TENSOR-BASED PREPOSITION REPRESENTATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Prepositions are among the most frequent words. Good prepositional representation is of great syntactic and semantic interest in computational linguistics. Existing methods on preposition representation either treat prepositions as content words (e.g., word2vec and GloVe) or depend heavily on external linguistic resources including syntactic parsing, training task and dataset-specific representations. In this paper we use *word-triple* counts (one of the words is a preposition) to capture the preposition's interaction with its head and children. Prepositional embeddings are derived via tensor decompositions on a large unlabeled corpus. We reveal a new geometry involving Hadamard products and empirically demonstrate its utility in paraphrasing of phrasal verbs. Furthermore, our prepositional embeddings are used as simple features to two challenging downstream tasks: preposition selection and prepositional attachment disambiguation. We achieve comparable to or better results than state of the art on multiple standardized datasets.

1 INTRODUCTION

Prepositions are a linguistically closed class comprising some of the most frequent words; they play an important role in the English language since they encode rich syntactic and semantic information. Many preposition-related tasks still remain unsolved in computational linguistics because of their polysemous nature and flexible usage patterns. An accurate understanding and representation of prepositions' linguistic role is key to several important NLP tasks such as grammatical error correction and prepositional phrase attachment. A first-order approach is to represent prepositions as real-valued vectors via word embeddings such as word2vec Mikolov et al. (2013) and GloVe Pennington et al. (2014).

Word embeddings have brought a renaissance in NLP research; they have been very successful in capturing word similarities as well as analogies (both syntactic and semantic) and are now mainstream in nearly all downstream NLP tasks (such as question-answering). Despite this success, no specific properties of word embeddings of prepositions have been highlighted in the literature. Indeed, many of the common prepositions have very similar vector representations as shown in Table 1 for preposition vectors trained using word2vec and GloVe.While this suggests that using available representations for prepositions diminishes the distinguishing feature between prepositions, one could hypothesize that this is primarily because standard word embedding algorithms treat prepositions no different from other content words such as verbs and nouns, i.e., embeddings are created based on co-occurrences with other words. However, prepositions are very frequent and co-occur with nearly all words, which means that their co-occurrence ought to be treated differently.

Table 1: Cosine Similarity of Centered Prepositions

prepositions	word2vec	GloVe	tensor
(above, below)	0.85	0.78	0.22
(above, beneath)	0.40	0.45	0.15
(after, before)	0.83	0.70	0.44
(after, during)	0.56	0.42	0.16
(amid, despite)	0.47	0.37	0.12
(amongst, besides)	0.46	0.37	0.21
(beneath, inside)	0.55	0.47	0.29

Modern descriptive linguistic theory proposes to understand a preposition via its interactions with *both* the head (attachment) and child (complement) Huddleston (1984); DeCarrico (2000). This theory naturally suggests that one should count co-occurrences of a given preposition with *pairs* of neighboring words. One way of achieving this would be by considering a *tensor* of triples ($word_1$, $word_2$, preposition), where we do not restrict $word_1$ and $word_2$ to be head- and child- words; instead we model a preposition's

interaction with *all* pairs of neighboring words via a *slice* of a tensor X – the slice is populated by word co-occurrences restricted to a context window of the specific preposition. Thus, the tensor dimension is $V \times V \times K$ where V is the vocabulary and K is the number of prepositions; since $K \approx 50$, we note that $V \gg K$.

Using such a representation, we find that the resulting tensor is low rank and extract embeddings for both preposition and non-preposition words using a combination of standard ideas from word representations (such as weighted spectral decomposition as in GloVe Pennington et al. (2014)) and tensor decompositions (alternating least squares (ALS) methods Sharan & Valiant (2017)). The preposition embeddings are discriminative, see preposition similarity of the tensor embedding in Table 1. We demonstrate that the resulting representation for prepositions captures the core linguistic property of prepositions. We do this using both intrinsic evaluations and downstream tasks, where we provide new state-of-the-art results on well-known NLP tasks involving prepositions.

Intrinsic evaluations: We show that the *Hadamard* product of the embeddings of a verb and a preposition closely approximates the representation of this phrasal verb's paraphrase. Example: $v_{\text{made}} \odot v_{\text{from}} \approx v_{\text{produced}}$ where $\odot$ represents the Hadamard product (i.e., pointwise multiplication) of two vectors; this approximation does not hold for the standard word embeddings of prepositions (word2vec or GloVe). We provide a mathematical interpretation for this new geometry as well as empirically demonstrate the generalization on a new data set of compositional phrasal verbs.

Extrinsic evaluations: Our preposition embeddings are used as features for a simple classifier in two well-known challenging downstream NLP classification tasks. In both tasks, we perform comparable to or strictly better than the state-of-the-art on multiple standardized datasets.

Preposition selection: The choice of prepositions significantly influences (and is governed by) the semantics of the context they occur in. Furthermore, the prepositional choice is usually very subtle (and consequently is one of the most frequent error types made by second language English speakers Leacock et al. (2010)). This task tests the choice of a preposition in a large set of contexts ($7,000$ instances of both CoNLL-2013 and SE datasets Prokofyev et al. (2014)). Our approach achieves 6% and 2% absolute improvement over the previous state-of-the-art on the respective datasets.

