Convolutional Sentence Kernel from Word Embeddings for Short Text Categorization

Jonghoon Kim **François Rousseau** **Michalis Vazirgiannis**

LIX, École Polytechnique, France
john.jonghoon.kim@gmail.com

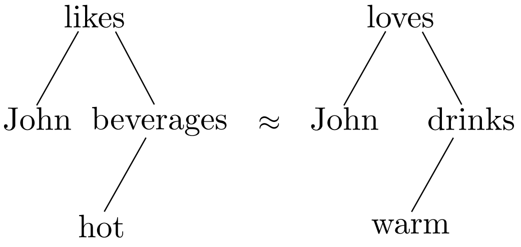
Abstract

This paper introduces a convolutional sentence kernel based on word embeddings. Our kernel overcomes the sparsity issue that arises when classifying short documents or in case of little training data. Experiments on six sentence datasets showed statistically significant higher accuracy over the standard linear kernel with n-gram features and other proposed models.

1 Introduction

With the proliferation of text data available online, text categorization emerged as a prominent research topic. Traditionally, words (unigrams) and phrases (n-grams) have been considered as document features and subsequently fed to a classifier such as an SVM (Joachims, 1998). In the SVM dual formulation that relies on kernels, i. e. similarity measures between documents, a linear kernel can be interpreted as the number of exact matching n-grams between two documents. Consequently, for short documents or when little training data is available, sparsity issues due to word synonymy arise, e. g., the sentences 'John likes hot beverages' and 'John loves warm drinks' have little overlap and therefore low linear kernel value (only 1) in the n-gram feature space, even with dependency tree representations and downward paths for n-grams as illustrated in Figure 1.

We propose to relax the exact matching between words by capitalizing on distances in word embeddings. We smooth the implicit delta word kernel, i. e. a Dirac similarity function between unigrams, behind the traditional linear document kernel to capture the similarity between words that are different, yet semantically close. We then aggregate these word and phrase kernels into sentence and documents kernels through convolution resulting in higher kernel values between semantically related sentences (e. g., close to 7 compared to 1 with bigram downward paths in Figure 1). Experiments on six standard datasets for sentiment analysis, subjectivity detection and topic spotting showed statistically significant higher accuracy for our proposed kernel over the bigram approaches. Our main goal is to demonstrate empirically that word distances from a given word vector space can easily be incorporated in the standard kernel between documents for higher effectiveness and little additional cost in efficiency.



(a) 'John likes hot beverages' (b) 'John loves warm drinks'

Figure 1: Dependency tree representations of semantically related sentences yet with little overlap.

The rest of this paper is structured as follows. Section 2 reviews the related work. Section 3 gives the detailed formulation of our kernel. Section 4 describes the experimental settings and the results we obtained on several datasets. Finally, Section 5 concludes our paper and mentions future work.

2 Related work

Siolas and d'Alché Buc (2000) pioneered the idea of *semantic kernels* for text categorization, capitalizing on WordNet (Miller, 1995) to propose continuous word kernels based on the inverse of the path lengths in the tree rather than the common delta word kernel used so far, i. e. exact matching between unigrams. Bloehdorn et al. (2006) extended it later to other tree-based similarity measures from WordNet while Mavroeidis et al. (2005) exploited its hierarchical structure to define a Generalized Vector Space Model kernel.

Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pages 775–780,
Lisbon, Portugal, 17-21 September 2015. ©2015 Association for Computational Linguistics.

In parallel, Collins and Duffy (2001) developed the first *tree kernels* to compare trees based on their topology (e. g., shared subtrees) rather than the similarity between their nodes. Culotta and Sorensen (2004) used them as Dependency Tree Kernels (DTK) to capture *syntactic similarities* while Bloehdorn and Moschitti (2007) and Croce et al. (2011) used them on parse trees with respectively Semantic Syntactic Tree Kernels (SSTK) and Smoothing Partial Tree Kernels (SPTK), adding node similarity based on WordNet to capture *semantic similarities* but limiting to comparisons between words of the same POS tag.

