# Discriminative Phrase Embedding for Paraphrase Identification

**Wenpeng Yin** and **Hinrich Schütze**
Center for Information and Language Processing
University of Munich, Germany
wenpeng@cis.lmu.de

## Abstract

This work, concerning paraphrase identification task, on one hand contributes to expanding deep learning embeddings to include continuous and discontinuous linguistic phrases. On the other hand, it comes up with a new scheme TF-KLD-KNN to learn the discriminative weights of words and phrases specific to paraphrase task, so that a weighted sum of embeddings can represent sentences more effectively. Based on these two innovations we get competitive state-of-the-art performance on paraphrase identification.

## 1 Introduction

This work investigates representation learning via deep learning in paraphrase identification task, which aims to determine whether two sentences have the same meaning. One main innovation of deep learning is that it learns distributed word representations (also called "word embeddings") to deal with various Natural Language Processing (NLP) tasks. Our goal is to use and refine embeddings to get competitive performance.

We adopt a supervised classification approach to paraphrase identification like most top performing systems. Our focus is representation learning of sentences. Following prior work (e.g., Blacoe and Lapata (2012)), we compute the vector of a sentence as the sum of the vectors of its components. But unlike prior work we use *single words, continuous phrases and discontinuous phrases* as the components, not just single words. Our rationale is that many semantic units are formed by multiple words – e.g., the continuous phrase "side effects" and the discontinuous phrase "pick ... off". The better we can discover and represent such components, the better the compositional sentence vector should be. We use the term *unit* to refer to single words, continuous phrases and discontinuous phrases.

Ji and Eisenstein (2013) show that not all words are equally important for paraphrase identification. They propose TF-KLD, a discriminative weighting scheme to address this problem. While they do not represent sentences as vectors composed of other vectors, TF-KLD is promising for a vector-based approach as well since the insight that units are of different importance still applies. A shortcoming of TF-KLD is its failure to define weights for words that do not occur in the training set. We propose TF-KLD-KNN, an extension of TF-KLD that computes the weight of an unknown unit as the average of the weights of its $k$ nearest neighbors. We determine nearest neighbors by cosine measure over embedding space. We then represent a sentence as the sum of the vectors of its units, weighted by TF-KLD-KNN.

We use (Madnani et al., 2012) as our baseline system. They used simple features – eight different machine translation metrics – yet got good performance. Based on above new sentence representations, we compute three kinds of features to describe a pair of sentences – cosine similarity, element-wise sum and absolute element-wise difference – and show that combining them with the features from Madnani et al. (2012) gets state-of-the-art performance on the Microsoft Research Paraphrase (MSRP) corpus (Dolan et al., 2004).



*Human Language Technologies: The 2015 Annual Conference of the North American Chapter of the ACL*, pages 1368–1373,
Denver, Colorado, May 31 – June 5, 2015. 

In summary, our first contribution lies in embedding learning of continuous and discontinuous phrases. Our second contribution is the weighting scheme TF-KLD-KNN.

This paper is structured as follows. Section 2 reviews related work. Section 3 describes our method for learning embeddings of units. Section 4 introduces a measure of unit discriminativity that can be used for differential weighting of units. Section 5 presents experimental setup and results. Section 6 concludes.

## 2 Related work

The key for good performance in paraphrase identification is the design of good features. We now discuss relevant prior work based on the linguistic granularity of feature learning.

The first line is compositional semantics, which learns representations for words and then composes them to representations of sentences. Blacoe and Lapata (2012) carried out a comparative study of three word representation methods (the simple distributional semantic space (Mitchell and Lapata, 2010), distributional memory tensor (Baroni and Lenci, 2010) and word embedding (Collobert and Weston, 2008)), along with three composition methods (addition, point-wise multiplication, and recursive autoencoder (Socher et al., 2011)). They showed that addition over word embeddings is competitive, despite its simplicity.