Prepositional attachment disambiguation: Prepositional phrase attachment is a common cause of structural ambiguity in natural language. In the sentence "Pierre Vinken *joined the board as a voting member*", the prepositional phrase "as a voting member" can attach to either "joined" (the VP) or "the board" (the NP); in this case the VP attachment is correct. Despite extensive study over decades of research, prepositional attachment continues to be a major source of syntactic parsing errors Brill & Resnik (1994); Kummerfeld et al. (2012); de Kok & Hinrichs (2016). We use our prepositional representations as simple features to a standard classifier on this task. Our approach tested on a widely studied standard dataset Belinkov et al. (2014) achieves 89% accuracy, essentially the same performance as state-of-the-art (90% accuracy). It is noteworthy that while the state-of-the-art results are obtained with significant linguistic resources, including syntactic parsers and WordNet, our approach does not rely on these resources to achieve a comparable performance.

We emphasize two aspects of our contributions:
(1) It is folklore within the NLP community that representations via *pairwise* word counts capture much of the benefits of the unlabeled sentence-data; example: Sharan & Valiant (2017) reports that their word representations via word-triple counts are better than others, but still significantly worse than regular word2vec representations. One of our main observations is that considering word-triple counts makes most (linguistic) sense when one of the words is a preposition. Furthermore, the sparsity of the corresponding tensor is no worse than the sparsity of the regular word co-occurrence matrix (since prepositions are so frequent and co-occur with essentially every word). Taken together, these two points strongly suggest the benefits of tensor representations in the context of prepositions.
(2) The word and preposition representations via tensor decomposition are simple features leading to a standard classifier. In particular, we do not use syntactic parsing (which many prior methods have relied on) or handcrafted features Prokofyev et al. (2014) or train task-specific representations on the annotated training dataset Belinkov et al. (2014). The simplicity combined with our strong empirical results (new state-of-the-art results on long standing datasets) lends credence to the strength of the prepositional representations found via tensor decompositions.

2 METHOD

We begin with a description of how the tensor with triples (word,word,preposition) is formed and empirically show that its slices are low-rank. Next, we derive low dimensional vector representations for words and prepositions via appropriate tensor decomposition methods.

Tensor creation: Suppose that K prepositions are in the preposition set $P = \{p_1, \ldots, p_K\}$; here K is 49 in our preposition selection task, and 76 in the attachment disambiguation task. The vocabulary, the set of all words except prepositions, contains N words $V = \{w_1, \ldots, w_N\}$, and $N \approx 1M$. We generate a third order tensor $\mathbf{X}_{N \times N \times (K+1)}$ from WikiCorpus Al-Rfou et al. (2013) in the following way. We say two words co-occur if they appear within distance t of each other in a sentence. For $k \leq K$, the entry $\mathbf{X}_{ijk}$ is the number of occurrences where word w_i co-occurs with preposition p_k, and w_j also co-occurs with preposition p_k in the same sentence, and this is counted across all sentences in a large WikiCorpus. Here we use a window of size $t = 3$. There are also a number of words which do not occur in the context of any preposition. To make full use of the data, we add an extra slice $\mathbf{X}[:, :, K+1]$: the entry $\mathbf{X}_{ij(K+1)}$ is the number of occurrences where w_i co-occurs with w_j (within distance $2t = 6$) but at least one of them is not within a distance of t of any preposition.

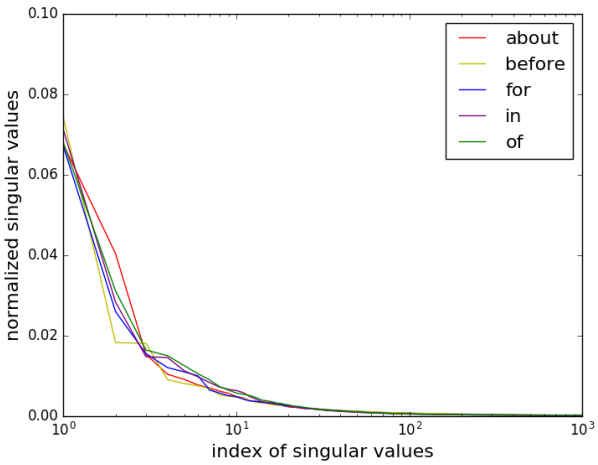

Figure 1: Decaying normalized singular values of slices.

Note that the preposition window of 3 is smaller than the word window of 6, since it is known that the interaction between prepositions and neighboring words usually weakens more sharply with the distance as compared to content words Hassani & Lee (2017).

Empirical properties of $\mathbf{X}$: We find that the tensor X is very sparse – only 1% of tensor elements are non-zeros. Furthermore, every slice $\log(1 + \mathbf{X}[:, :, k])$ is low rank (here the logarithm is applied componentwise to every entry of the tensor slice). We choose slices corresponding to prepositions "about", "before","for", "in" and "of", and plot their normalized singular values in Figure 1. We see that the singular values decay dramatically, suggesting the low rank structure in each slice.

Tensor decomposition: We combine standard ideas from word embedding algorithms and tensor decomposition algorithms to arrive at the low rank approximation to the tensor $\log(1 + \mathbf{X})$. In particular, we consider two separate methods:

1. *Alternating Least Squares (ALS)*. A generic method to decompose the tensor into its modes is via the CANDECOMP/PARAFAC (CP) decomposition Kolda & Bader (2009). The tensor $\log(1 + \mathbf{X})$ is decomposed into three modes: $\mathbf{U}_{d \times N}$, $\mathbf{W}_{d \times N}$ and $\mathbf{Q}_{d \times (K+1)}$, based on the solutions to the optimization problem (1). Here $\mathbf{u}_i$, $\mathbf{w}_i$ and $\mathbf{q}_i$ are the i-th column of U, W and Q, respectively.