Similarly, Gärtner et al. (2003) developed *graph kernels* based on random walks and Srivastava et al. (2013) used them on dependency trees with Vector Tree Kernels (VTK), adding node similarity based on word embeddings from SENNA (Collobert et al., 2011) and reporting improvements over SSTK. The change from WordNet to SENNA was supported by the recent progress in low-dimension Euclidean vector space representations of words that are better suited for computing distances between words. Actually, in our experiments, word2vec by Mikolov et al. (2013a) led to better results than with SENNA for both VTK and our kernels. Moreover, it possesses an additional additive compositionality property obtained from the Skip-gram training setting (Mikolov et al., 2013b), e. g., the closest word to 'Germany' + 'capital' in the vector space is found to be 'Berlin'.

More recently, for short text similarity, Song and Roth (2015) and Kenter and de Rijke (2015) proposed additional semantic meta-features based on word embeddings to enhance classification.

3 Formulation

We denote the embedding of a word w by $\mathbf{w}$.

3.1 Word Kernel (WK)

We define a kernel between two words as a polynomial kernel over a cosine similarity in the word embedding space:

$$\mathrm{WK}(w_1, w_2) = \left[\frac{1}{2}\left(1 + \frac{\langle \mathbf{w}_1, \mathbf{w}_2 \rangle}{\|\mathbf{w}_1\| \|\mathbf{w}_2\|}\right)\right]^{\alpha} \quad (1)$$

where α is a scaling factor. We also tried Gaussian, Laplacian and sigmoid kernels but they led to poorer results in our experiments. Note that a delta word kernel, i. e. the Dirac function $\mathbb{1}_{w_1=w_2}$, leads to a document kernel corresponding to the standard linear kernel over n-grams.

3.2 Phrase Kernel (PhK)

Next we define a kernel between phrases consisting of several words. In our work, we considered two types of phrases: (1) *co-occurrence phrases* defined as contiguous sequences of words in the text; and (2) *syntactic phrases* defined as downward paths in the dependency tree representation, e. g., respectively 'hot beverages' and 'beverages hot' in Figure 1. With this dependency tree involved, we expect to have phrases that are syntactically more meaningful. Note that VTK considers *random walks* in dependency trees instead of downward paths, i. e. potentially taking into account same nodes multiple times for phrase length greater than two, phenomenon known as tottering.

Once we have phrases to compare, we may construct a kernel between them as the product of word kernels if they are of the same length l. That is, we define the Product Kernel (PK) as:

$$\mathrm{PK}(p_1, p_2) = \prod_{i=1}^{l} \mathrm{WK}(w_i^1, w_i^2) \quad (2)$$

where w_i^j is the i-th word in phrase p_j of length l. Alternatively, in particular for phrases of different lengths, we may embed phrases into the embedding space by taking a composition operation on the constituent word embeddings. We considered two common forms of composition (Blacoe and Lapata, 2012): vector addition ($+$) and element-wise multiplication ($\odot$). Then we define the Composition Kernel (CK) between phrases as:

$$\mathrm{CK}(p_1, p_2) = \mathrm{WK}(\mathbf{p}_1, \mathbf{p}_2) \quad (3)$$

where $\mathbf{p}_j$, the embedding of the phrase p_j, can be obtained either by addition ($\mathbf{p}_j = \sum_{i=1}^{l} \mathbf{w}_i^j$) or by element-wise multiplication ($\mathbf{p}_j = \bigodot_{i=1}^{l} \mathbf{w}_i^j$) of its word embeddings. For CK, we do not require the two phrases to be of the same length so the kernel has a desirable property of being able to compare 'Berlin' with 'capital of Germany' for instance.