The second category directly seeks sentence-level features. Ji and Eisenstein (2013) explored unigrams, bigrams and dependency pairs as sentence features. They proposed TF-KLD to weight features and used non-negative factorization to learn latent sentence representations. Our method TF-KLD-KNN is an extension of their work.

The third line directly computes features for sentence pairs. Wan et al. (2006) used N-gram overlap, dependency relation overlap, dependency tree-edit distance and difference of sentence lengths. Finch et al. (2005) and Madnani et al. (2012) combined several machine translation metrics. Das and Smith (2009) presented a generative model over two sentences' dependency trees, incorporating syntax, lexical semantics, and hidden loose alignments between the trees to model generating a paraphrase of a given sentence. Socher et al. (2011) used recursive autoencoders to learn representations for words and word sequences on each layer of the sentence parsing tree, and then proposed dynamic pooling layer to form a fixed-size matrix as the representation of the two sentences. Other work representative of this line is by Kozareva and Montoyo (2006), Qiu et al. (2006), Ul-Qayyum and Altaf (2012).

Our work, first learning unit embeddings, then adding them to form sentence representations, finally calculating pair features (cosine similarity, absolute difference and MT metrics) actually is a combination of above three lines.

## 3 Embedding learning for units

As explained in Section 1, "units" in this work include single words, continuous phrases and discontinuous phrases. Phrases have a larger linguistic granularity than words and thus will in general contain more meaning aspects for a sentence. For example, successful detection of continuous phrase "side effects" and discontinuous phrase "pick $\cdots$ off" is helpful to understand the sentence meaning correctly. This section focuses on how to detect phrases and how to represent them.

### 3.1 Phrase collection

Phrases defined by a lexicon have not been investigated extensively before in deep learning. To collect canonical phrase set, we extract two-word phrases defined in Wiktionary[1] and Wordnet (Miller and Fellbaum, 1998) to form a collection of size 95,218. This collection contains *continuous phrases* – phrases whose parts always occur next to each other (e.g., "side effects") – and *discontinuous phrases* – phrases whose parts more often occur separated from each other (e.g., "pick ... off").

### 3.2 Identification of phrase continuity

Wiktionary and WordNet do not categorize phrases as continuous or discontinuous. So we need a heuristic to determine this automatically.

For each phrase "A_B", we compute [$c_1$, $c_2$, $c_3$, $c_4$, $c_5$] where $c_i, 1 \leq i \leq 5$, indicates there are $c_i$ occurrences of A and B in that order with a distance

[1] http://en.wiktionary.org

of $i$. We compute these statistics for a corpus consisting of English Gigaword (Graff et al., 2003) and Wikipedia. We set the maximal distance to 5 because discontinuous phrases are rarely separated by more than 5 tokens.

If $c_1$ is 10 times higher than $(c_2+c_3+c_4+c_5)/4$, we classify "A_B" as *continuous*, otherwise as *discontinuous*. For example, $[c_1, \ldots, c_5]$ is [1121, 632, 337, 348, 4052] for "pick_off", so $c_1$ is smaller than the average 1342.25 and "pick_off" is set as "discontinuous"; $[c_1, \ldots, c_5]$ is [14831, 16, 177, 331, 3471] for "Cornell University", $c_1$ is 10 times larger than the average and this phrase is set to "continuous".

We found that that this heuristic for distinguishing between continuous and discontinuous phrases works well and leave the development of a more principled method for future work.

### 3.3 Sentence reformatting

Sentence "... A ... B ..." is

- reformatted as "... A_B ..." if A and B form a continuous phrase and no word intervenes between them and
- reformatted as "... A_B ... A_B ..." if A and B form a discontinuous phrase and are separated by 1 to 4 words. We replace each of the two component words with A_B to make the context of both constituents available to the phrase in learning.

This method of phrase detection will generate some false positives, e.g., if "pick" and "off" occur in a context like "she picked an island off the coast of Maine". However, our experimental results indicate that it is robust enough for our purposes.