$$L = \min_{\mathbf{U},\mathbf{W},\mathbf{Q}} \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{k=1}^{K+1} \left(\langle \mathbf{u}_i, \mathbf{w}_j, \mathbf{q}_k \rangle - \log(1 + \mathbf{X}_{ijk}) \right)^2, \tag{1}$$

where $\langle \mathbf{a}, \mathbf{b}, \mathbf{c} \rangle = \mathbf{1}^t (\mathbf{a} \odot \mathbf{b} \odot \mathbf{c})$ is the inner product of three vectors $\mathbf{a}, \mathbf{b}$ and $\mathbf{c}$. Here $\mathbf{1}$ is the column vector of all ones and $\odot$ refers to the Hadamard product. We can interpret the columns of U as the word representations and the columns of Q as the preposition representations, each of dimension d (equal to 200 in this paper). There are several algorithmic solutions to this optimization problem in the literature, most of which are based on alternating least squares methods Kolda & Bader (2009); Comon et al. (2009); Anandkumar et al. (2014) and we employ a recent one named Orth-ALS Sharan & Valiant (2017) in this paper. Orth-ALS periodically orthogonalizes decomposed components while fixing two modes and updating the remaining one. It is supported by theoretical guarantees and empirically outperforms standard ALS method in different applications.

Table 2: Paraphrasing of Prepositional Phrases

Phrase	involved in	made from	approved of	dreamed of	blocked off	sparked off
Paraphrase	included	produced	issued	wants	intercepted	prompted

2. *Weighted Decomposition (WD)*: Based on ideas from the literature on word embedding algorithms, we also consider weighting different elements of the tensors differently in order to reduce the effect of the large dynamic range of the tensor values. Specifically, we employ the GloVe objective function to our tensor model and minimize the objective function (2):

$$L_{\text{weighted}} = \min_{\mathbf{U},\mathbf{W},\mathbf{Q}} \sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=1}^{K+1} \omega_{ijk} \left(\langle \mathbf{u}_i, \mathbf{w}_j, \mathbf{q}_k \rangle + b_{Ui} + b_{Wj} + b_{Qk} - \log(\mathbf{X}_{ijk}+1)\right)^2, \quad (2)$$

where b_{Ui} is the scalar bias for word i in matrix U. Similarly, b_{Wj} is the bias for word j in matrix W, and b_{Qk} is for preposition k in matrix Q. Bias terms are learned to minimize the loss function. Here ω_{ijk} is the weight assigned to each tensor element X_{ijk}, and we use the weighting proposed by GloVe: $\omega_{ijk} = \min\left(\left(\frac{\mathbf{X}_{ijk}}{x_{\max}}\right)^{\alpha}, 1\right)$. We set hyperparameters $x_{\max} = 10$, and $\alpha = 0.75$ in this work. We solve this optimization problem via standard gradient descent, arriving at word representations $\mathbf{U}$ and tensor representations $\mathbf{Q}$.

3 Geometry of Phrasal Verbs

Representation Interpretation Suppose that we have a phrase (h, p_i, c) where h, p_i and c are head word, preposition $i(1 \leq i \leq K)$ and child respectively. A phrase example is *split off something*. The inner product of word vectors of h, p_i and c reflects how frequently h and c cooccur in the context of p. It also reflects how cohesive the triple is.

Recall that there is an extra $(K+1)-$th slice that describes the word co-occurrences outside the preposition window, which considers cases such as the phrasal verb (v, c) where v and c are the verb and the child. The verb phrase *divide something* is equivalent to the phrase *split off something*. For any word c that fits in this phrase semantically, we can expect that

$$\langle \mathbf{u}_h, \mathbf{q}_i, \mathbf{w}_c \rangle \approx \langle \mathbf{u}_v, \mathbf{q}_{K+1}, \mathbf{w}_c \rangle.$$

In other words $\mathbf{u}_h \odot \mathbf{q}_i \approx \mathbf{u}_v \odot \mathbf{q}_{K+1}$, where $\mathbf{a} \odot \mathbf{b}$ denotes the pointwise multiplication (Hadamard product) of vectors $\mathbf{a}$ and $\mathbf{b}$. This suggests that we could paraphrase the verb phrase (h, p_i) by finding the verb v such that $\mathbf{u}_v \odot \mathbf{q}_{(K+1)}$ is closest to $\mathbf{u}_{(h)} \odot \mathbf{q}_i$.

$$\text{paraphrase} = \arg\min_v \|\mathbf{u}_v \odot \mathbf{q}_{K+1} - \mathbf{u}_h \odot \mathbf{q}_i\|. \quad (3)$$

Well-trained embeddings should be able to capture the relation between the prepositional phrases and their equivalent phrasal verbs. In Table 2, we list seven paraphrases of verb phrases, as generated from the weighted tensor decomposition. A detailed list of paraphrases on a new dataset of compositional verb phrases is available in Table 10 in Appendix B, where we also compare paraphrasing results using regular word embeddings and via both addition and Hadamard product operations. The combination of tensor representations and Hadamard product results in vastly superior paraphrasing.

In the next two sections, we evaluate tensor-based preposition embeddings in the context of two important NLP downstream tasks: preposition selection and preposition attachment disambiguation. In this work, we use WikiCorpus as the training corpus for different sets of embeddings. We train tensor embeddings with both Orth-ALS and weighted decomposition. The implementation of Orth-ALS is built upon he SPLATT toolkit Smith & Karypis (2016). We perform orthogonalization in the first 5 iterations in Orth-ALS decomposition, and the training is completed when its performance stabilizes. As for the weighted decomposition, we train for 20 iterations, and its hyperparameters are set as $x_{\max} = 10$, and $\alpha = 0.75$.