3.3 Sentence Kernel (SK)

We can then formulate a sentence kernel in a similar way to Zelenko et al. (2003). It is defined through convolution as the sum of all local phrasal similarities, i. e. kernel values between phrases contained in the sentences:

$$\mathrm{SK}(s_1, s_2) = \sum_{\substack{p_1 \in \phi(s_1), \\ p_2 \in \phi(s_2)}} {\lambda_1}^{\epsilon} \, {\lambda_2}^{\eta} \, \mathrm{PhK}(p_1, p_2) \quad (4)$$

where $\phi(s_k)$ is the set of either statistical or syntactic phrases (or set of random walks for VTK) in sentence s_k, λ_1 is a decaying factor penalizing longer phrases, $\epsilon = \max\{|p_1|, |p_2|\}$ is the maximum length of the two phrases, λ_2 is a distortion parameter controlling the length difference η between the two phrases ($\eta = ||p_1| - |p_2||$) and PhK is a phrase kernel, either PK, CK$^+$ or CK$^\odot$.

Since the composition methods we consider are associative, we employed a dynamic programming approach in a similar fashion to Zelenko et al. (2003) to avoid duplicate computations.

3.4 Document Kernel

Finally, we sum sentence kernel values for all pairs of sentences between two documents to get the document kernel. Once we have obtained all document kernel values K_{ij} between documents i and j, we may normalize them by $\sqrt{K_{ii}K_{jj}}$ as the length of input documents might not be uniform.

4 Experiments

We evaluated our kernel with co-occurrence and syntactic phrases on several standard text categorization tasks.

4.1 Datasets

We considered four tasks: (1) binary *sentiment analysis* with a movie review dataset of 10,662 sentences (PL05) (Pang and Lee, 2005) and a product review dataset (Amazon) of 2,000 multi-line documents for 4 different product groups (Blitzer et al., 2007) (we will report the average effectiveness over the 4 sub-collections); (2) ternary *sentiment analysis* with the SemEval 2013 Task B dataset (Twitter) containing 12,348 tweets classified as positive, neutral or negative (Nakov et al., 2013); (3) binary *subjectivity detection* with a dataset of 10,000 sentences (PL04) (Pang and Lee, 2004) and another of 11,640 sentences (MPQA) (Wiebe et al., 2005); and (4) seven-class *topic spotting* with a news dataset (News) of 32,602 one-line news summaries (Vitale et al., 2012).

4.2 Experimental settings

In all our experiments, we used the FANSE parser (Tratz and Hovy, 2011) to generate dependency trees and the pre-trained version of word2vec[1], a 300 dimensional representation of 3 million English words trained over a Google News dataset of 100 billion words using the Skip-gram model and a context size of 5. While fine-tuning the embeddings to a specific task or on a given dataset may improve the result for that particular task or dataset (Levy et al., 2015), it makes the expected results less generalizable and the method harder to use as an off-the-shelf solution – re-training the neural network to obtain task-specific embeddings requires a certain amount of training data, admittedly unlabeled, but still not optimal under our scenario with short documents and little task-specific training data available. Moreover, tuning the hyperparameters to maximize the classification accuracy needs to be carried out on a validation set and therefore requires additional labeled data. Here, we are more interested in showing that distances in a given word vector space can enhance classification in general. As for the dependency-based word embeddings proposed by Levy and Goldberg (2014), we do not think they are better suited for the problem we are tackling. As we will see in the results, we do benefit from the dependency tree structure in the phrase kernel but we still want the word kernel to be based on topical similarity rather than functional similarity.

To train and test the SVM classifier, we used the LibSVM library (Chang and Lin, 2011) and employed the one-vs-one strategy for multi-class tasks. To prevent overfitting, we tuned the parameters using cross-validation on 80% of PL05 dataset ($\alpha = 5$, $\lambda_1 = 1$ for PK since there is no need for distortion as the phrases are of the same length by definition, and $\lambda_1 = \lambda_2 = 0.5$ for CK) and used the same set of parameters on the remaining datasets. We performed normalization for our kernel and baselines only when it led to performance improvements on the training set (PL05, News, PL04 and MPQA).