We run word2vec (Mikolov et al., 2013) on the reformatted Wikipedia corpus to learn embeddings for all units. Embedding size is set to 200.

## 4 Measure of unit discriminativity

We will represent a sentence as the sum of the embeddings of its units. Building on Ji and Eisenstein (2013)'s TF-KLD, we want to weight units according to their ability to discriminate two sentences specific to the paraphrase task.

TF-KLD assumes a training set of sentence pairs in the form $\langle u_i, v_i, t_i \rangle$, where $u_i$ and $v_i$ denote the binary unit occurrence vectors for the sentences in the $i$th pair and $t_i \in \{0, 1\}$ is the gold tag. Then, we define $p_k$ and $q_k$ as follows.

- $p_k = P(u_{ik}|v_{ik} = 1, t_i = 1)$. This is the probability that unit $w_k$ occurs in sentence $u_i$ given that $w_k$ occurs in its counterpart $v_i$ and they are paraphrases.
- $q_k = P(u_{ik}|v_{ik} = 1, t_i = 0)$. This is the probability that unit $w_k$ occurs in sentence $u_i$ given that $w_k$ occurs in its counterpart $v_i$ and they are not paraphrases.

TF-KLD computes the discriminativity of unit $w_k$ as the Kullback-Leibler divergence of the Bernoulli distributions ($p_k$, 1-$p_k$) and ($q_k$, 1-$q_k$)

TF-KLD has a serious shortcoming for unknown units. Unfortunately, the test data of the commonly used MSPR corpus in paraphrase task has about 6% unknown words and 62.5% of its sentences contain unknown words. It motivates us to design an improved scheme TF-KLD-KNN to reweight the features.

TF-KLD-KNN weights are the same as TF-KLD weights for known units. For a unit that did not occur in training, TF-KLD-KNN computes its weight as the average of the weights of its $k$ nearest neighbors in embedding space, where unit similarity is calculated by cosine measure.[2]

Word2vec learns word embeddings based on the word context. The intuition of TF-KLD-KNN is that words with similar context have similar discriminativities. This enables us to transfer the weights of features in training data to the unknown features in test data, greatly helping to address problems of sparseness.

## 5 Experiments

### 5.1 Data and baselines

We use the MSRP corpus (Dolan et al., 2004) for evaluation. It consists of a training set of 2753 true paraphrase pairs and 1323 false paraphrase pairs and a test set of 1147 true and 578 false pairs.

[2]Unknown words without embeddings (only seven cases in our experiments) are ignored. This problem can be effectively relieved by training embedding on larger corpora.

For our new method, it is interesting to measure the improvement on the subset of those MSRP sentences that contain at least one phrase. In the standard MSRP corpus, 3027 training pairs (2123 true, 904 false) and 1273 test pairs (871 true, 402 false) contain phrases; we denote this subset as *subset*. We carry out experiments on *overall* (all MSRP sentences) as well as *subset* cases.

We compare six methods for paraphrase identification.

- **NOWEIGHT.** Following Blacoe and Lapata (2012), we simply represent a sentence as the unweighted sum of the embeddings of all its units.

- **MT** is the method proposed by Madnani et al. (2012): the sentence pair is represented as a vector of eight different machine translation metrics.

- **Ji and Eisenstein (2013)**. We reimplemented their "inductive" setup which is based on matrix factorization and is the top-performing system in paraphrasing task.[3]

  The following three methods not only use this vector of eight MT metrics, but use three kinds of additional features given two sentence representations $s_1$ and $s_2$: cosine similarity, element-wise sum $s_1 + s_2$ and element-wise absolute difference $|s_1 - s_2|$. We now describe how each of the three methods computes the sentence vectors.

- **WORD.** The sentence is represented as the sum of all single-word embeddings, weighted by TF-KLD-KNN.