We also include two baselines, word2vec's CBOW model and GloVe, for comparison. We set 20 training iterations to both models. Hyperparameters in word2vec are set as: window size=6, negative sampling=25 and down sampling=1e-4. Hyperparameters in GloVe are set as: window size=6, x_{max}=10, α=0.75 and minimum word count=5. We note that all the representations in this study – word2vec, GloVe and our tensor embedding – are of dimension 200.

4 DOWNSTREAM APPLICATION: PREPOSITION SELECTION

The detection and correction of grammatical errors is an important task in NLP. Second language learners tend to make more mistakes and in particular, prepositional errors make up about 13% of all errors, ranking the second among most common error types Leacock et al. (2010). This is due to the fact that prepositions are highly polysemous and have flexible usage. Accurate preposition selection needs to well capture the interaction between preposition and its context. This task is natural to evaluate how well the lexical interactions are captured by different methods.

Task. Given a sentence in English with a preposition, we either replace the preposition (to the correct one) or retain it. For example, "to" should be corrected as "of" in the sentence "It can save the effort to carrying a lot of cards". Formally, there is a closed set of preposition candidates $P = \{p_1, \ldots, p_m\}$. A preposition p is used in a sentence s consisting of words $s = \{\ldots, w_{-2}, w_{-1}, p, w_1, w_2, \ldots\}$. If used incorrectly, we need to replace p by another preposition $\hat{p} \in P$ based on the context.

Table 3: Dataset Statistics

FEC	# of sent	27119
	# of prep	60279
	error ratio	4.8
CoNLL	# of sent	1375
	# of prep	3241
	error ratio	4.7
SE	# of sent	5917
	# of prep	15814
	error ratio	38.2

Table 4: Performance on Preposition Selection

Dataset	Method	Precision	Recall	F1 score
CoNLL	state-of-the-art	0.2592	0.3611	0.3017
	word2vec	0.1558	0.1579	0.1569
	GloVe	0.1538	0.1578	0.1558
	Our method (ALS)	0.3355	0.3355	0.3355
	Our method (WD)	0.3590	0.3684	**0.3636**
SE	state-of-the-art	0.2704	0.2961	0.2824
	word2vec	0.2450	0.2585	0.2516
	GloVe	0.2454	0.2589	0.2520
	Our method (ALS)	0.2958	0.3146	**0.3049**
	Our method (WD)	0.2899	0.3055	0.2975

Dataset. We use training data from Cambridge First Certificate in English (FCE) exam, the same data used by state-of-the-art on preposition error correction Prokofyev et al. (2014). As for test data, we use two datasets: CoNLL-2013 and Stack Exchange (SE) dataset. CoNLL dataset on preposition error correction is published by CoNLL 2013 shared task Ng et al. (2014), collected from 50 essays written by 25 non-native English learners at a university. SE dataset consists of texts generated by non-native speakers on Stack Exchange website. Detailed statistics are shown in Table 3. We focus on the most frequent 49 prepositions listed in Appendix A.

Evaluation metric. Three metrics, precision, recall and F1 score (harmonic mean of precision and recall) are used to evaluate preposition selection performance.

Our algorithm. We first preprocess the dataset by removing articles, determiners and pronouns, and take a context window of 3. We divide the task into two steps: error identification and error correction. Firstly, we decide whether a preposition is used correctly in the context. If not, we suggest another preposition as replacement in the second step. Identification step uses only three features: cosine similarity between the current preposition embedding and the average context embedding, rank of the preposition in terms of cosine similarity, and probability that this preposition is not changed in training corpus. We build a decision tree classifier with these three features and find that we can identify errors with 98% F1 score in the CoNLL dataset and 96% in the SE dataset.

When it comes to error correction, we only focus on identified errors in the first stage. Suppose that the original preposition is q, and the candidate preposition is p. Word vectors in the left context window are averaged as left context embedding v_ℓ, and right vectors are averaged as right context embedding v_r. We have following features:
(1) embedding features: $\mathbf{v}_\ell, \mathbf{v}_p$ and $\mathbf{v}_r$;
(2) pair similarity: maximum of the similarity of the preposition between left and right context, i.e., pair sim $= \max\left(\frac{\mathbf{v}_\ell^T \mathbf{v}_p}{\|\mathbf{v}_\ell\| \cdot \|\mathbf{v}_r\|}, \frac{\mathbf{v}_r^T \mathbf{v}_p}{\|\mathbf{v}_r\| \cdot \|\mathbf{v}_r\|}\right)$;
(3) triple similarity: triple sim $= \frac{\langle \mathbf{v}_\ell, \mathbf{v}_p, \mathbf{v}_r \rangle}{\|\mathbf{v}_\ell\|_3 \cdot \|\mathbf{v}_p\|_3 \cdot \|\mathbf{v}_r\|_3}$;
(4) Confusion probability: the probability that q is replaced by p in the training data.

A two-layer feedforward neural network (FNN) with hidden sizes of 500 and 10 is trained with these features to score prepositions in each sentence. The one with the highest score is the suggested edit.

Baseline. State-of-the-art on preposition selection uses n-gram statistics from a large corpus Prokofyev et al. (2014). Features such as pointwise mutual information (PMI) and part-of-speech tags are fed into a supervised scoring system. Prepositions with highest score are chosen as suggested ones.

The performance is affected by both the system architecture and features. To evaluate the benefits brought by our tensor embedding-based features, we also consider other baselines which have the same two-step architecture whereas features are generated from word2vec and GloVe embeddings. These baselines allow us to compare the representation power independent of the classifier.