We report accuracy on the remaining 20% for PL05, on the standard test split for Twitter (25%) and News (50%) and from 5-fold cross-validation for the other datasets (Amazon, PL04 and MPQA). We only report accuracy as the macro-average F1-scores led to similar conclusions (and except for Twitter and News, the class label distributions are balanced). Results for phrase lengths longer than two were omitted since they were marginally different at best. Statistical significance of improvement over the bigram baseline with the same phrase definition was assessed using the micro sign test ($p < 0.01$) (Yang and Liu, 1999).

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

Table 1: Accuracy results on the test set for PL05 (20%), standard test split for Twitter (25%) and News (50%) and from 5-fold CV for the other datasets (Amazon, PL04 and MPQA). Bold font marks the best performance in the column. * indicates statistical significance at $p < 0.01$ using micro sign test against the bigram baseline (delta word kernel) of the same column and with the same phrase definition.

phrase definition	phrase kernel	phrase length	word kernel	PL05	Amazon	Twitter	News	PL04	MPQA
co-occurrence	PK	1	delta	0.742	0.768	0.623	0.769	0.904	0.754
co-occurrence	PK	2	delta	0.739	0.765	0.611	0.766	0.907	0.754
syntactic	PK	2	delta	0.748	0.791	0.646	0.767	0.910	0.757
random walk	PK	2	poly	0.799	0.810	0.698	0.802	**0.927**	**0.797**
co-occurrence	PK	1	poly	0.789*	0.797	0.776*	**0.806***	0.923*	0.793*
co-occurrence	PK	2	poly	0.784*	0.798	0.762*	0.801*	0.926*	0.794*
co-occurrence	CK$^+$	2	poly	0.796*	0.778	0.613	0.792*	0.917*	0.796*
co-occurrence	CK$^\odot$	2	poly	**0.801***	0.783	0.757*	0.793*	0.918*	0.794*
syntactic	PK	2	poly	0.796*	**0.813***	**0.808***	0.805*	**0.927***	0.796*
syntactic	CK$^+$	2	poly	0.794*	0.780	0.741*	0.788*	0.918*	0.794*
syntactic	CK$^\odot$	2	poly	0.797*	0.774	0.744*	0.792*	0.918*	0.794*

4.3 Results

Table 1 presents results from our convolutional sentence kernel and the baseline approaches. Note again that a delta word kernel leads to the typical unigram and bigram baseline approaches (first three rows). The 3rd row corresponds to DTK (Culotta and Sorensen, 2004) and the 4th one to VTK (Srivastava et al., 2013) – the difference with our model on the 9th row lies in the function $\phi(\cdot)$ that enumerates all random walks in the dependency tree representation following Gärtner et al. (2003) whereas we only consider the downward paths.

Overall, we obtained better results than the n-gram baselines, DTK and VTK, especially with syntactic phrases. VTK shows good performance across all datasets but its computation was more than 700% slower than with our kernel. Regarding the phrase kernels, PK generally produced better results than CK, implying that the semantic linearity and ontological relation encoded in the embedding is not sufficient enough and treating them separately is more beneficial. However, we believe CK has more room for improvement with the use of more accurate phrase embeddings such as the ones from Le and Mikolov (2014), Yin and Schütze (2014) and Yu and Dredze (2015).

There was little contribution to the accuracy from non-unigram features, indicating that large part of the performance improvement is credited to the word embedding resolving the sparsity issue.

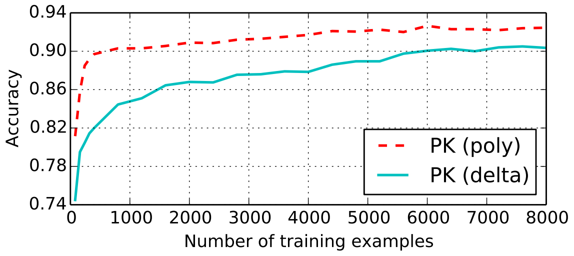


Figure 2: Test accuracy vs. number of training examples for our kernel and the bigram baseline.