- **WORD+PHRASE.** The sentence is represented as the sum of the embeddings of all its units (including phrases), weighted by TF-KLD-KNN.

- **WORD+GOOGLE.** Mikolov et al. (2013) use a data-driven method to detect statistical phrases which are mostly continuous bigrams. We implement their system by first exploiting word2phrase[4] to reformat Wikipedia, then using word2vec skip-gram model to train phrase embeddings.

We use the same weighting scheme TF-KLD-KNN for the three weighted sum approaches: WORD, WORD+PHRASE and WORD+GOOGLE. Note however that there is an interaction between representation space and nearest neighbor search. We limit the neighbor range of unknown words for WORD to single words; in contrast, we search the space of all single words and linguistic (resp. Google) phrases for WORD+PHRASE (resp. WORD+GOOGLE).

We use LIBLINEAR (Fan et al., 2008) as our linear SVM implementation. 20% training data is used as development data. Parameter $k$ is fine-tuned on development set and the best value 3 is finally used in following reported results.

### 5.2 Experimental results

Table 1 shows performance for the six methods as well as for the majority baseline. In the *overall* (resp. *subset*) setup, WORD+PHRASE performs best and outperforms (Ji and Eisenstein, 2013) by .009 (resp. .052) on accuracy. Interestingly, Ji and Eisenstein (2013)'s method obtains worse performance on $subset$. This can be explained by the effect of matrix factorization in their work: it works less well for smaller datasets like $subset$. This is a shortcoming of their approach. WORD+GOOGLE has a slightly worse performance than WORD+PHRASE; this suggests that linguistic phrases might be more effective than statistical phrases in identifying paraphrases.

Cases *overall* and *subset* both suggest that phrase embeddings improve sentence representations. The accuracy of WORD+PHRASE is lower on *overall* than on *subset* because WORD+PHRASE has no advantage over WORD for sentences without phrases.

### 5.3 Effectiveness of TF-KLD-KNN

The key contribution of TF-KLD-KNN is that it achieves full coverage of feature weights in the face of data sparseness. We now compare four weighting methods on overall corpus and with the combi-

[3] They report even better performance in a "transductive" setup that makes use of test data. We only address paraphrase identification for the case that the test data are not available for training the model in this paper.

[4] `https://code.google.com/p/word2vec/`

| | overall | | subset | |
|---|---|---|---|---|
| method | acc | $F_1$ | acc | $F_1$ |
| baseline | .665 | .799 | .684 | .812 |
| NOWEIGHT | .708 | .809 | .713 | .823 |
| MT | .774 | .841 | .772 | .839 |
| Ji and Eisenstein (2013) | .778 | .843 | .749 | .827 |
| WORD | .775 | .839 | .776 | .843 |
| WORD+GOOGLE | .780 | .843 | .795 | .853 |
| WORD+PHRASE | **.787** | **.848**$^*$ | **.801** | **.857**$^*$ |

Table 1: Results on overall and subset corpus. Significant improvements over MT are marked with $*$ (approximate randomization test, Padó (2006), $p < .05$).

| method | acc | $F_1$ |
|---|---|---|
| NOWEIGHT | .746 | .815 |
| TF-IDF | .752 | .821 |
| TF-KLD | .774 | .842 |
| TF-KLD-KNN | **.787** | **.848** |

Table 2: Effects of different reweighting methods on overall.

nation of MT features: NOWEIGHT, TF-IDF, TF-KLD, TF-KLD

Table 2 suggests that task-specific reweighting approaches (including TF-KLD and TF-KLD-KNN) are superior to unspecific schemes (NOWEIGHT and TF-IDF). Also, it demonstrates the effectiveness of our weight learning solution for unknown units in paraphrase task.