Table 5: Ablation Analysis in Preposition Selection

removed feature		left context embedding	prep embedding	right context embedding	pair similarity	triple similarity	confusion score
CoNLL	Precision	0.1558	0.2662	0.3117	0.3247	0.3247	0.3506
	Recall	0.1579	0.2697	0.3158	0.3289	0.3289	0.3553
	F1 score	0.1569	0.2680	0.3137	0.3268	0.3268	0.3529
SE	Precision	0.2587	0.2796	0.2649	0.2658	0.2647	0.1993
	Recall	0.2743	0.2964	0.2801	0.2818	0.2807	0.2114
	F1 score	0.2663	0.2877	0.2726	0.2735	0.2725	0.2052

Result. We compare our methods against baselines mentioned above in Table 4. As is seen, tensor embeddings achieve the best performance among all approaches. In particular, tensor with weighted decomposition has the highest F1 score on CoNLL dataset, 6% improvement over the state of the art. The tensor with ALS decomposition performs the best on SE dataset, achieving 2% improvement. We also note that with the same architecture, tensor embeddings perform much better than word2vec and GloVe embeddings on both datasets. It validates the representation power of tensor embeddings.

To have a deeper insight into feature importance in the preposition selection task, we also perform an ablation analysis of the tensor method with weighted decomposition as shown in Table 5. We remove one feature each time, and report the performance achieved by remaining features. It is found that left context is the most important feature in CoNLL dataset, whereas confusion score is the most important in SE dataset. Pair similarity and triple similarity are less important compared with other features. This is because the neural network could learn lexical similarity from embedding features, and diminishes the importance of similarity features.

Discussion. Now we analyze the reasons why our approach selects wrong prepositions in some sentences. (1) *Limited context window*. We focus on the local context within preposition's window. In some cases, we find that head words might be out of the context window. In the sentence "prevent more of this kind of tragedy *to* happening" where *to* should be corrected as *from*. Given the context window of 3, we cannot get the lexical clues provided by *prevent*, which leads to the selection error in our approach. (2) *Preposition selection requires more context*. Even when the context window contains all words on which the preposition depends, it still may not be sufficient to select the right preposition. For example, in the sentence "it is controlled by bad men *in* a not good purpose" where our approach replaces the preposition *in* with the preposition *on* given the high frequency of the phrase "on purpose". The correct preposition should be *for* based on the whole sentence.

5 Downstream Application: Prepositional Attachment

In this section, we discuss prepositional phrase (PP) attachment disambiguation, a well-studied, but still open, hard task in syntactic parsing. A prepositional phrase usually consists of head words, a preposition and child words. An example is "he saw an elephant with long tusks", where "with" is attached to the noun "elephant". In another example "he saw an elephant with his telescope", "with" is attached to the verb "saw". Head words can be different when only child word is changed. PP attachment disambiguation inherently requires accurate description of interactions among head, preposition and child, which becomes an ideal task to evaluate our tensor-based embeddings.

Task. The English dataset used in our work is collected from a linguistic treebank by Belinkov et al. (2014). Table 6 enumerates statistics associated with this dataset. Each instance consists of several head candidates, a preposition and a child word. We need to pick the head to which the preposition is attached. In the examples above, words "saw" and "elephant" are head candidates.

Our algorithm. Let $\mathbf{v}_h, \mathbf{v}_p$ and $\mathbf{v}_c$ be embeddings for the head candidate h, preposition p and child c respectively. Features we use for the attachment disambiguation are:

Table 6: Prepositional Attachment Disambiguation Dataset

	instances	avg head candidates	head set size	prep set size	child set size
Train	35,359	3.7	10,395	72	5,504
Test	1,951	3.6	2,133	46	983

Table 7: Accuracy in Prepositional Attachment Disambiguation

classifier	HPCD (enriching)	LRFR	OntoLSTM	FNN	FNN	FNN	FNN
embedding method	GloVe	word2vec	Glove-extended	word2vec	GloVe	Our method (ALS)	Our method (WD)
resources	pos tag, WordNet, VerbNet, syntactic parsing	pos tag, WordNet, VerbNet	pos tag, WordNet	pos tag	pos tag	pos tag	pos tag
accuracy	0.887	**0.903**	0.897	0.866	0.858	0.883	0.892

(1) embedding feature: candidate, preposition and child embedding;
(2) triple similarity: triple $\text{sim}(h, p, c) = \frac{\langle \mathbf{v}_h, \mathbf{v}_p, \mathbf{v}_c \rangle}{\|\mathbf{v}_h\|_3 \cdot \|\mathbf{v}_p\|_3 \cdot \|\mathbf{v}_c\|_3}$;
(3) head-preposition similarity: $\text{sim}(h, p) = \frac{\mathbf{v}_h^T \mathbf{v}_p}{\|\mathbf{v}_h\|_2 \cdot \|\mathbf{v}_p\|_2}$;
(4) head-child similarity: $\text{sim}(h, c) = \frac{\mathbf{v}_h^T \mathbf{v}_c}{\|\mathbf{v}_h\|_2 \cdot \|\mathbf{v}_c\|_2}$;
(5) part-of-speech (pos) tag of the candidate and its next word;
(b) distance between h and p.

We use a basic neural network, a two-layer feedforward network (FNN) with hidden sizes of 1000 and 20 to take input features and predict the probability that a candidate is the head. The candidate with the highest likelihood is chosen as the head.

Baseline. We include following state-of-the-art approaches in preposition attachment disambiguation. The linguistic resources they used to enrich features are listed in Table 7.
(1) Head-Prep-Child-Dist (HPCD) Model Belinkov et al. (2014): this compositional neural network is used to train task-specific word representations.
(2) Low-Rank Feature Representation (LRFR) Yu et al. (2016): this method incorporates word parts, contexts and labels into a tensor, and uses decomposed vectors as features for disambiguation.
(3) Ontology LSTM (OntoLSTM) Dasigi et al. (2017): Word vectors are initialized with GloVe-extended from AutoExtend Rothe & Schütze (2015), and then trained via LSTMs for head selection.