This can be well observed with the following experiment on the number of training examples. Figure 2 shows the accuracy on the same test set (20% of the dataset) when the learning was done on 1% to 100% of the training set (80% of the dataset) for the bigram baseline and our bigram PK phrase kernel, both with dependency tree representation, on PL04. We see that our kernel starts to plateau earlier in the learning curve than the baseline and also reaches the maximum baseline accuracy with only about 1,500 training examples.

4.4 Computational complexity

Solving the SVM in the primal for the baselines requires $\mathcal{O}(NnL)$ time where N is the number of training documents, n is the number of words in the document and L is the maximum phrase length considered. The computation of VTK reduces down to power series computation of the adjacency matrix of the product graph, and since we require kernel values between all documents, it requires $\mathcal{O}(N^2(n^2d + n^4L))$ time where d is the dimension of the word embedding space.

Our kernel is the sum of phrase kernels (PhK) starting from every pair of nodes between two sentences, for all phrase lengths (l) and distortions (λ_2) under consideration. By storing intermediate values of composite vectors, a phrase kernel can be computed in $\mathcal{O}(d)$ time regardless of the phrase length, therefore the whole computation process has $\mathcal{O}(N^2n^2L^2d)$ complexity. Although our kernel has the squared terms of the baseline's complexity, we are tackling the sparsity issue that arises with short text (small n) or when little training data is available (small N). Moreover, we were able to get better results with only bigrams (small L). Hence, the loss in efficiency is acceptable considering significant gains in effectiveness.

5 Conclusion

In this paper, we proposed a novel convolutional sentence kernel based on word embeddings that overcomes the sparsity issue, which arises when classifying short documents or when little training data is available. We described a general framework that can encompass the standard n-gram baseline approach as well as more relaxed versions with smoother word and phrase kernels. It achieved significant improvements over the baselines across all datasets when taking into account the additional information from the latent word similarity (word embeddings) and the syntactic structure (dependency tree).

Future work might involve designing new kernels for syntactic parse trees with appropriate similarity measures between non-terminal nodes as well as exploring recently proposed phrase embeddings for more accurate phrase kernels.

References

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*, EMNLP-CoNLL '12, pages 546–556. ACL.

John Blitzer, Mark Dredze, and Fernando Pereira. 2007. Biographies, bollywood, boomboxes and blenders: Domain adaptation for sentiment classification. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, ACL '07, pages 440–447. ACL.

Stephan Bloehdorn and Alessandro Moschitti. 2007. Structure and Semantics for Expressive Text Kernels. In *Proceedings of the 16th ACM international conference on Information and knowledge management*, CIKM '07, pages 861–864. ACM.

Stephan Bloehdorn, Roberto Basili, Marco Cammisa, and Alessandro Moschitti. 2006. Semantic kernels for text classification based on topological measures of feature similarity. In *Proceedings of the 6th IEEE International Conference on Data Mining*, ICDM '06, pages 808–812. IEEE Computer Society.

Chih-Chung Chang and Chih-Jen Lin. 2011. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3):27:1–27:27.

Michael Collins and Nigel Duffy. 2001. Convolution Kernels for Natural Language. In *Advances in Neural Information Processing Systems 14*, NIPS '01, pages 625–632. The MIT Press.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural Language Processing (Almost) from Scratch. *Journal of Machine Learning Research*, 12:2493–2537, November.

Danilo Croce, Alessandro Moschitti, and Roberto Basili. 2011. Structured Lexical Similarity via Convolution Kernels on Dependency Trees. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 1034–1046. ACL.

Aron Culotta and Jeffrey Sorensen. 2004. Dependency Tree Kernels for Relation Extraction. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*, ACL '04, pages 423–429. ACL.

Thomas Gärtner, Peter Flach, and Stefan Wrobel. 2003. On graph kernels: Hardness results and efficient alternatives. In *Proceedings of the Annual Conference on Computational Learning Theory*, COLT '03, pages 129–143.

Thorsten Joachims. 1998. Text categorization with Support Vector Machines: Learning with many relevant features. In *Proceedings of the 10th European Conference on Machine Learning*, ECML '98, pages 137–142.

