, Hashimoto et al. (2014) introduced a method which jointly learns word and phrase embeddings by using a variety of predicate-argument structures. While their method achieves state-of-the-art results on phrase similarity tasks, the interaction between predicates and their arguments is weak.

6 Conclusion and Future Work

We have presented an implicit matrix factorization method for learning the embeddings of transitive verb phrases. The verb matrices learned by our method capture the multiple meanings of transitive verbs and we have shown that adjuncts play an important role in learning the meanings of transitive verb phrases. In our experiments, our method outperforms the previous state of the art on a transitive verb disambiguation task. In future work, we will investigate how the learned phrase embeddings improve real-world NLP applications.

Acknowledgments

We thank the anonymous reviewers for their helpful comments and suggestions.

References

Nathanael Chambers and Daniel Jurafsky. 2010. Improving the Use of Pseudo-Words for Evaluating Selectional Preferences. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 445–453.

Bob Coecke, Mehrnoosh Sadrzadeh, and Stephen Clark. 2010. Mathematical Foundations for a Compositional Distributional Model of Meaning. *CoRR*, abs/1003.4394.

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.

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

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.

Edward Grefenstette, Georgiana Dinu, Yao-Zhong Zhang, Mehrnoosh Sadrzadeh, and Marco Baroni. 2013. Multi-Step Regression Learning for Compositional Distributional Semantics. In *Proceedings of the 10th International Conference on Computational Semantics*, pages 131–142.

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*, pages 1544–1555.

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.

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 and Mehrnoosh Sadrzadeh. 2014. A Study of Entanglement in a Categorical Framework of Natural Language. In *Proceedings of the 11th Workshop on Quantum Physics and Logic (QPL)*, Kyoto, Japan, June.

Dimitri Kartsaklis, Mehrnoosh Sadrzadeh, and Stephen Pulman. 2012. A Unified Sentence Space for Categorical Distributional-Compositional Semantics: Theory and Experiments. In *Proceedings of the 24th International Conference on Computational Linguistics*, pages 549–558.

Omer Levy and Yoav Goldberg. 2014. Neural Word Embedding as Implicit Matrix Factorization. In *Advances in Neural Information Processing Systems 27*, pages 2177–2185.

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 26*, 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*, pages 708–719.

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, March.

Yusuke Miyao, Tomoko Ohta, Katsuya Masuda, Yoshimasa Tsuruoka, Kazuhiro Yoshida, Takashi Ninomiya, and Jun'ichi Tsujii. 2006. Semantic Retrieval for the Accurate Identification of Relational Concepts in Massive Textbases. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 1017–1024.

Masayasu Muraoka, Sonse Shimaoka, Kazeto Yamamoto, Yotaro Watanabe, Naoaki Okazaki, and Kentaro Inui. 2014. Finding The Best Model Among Representative Compositional Models. In *Proceedings of the 28th Pacific Asia Conference on Language, Information, and Computation*, pages 65–74.

Tamara Polajnar, Laura Rimell, and Stephen Clark. 2014. Using Sentence Plausibility to Learn the Semantics of Transitive Verbs. In *Proceedings of Workshop on Learning Semantics at the 2014 Conference on Neural Information Processing Systems*.

Richard Socher, Eric H. Huang, Jeffrey Pennin, Christopher D Manning, and Andrew Y. Ng. 2011. Dynamic Pooling and Unfolding Recursive Autoencoders for Paraphrase Detection. In *Advances in Neural Information Processing Systems 24*, pages 801–809.

Richard Socher, John Bauer, Christopher D. Manning, and Ng Andrew Y. 2013. Parsing with Compositional Vector Grammars. In *Proceedings of the*

51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 455–465.

Rohini Srihari and Wel Li. 2000. A Question Answering System Supported by Information Extraction. In *Proceedings of the Sixth Conference on Applied Natural Language Processing*, pages 166–172.

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.

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.

Tim Van de Cruys. 2014. A Neural Network Approach to Selectional Preference Acquisition. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 26–35.