Similar to the experiments in preposition selection, we also include baselines which have the same feedforward network architecture but generate features with vectors trained by word2vec and GloVe. They are denoted as FNN with different initializations in Table 7. Since the attachment disambiguation is a selection task, accuracy is a natural evaluation metric.

Table 8: Ablation Analysis in Preposition Attachment Disambiguation

removed feature	head vector	prep vector	child vector	head-prep similarity	head-child similarity	triple similarity	POS	distance
accuracy	0.843	0.871	0.880	0.877	0.885	0.873	0.850	0.872

Result. We compare results and linguistic resources of different approaches in Table 7, where we see that our simple classifier built on the tensor representations is within 1% of the state of the art; prior state of the art results have used significant linguistic resources enumerated in Table 7. With the same feedforward neural network as the classifier, our tensor-based approaches (both ALS and WD) achieve better performance than word2vec and GloVe.

Ablation analysis in Table 8 shows that head vector feature affects the performance most (indicating that heads interact more closely with prepositions), and POS tag comes second. Similarity features appear less important since the classifier has access to lexical relatedness via the embedding features. Distance feature is reported to be important in previous works since 81.7% sentences take the word closest to the preposition as their head. In our experiments, distance becomes less important compared with embedding features.

Discussion. We find that one source of attachment disambiguation error is the lack of broader context in our features.Broader context is critical in examples such as "worked" and "system" which

are head candidates of "for trades" in a sentence. They are reasonable heads in expressions "worked for trades" and "system for trades". It requires more context to decide that "system" rather than "worked" is the head in the given sentence.

We further explore the difference in identifying head verbs and head nouns. We have found that tensor's geometry could aid in paraphrasing verb phrases, and thus it well captures the interaction between verbs and prepositions. In this task, we want to see whether our approach could do better in identifying head verbs than head nouns. There are 883 instances with head verbs on which we could achieve an accuracy of 0.897, and 1068 instances with head nouns where the accuracy is 0.887. We do better in selecting head verbs, but performance does not differ too much across verbs and nouns.

6 Related Work

Tensor Decomposition. Tensors embed higher order interaction among different modes, and the tensor decomposition captures the relations via lower dimensional representations. There are several decomposition methods such as Alternating Least Square (ALS) Kolda & Bader (2009), Simultaneous Diagonalization (SD) Kuleshov et al. (2015) and optimization-based methods Liu & Nocedal (1989); Moré (1978). Orthogonalized Alternating Least Square (Orth-ALS) adds the step of component orthogonalization to each update step in the ALS method Sharan & Valiant (2017). Orth-ALS, supported by theoretical guarantees and, more relevantly, good empirical performance, is the algorithm of choice in this paper.

Preposition Selection. Preposition selection, a major area of study in both syntactic and semantic computational linguistics, is also a very practical topic in the context of grammar correction and second language learning. Prior works typically use hand-crafted heuristic rules in preposition correction Xiang et al. (2013); lexical n-gram features are also known to be very useful Prokofyev et al. (2014); Rozovskaya et al. (2013). Syntactic information such as POS tags and dependency parsing can further enrich features Kao et al. (2013), and are standard in generic tasks involving prepositions.

Prepositional Attachment Disambiguation. There is a storied literature on prepositional attachment disambiguation, long recognized as an important part of syntactic parsing Kiperwasser & Goldberg (2016). Recent works, based on word embeddings have pushed the boundary of state of the art empirical results. A seminal work in this direction is the Head-Prep-Child-Dist (HPCD) Model, which trained word embeddings in a compositional network designed to maximize the accuracy of head prediction Belinkov et al. (2014). A very recent work has proposed an initialization with semantics-enriched GloVe embeddings, and retrained representations with LSTM-RNNs Dasigi et al. (2017). Another recent work has used tensor decompositions to capture the relation between word representations and their labels Yu et al. (2016).

7 Conclusion

Co-occurrence counts of word pairs in sentences and the resulting word vector representations (embeddings) have revolutionalized NLP research. A natural generalization is to consider co-occurrence counts of word triples, resulting in a third order tensor. Partly due to the size of the tensor (a vocabulary of 1M, leads to a tensor with 10^{18} entries!) and partly due to the extreme dynamic range of entries (including sparsity), word vector representations via tensor decompositions have largely been inferior to their lower order cousins (i.e., regular word embeddings).

In this work, we trek this well-trodden terrain but restricting word triples to the scenario when one of the words is a preposition. This is linguistically justified, since prepositions are understood to model interactions between pairs of words. Numerically, this is also very well justified since the sparsity and dynamic range of the resulting tensor is no worse than the original matrix of pairwise co-occurrence counts; this is because prepositions are very frequent and co-occur with essentially every word in the vocabulary.

Our intrinsic evaluations and new state of the art results in downstream evaluations lend strong credence to the tensor-based approach to prepositional representation. We expect our vector representations of prepositions to be widely used in more complicated downstream NLP tasks where prepositional role is crucial, including "text to programs" Guu et al. (2017).

REFERENCES

Rami Al-Rfou, Bryan Perozzi, and Steven Skiena. Polyglot: Distributed word representations for multilingual nlp. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pp. 183–192, Sofia, Bulgaria, August 2013. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/W13-3520.

Animashree Anandkumar, Rong Ge, and Majid Janzamin. Guaranteed non-orthogonal tensor decomposition via alternating rank-1 updates. *arXiv preprint arXiv:1402.5180*, 2014.