Tom Kenter and Maarten de Rijke. 2015. Short text similarity with word embeddings. In *Proceedings of the 24th ACM international conference on Information and knowledge management*, CIKM '15. ACM.

Quoc Le and Tomas Mikolov. 2014. Distributed Representations of Sentences and Documents. In *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *ICML '14*, pages

1188–1196. JMLR Workshop and Conference Proceedings.

Omer Levy and Yoav Goldberg. 2014. Dependency-Based Word Embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, volume 2 of *ACL '14*, pages 302–308. ACL.

Omer Levy, Yoav Goldberg, and Ido Dagan. 2015. Improving distributional similarity with lessons learned from word embeddings. *Transactions of the Association for Computational Linguistics*, 3:211–225.

Dimitrios Mavroeidis, George Tsatsaronis, Michalis Vazirgiannis, Martin Theobald, and Gerhard Weikum. 2005. Word Sense Disambiguation for Exploiting Hierarchical Thesauri in Text Classification. In *Proceedings of the 9th European conference on Principles and Practice of Knowledge Discovery in Databases*, ECML PKDD '05, pages 181–192. Springer-Verlag Berlin.

Tomas Mikolov, Kai Chen, Greg S. Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. In *Proceedings of Workshop at International Conference on Learning Representations*, ICLR '13.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeffrey Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26*, NIPS '13, pages 3111–3119. Neural Information Processing Systems.

George A. Miller. 1995. WordNet: A Lexical Database for English. *Communications of the ACM*, 38(11):39–41, November.

Preslav Nakov, Zornitsa Kozareva, Alan Ritter, Sara Rosenthal, Veselin Stoyanov, and Theresa Wilson. 2013. Semeval-2013 task 2: Sentiment analysis in twitter. In *Proceedings of the 7th International Workshop on Semantic Evaluation*, SemEval-2013.

Bo Pang and Lilian Lee. 2004. A sentimental education: sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, ACL '04, pages 271–278. ACL.

Bo Pang and Lilian Lee. 2005. Seeing stars: exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 115–124. ACL.

Georges Siolas and Florence d'Alché Buc. 2000. Support Vector Machines Based on a Semantic Kernel for Text Categorization. In *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks*, volume 5 of *IJCNN '00*, pages 205–209. IEEE Computer Society.

Yangqiu Song and Dan Roth. 2015. Unsupervised Sparse Vector Densification for Short Text Similarity. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, NAACL-Short '15, pages 1275–1280. ACL.

Shashank Srivastava, Dirk Hovy, and Eduard H. Hovy. 2013. A Walk-Based Semantically Enriched Tree Kernel Over Distributed Word Representations. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, EMNLP '13, pages 1411–1416. ACL.

Stephen Tratz and Eduard H. Hovy. 2011. A Fast, Accurate, Non-projective, Semantically-enriched Parser. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 1257–1268. ACL.

Daniele Vitale, Paolo Ferragina, and Ugo Scaiella. 2012. Classification of Short Texts by Deploying Topical Annotations. In *Proceedings of the 34th European Conference on Information Retrieval*, ECIR'12, pages 376–387. Springer-Verlag.

Janyce M. Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language resources and evaluation*, 39(2-3):165–210.

Yiming Yang and Xin Liu. 1999. A Re-examination of Text Categorization Methods. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '99, pages 42–49. ACM.

Wenpeng Yin and Hinrich Schütze. 2014. An Exploration of Embeddings for Generalized Phrases. In *Proceedings of the ACL Student Research Workshop*, ACLstudent '14, pages 41–47. ACL.

Mo Yu and Mark Dredze. 2015. Learning Composition Models for Phrase Embeddings. *Transactions of the Association for Computational Linguistics*, 3:227–242.

Dmitry Zelenko, Chinatsu Aone, and Anthony Richardella. 2003. Kernel Methods for Relation Extraction. *Journal of Machine Learning Research*, 3:1083–1106.