### 5.4 Reweighting schemes for unseen units

We compare our reweighting scheme **KNN** (i.e., TF-KLD-KNN) with three other reweighting schemes. **Zero**: zero weight, i.e., ignore unseen units; **Type-average**: take the average of weights of all known unit types in test set; **Context-average**: average of the weights of the adjacent known units of the unknown unit (two, one or defaulting to Zero, depending on how many there are). Figure 1 shows that KNN performs best.

## 6 Conclusion

This work introduced TF-KLD-KNN, a new reweighting scheme that learns the discriminativities of known as well as unknown units effectively. We further improved paraphrase identification performance by the utilization of continuous and discontinuous phrase embeddings.

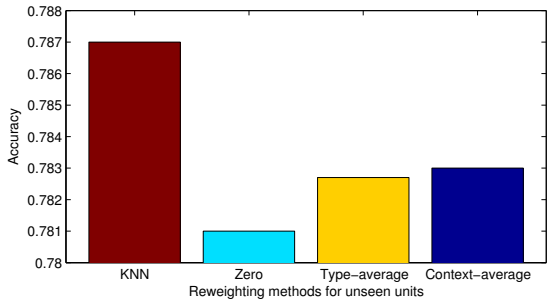


Figure 1: Performance of different reweighting schemes for unseen units on overall.

In future, we plan to do experiments in a cross-domain setup and enhance our algorithm for domain adaptation paraphrase identification.

## Acknowledgments

We are grateful to members of CIS for comments on earlier versions of this paper. This work was supported by Baidu (through a Baidu scholarship awarded to Wenpeng Yin) and by Deutsche Forschungsgemeinschaft (grant DFG SCHU 2246/8-2, SPP 1335).

## References

Marco Baroni and Alessandro Lenci. 2010. Distributional memory: A general framework for corpus-based semantics. *Computational Linguistics*, 36(4):673–721.

William Blacoe and Mirella Lapata. 2012. A comparison of vector-based representations for semantic composition. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 546–556. Association for Computational Linguistics.

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

Dipanjan Das and Noah A Smith. 2009. Paraphrase identification as probabilistic quasi-synchronous recognition. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Pro-*

*cessing of the AFNLP: Volume 1-Volume 1*, pages 468–476. Association for Computational Linguistics.

Bill Dolan, Chris Quirk, and Chris Brockett. 2004. Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. In *Proceedings of the 20th international conference on Computational Linguistics*, pages 350–356. Association for Computational Linguistics.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. Liblinear: A library for large linear classification. *The Journal of Machine Learning Research*, 9:1871–1874.

Andrew Finch, Young-Sook Hwang, and Eiichiro Sumita. 2005. Using machine translation evaluation techniques to determine sentence-level semantic equivalence. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*, pages 17–24.

David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. 2003. English gigaword. *Linguistic Data Consortium, Philadelphia*.

Yangfeng Ji and Jacob Eisenstein. 2013. Discriminative improvements to distributional sentence similarity. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Zornitsa Kozareva and Andrés Montoyo. 2006. Paraphrase identification on the basis of supervised machine learning techniques. In *Advances in natural language processing*, pages 524–533. Springer.

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

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

George Miller and Christiane Fellbaum. 1998. Wordnet: An electronic lexical database.

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

Sebastian Padó, 2006. *User's guide to `sigf`: Significance testing by approximate randomisation*.

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

Richard Socher, Eric H Huang, Jeffrey Pennington, Andrew Y Ng, and Christopher D Manning. 2011. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In *Advances in Neural Information Processing Systems*, volume 24, pages 801–809.

Zia Ul-Qayyum and Wasif Altaf. 2012. Paraphrase identification using semantic heuristic features. *Research Journal of Applied Sciences, Engineering and Technology*, 4(22):4894–4904.

Stephen Wan, Mark Dras, Robert Dale, and Cécile Paris. 2006. Using dependency-based features to take the para-farce out of paraphrase. In *Proceedings of the Australasian Language Technology Workshop*, volume 2006, pages 131–138.