Yonatan Belinkov, Tao Lei, Regina Barzilay, and Amir Globerson. Exploring compositional architectures and word vector representations for prepositional phrase attachment. *Transactions of the Association for Computational Linguistics*, 2:561–572, 2014.

Steven Bird, Ewan Klein, and Edward Loper. *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media, Inc.", 2009.

Eric Brill and Philip Resnik. A rule-based approach to prepositional phrase attachment disambiguation. In *Proceedings of the 15th conference on Computational linguistics-Volume 2*, pp. 1198–1204. Association for Computational Linguistics, 1994.

Pierre Comon, Xavier Luciani, and André LF De Almeida. Tensor decompositions, alternating least squares and other tales. *Journal of chemometrics*, 23(7-8):393–405, 2009.

Pradeep Dasigi, Waleed Ammar, Chris Dyer, and Eduard Hovy. Ontology-aware token embeddings for prepositional phrase attachment. *arXiv preprint arXiv:1705.02925*, 2017.

Daniël de Kok and Erhard Hinrichs. Transition-based dependency parsing with topological fields. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, volume 2, pp. 1–7, 2016.

Jeanette S DeCarrico. *The structure of English: Studies in form and function for language teaching*, volume 1. University of Michigan Press/ESL, 2000.

Samuel J Gershman and Joshua B Tenenbaum. Phrase similarity in humans and machines. In *Proceedings of the 37th Annual Conference of the Cognitive Science Society*. Citeseer, 2015.

Kelvin Guu, Panupong Pasupat, Evan Zheran Liu, and Percy Liang. From language to programs: Bridging reinforcement learning and maximum marginal likelihood. *arXiv preprint arXiv:1704.07926*, 2017.

Kaveh Hassani and Won-Sook Lee. Disambiguating spatial prepositions using deep convolutional networks. In *AAAI*, pp. 3209–3215, 2017.

Rodney Huddleston. *Introduction to the Grammar of English*. Cambridge University Press, 1984.

Ting-Hui Kao, Yu-Wei Chang, Hsun-Wen Chiu, Tzu-Hsi Yen, Joanne Boisson, Jian-Cheng Wu, and Jason S Chang. Conll-2013 shared task: Grammatical error correction nthu system description. In *CoNLL Shared Task*, pp. 20–25, 2013.

Eliyahu Kiperwasser and Yoav Goldberg. Simple and accurate dependency parsing using bidirectional lstm feature representations. *arXiv preprint arXiv:1603.04351*, 2016.

Tamara G Kolda and Brett W Bader. Tensor decompositions and applications. *SIAM review*, 51(3): 455–500, 2009.

Volodymyr Kuleshov, Arun Chaganty, and Percy Liang. Tensor factorization via matrix factorization. In *Artificial Intelligence and Statistics*, pp. 507–516, 2015.

Jonathan K Kummerfeld, David Hall, James R Curran, and Dan Klein. Parser showdown at the wall street corral: An empirical investigation of error types in parser output. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pp. 1048–1059. Association for Computational Linguistics, 2012.

Claudia Leacock, Martin Chodorow, Michael Gamon, and Joel Tetreault. Automated grammatical error detection for language learners. *Synthesis lectures on human language technologies*, 3(1): 1–134, 2010.

Dong C Liu and Jorge Nocedal. On the limited memory bfgs method for large scale optimization. *Mathematical programming*, 45(1):503–528, 1989.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

Jeff Mitchell and Mirella Lapata. Composition in distributional models of semantics. *Cognitive science*, 34(8):1388–1429, 2010.

Jorge J Moré. The levenberg-marquardt algorithm: implementation and theory. In *Numerical analysis*, pp. 105–116. Springer, 1978.

Hwee Tou Ng, Siew Mei Wu, Ted Briscoe, Christian Hadiwinoto, Raymond Hendy Susanto, and Christopher Bryant. The conll-2014 shared task on grammatical error correction. In *CoNLL Shared Task*, pp. 1–14, 2014.

Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543, 2014.

Roman Prokofyev, Ruslan Mavlyutov, Martin Grund, Gianluca Demartini, and Philippe Cudré-Mauroux. Correct me if i'm wrong: Fixing grammatical errors by preposition ranking. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, pp. 331–340. ACM, 2014.

Sascha Rothe and Hinrich Schütze. Autoextend: Extending word embeddings to embeddings for synsets and lexemes. *arXiv preprint arXiv:1507.01127*, 2015.

Alla Rozovskaya, Kai-Wei Chang, Mark Sammons, and Dan Roth. The university of illinois system in the conll-2013 shared task. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning: Shared Task*, pp. 13–19, 2013.

Vatsal Sharan and Gregory Valiant. Orthogonalized als: A theoretically principled tensor decomposition algorithm for practical use. *arXiv preprint arXiv:1703.01804*, 2017.

Tom De Smedt. Available at: `https://www.nodebox.net/code/index.php/Linguistics`, 2016. Accessed:2017-10-01.

Shaden Smith and George Karypis. SPLATT: The Surprisingly ParalleL spArse Tensor Toolkit. `http://cs.umn.edu/~splatt/`, 2016.

Yang Xiang, Bo Yuan, Yaoyun Zhang, Xiaolong Wang, Wen Zheng, and Chongqiang Wei. A hybrid model for grammatical error correction. In *CoNLL Shared Task*, pp. 115–122, 2013.

Mo Yu, Mark Dredze, Raman Arora, and Matthew Gormley. Embedding lexical features via low-rank tensors. *arXiv preprint arXiv:1604.00461*, 2016.

A ROSTER OF PREPOSITIONS

The list of most frequent 49 Prepositions in the task of preposition selection is shown below:

about, above, absent, across, after, against, along, alongside, amid, among, amongst, around, at, before, behind, below, beneath, beside, besides, between, beyond, but, by, despite, during, except, for, from, in, inside, into, of, off, on, onto, opposite, outside, over, since, than, through, to, toward, towards, under, underneath, until, upon, with.

B PARAPHRASING OF PHRASAL VERBS

In Section 3 we have provided a simple linear algebraic method to generate paraphrases to compositional phrasal verbs. We approximate the paraphrase representation $\mathbf{u}_v$ via Eq. 3, and get a list of words which have similar representations as candidate paraphrases. These candidates do not include words that are the same as component words in the phrase. We also require that a reasonable paraphrase should be a verb. Therefore we choose the verb which is most similar to $\mathbf{u}_v$ among candidates. We filter verbs with Python NLTK tools Bird et al. (2009) and Linguistics library of NodeBoxSmedt (2016).

Sample examples of the top paraphrases are provided in Table 2. Here we provide a detailed enumeration of the results of our linear algebraic method on a new dataset of 60 compositional phrases. In the paraphrasing task, we consider three sets of embeddings, word2vec, GloVe and tensor embeddings from weighted decomposition. We also have two composition methods: addition and Hadamard product to approximate the paraphrase representation from verb and preposition vectors. Addition is included here because it has been widely used to approximate phrasal embedding in previous works Mitchell & Lapata (2010); Gershman & Tenenbaum (2015). We enumerate paraphrases generated by six combinations of embeddings and composition methods, validating the representation power of tensor embeddings and the multiplication (Hadamard product) composition method.

As we can see from Table 9 and 10, tensor embedding works better with multiplicative composition, whereas word2vec and GloVe work better with additive composition. Overall, tensor embedding together with multiplication gives better paraphrases than other approaches.

Table 9: Paraphrasing of Phrasal Verbs

	word2vec		GloVe		Tensor (WD)	
	addition	multiplication	addition	multiplication	addition	multiplication
sparked off	ignited	strip	came	beginning	led	prompted
involved in	interested	spoof	having	came	featured	included
accuse of	criticize	consists	involved	scrapped	posted	convicted
approved of	mandated	entering	given	followed	recommended	issued
stuck with	dragging	romanize	having	came	posted	stalled
derived from	originates	modeled	based	came	reading	generated
resign from	withdraw	assaulted	leaving	came	march	retire
attached to	desired	erect	take	followed	assigned	subordinate
passed down	lifted	captained	put	beginning	voted	delivered
adjust to	modify	decorate	help	took	lose	adapt
regarded as	considered	depicts	known	last	considered	perceived
differ from	vary	diet	derived	took	vary	vary
treated as	tolerated	co-founded	known	starting	employed	regarded

Table 10: Paraphrasing of Phrasal Verbs

	word2vec		GloVe		Tensor (WD)	
	addition	multiplication	addition	multiplication	addition	multiplication
conform to	modify	coloured	make	set	comply	comply
correspond to	specify	nominate	give	last	represent	correlate
replied to	said	intercept	give	lost	posted	answered
blend in	mix	sort	place	starting	feature	mix
switched over	dropped	witness	having	beginning	kent	transferred
carried on	laid	shortlisted	taken	came	employed	kept
switched on	shifted	briefing	moved	came	kent	tapped
carried in	laid	alternated	came	came	featured	created
handed in	taken	awarded	taken	came	aged	passed
kicked in	knocked	fencing	went	last	scored	intercepted
locked in	sealed	vending	having	came	last	placed
put in	brought	highlights	came	came	took	place
smashed in	shattered	offers	came	came	dated	crashing
handed over	stripped	disadvantaged	took	came	posted	tried
blocked off	stopped	attire	cut	came	posted	intercepted
scraped off	trap	stripe	turn	inspire	aged	ripped
shook off	smashed	lease	struck	followed	aged	shattered
split off	crashing	posing	turn	came	starting	broken
tailed off	trap	traps	put	manufactures	aged	masked
wiped off	crushed	kent	leaving	followed	remaining	knocked
slapped down	forward	inspire	put	stripe	totaled	stabbed
fit in	adjust	singled	set	came	feature	conform
brought in	came	conditioned	came	came	following	received
invest in	acquire	trailing	did	came	spend	market
convinced of	assured	resurfaced	take	came	march	revealed
dreamed of	imagined	encompassed	making	starting	aged	wants
despair of	utter	leach	taken	followed	aged	recall
pray for	bless	slide	seeking	came	dated	hope
prepared for	required	truncated	used	came	delivered	sponsored
dealt with	coupled	tide	having	came	covering	experimented
emerged from	persisted	stealing	brought	came	saw	occurred
made from	followed	co-authored	came	came	following	produced
distract from	avoid	distinguishing	taken	starting	mark	weaken
recover from	reclaim	exited	return	came	hunt	retrieve
knocked over	smashed	tailored	lost	came	injured	blown
fell down	collapsed	fool	put	came	lost	surrendered
leave behind	hide	transit	rest	followed	return	fall
come into	disappear	styled	take	followed	get	go
asked for	requested	curate	take	came	wrote	requested
fight for	defeat	incorporated	return	came	challenge	battle
fight against	defeat	reprinted	defeat	last	last	battle
acted as	functioned	argued	known	came	appeared	disguised
stripped off	smashed	fictionalized	taken	came	giving	lowered
melt down	explode	commemorate	turn	put	wade	overcome
looked at	seemed	archived	came	having	got	corresponded
lived in	resided	attained	came	came	died	located
asking about	told	opened	stated	starting	win	requests