Low-Rank Tensors for Verbs in Compositional Distributional Semantics

Daniel Fried, Tamara Polajnar, and **Stephen Clark**
University of Cambridge
Computer Laboratory
{df345,tp366,sc609}@cam.ac.uk

Abstract

Several compositional distributional semantic methods use tensors to model multi-way interactions between vectors. Unfortunately, the size of the tensors can make their use impractical in large-scale implementations. In this paper, we investigate whether we can match the performance of full tensors with low-rank approximations that use a fraction of the original number of parameters. We investigate the effect of low-rank tensors on the transitive verb construction where the verb is a third-order tensor. The results show that, while the low-rank tensors require about two orders of magnitude fewer parameters per verb, they achieve performance comparable to, and occasionally surpassing, the unconstrained-rank tensors on sentence similarity and verb disambiguation tasks.

1 Introduction

Distributional semantic methods represent word meanings by their contextual distributions, for example by computing word-context co-ocurrence statistics (Schütze, 1998; Turney and Pantel, 2010) or by learning vector representations for words as part of a context prediction model (Bengio et al., 2003; Collobert et al., 2011; Mikolov et al., 2013). Recent research has also focused on compositional distributional semantics (CDS): combining the distributional representations for words, often in a syntax-driven fashion, to produce distributional representations of phrases and sentences (Mitchell and Lapata, 2008; Baroni and Zamparelli, 2010; Socher et al., 2012; Zanzotto and Dell'Arciprete, 2012).

One method for CDS is the Categorial framework (Coecke et al., 2011; Baroni et al., 2014), where each word is represented by a tensor whose order is determined by the Categorial Grammar type of the word. For example, nouns are an atomic type represented by a vector, and adjectives are matrices that act as functions transforming a noun vector into another noun vector (Baroni and Zamparelli, 2010). A transitive verb is a third-order tensor that takes the noun vectors representing the subject and object and returns a vector in the sentence space (Polajnar et al., 2014).

However, a concrete implementation of the Categorial framework requires setting and storing the values, or parameters, defining these matrices and tensors. These parameters can be quite numerous for even low-dimensional sentence spaces. For example, a third-order tensor for a given transitive verb, mapping two 100-dimensional noun spaces to a 100-dimensional sentence space, would have 100^3 parameters in its full form. All of the more complex types have corresponding tensors of higher order, and therefore a barrier to the practical implementation of this framework is the large number of parameters required to represent an extended vocabulary and a variety of grammatical constructions.

We aim to reduce the size of the models by demonstrating that reduced-rank tensors, which can be represented in a form requiring fewer parameters, can capture the semantics of complex types as well as the full-rank tensors do. We base our experiments on the transitive verb construction for which there are established tasks and datasets (Grefenstette and Sadrzadeh, 2011; Kartsaklis and Sadrzadeh, 2014).

Previous work on the transitive verb construction within the Categorial framework includes a two-step linear-regression method for the construction of the full verb tensors (Grefenstette et al., 2013) and a multi-linear regression method combined with a two-dimensional plausibility space (Polajnar et al., 2014). Polajnar et al. (2014)

also introduce several alternative ways of reducing the number of tensor parameters by using matrices. The best performing method uses two matrices, one representing the subject-verb interactions and the other the verb-object interactions. Some interaction between the subject and the object is re-introduced through a softmax layer. A similar method is presented in Paperno et al. (2014). Milajevs et al. (2014) use vectors generated by a neural language model to construct verb matrices and several different composition operators to generate the composed subject-verb-object sentence representation.

In this paper, we use tensor rank decomposition (Kolda and Bader, 2009) to represent each verb's tensor as a sum of tensor products of vectors. We learn the component vectors and apply the composition without ever constructing the full tensors and thus we are able to improve on both memory usage and efficiency. This approach follows recent work on using low-rank tensors to parameterize models for dependency parsing (Lei et al., 2014) and semantic role labelling (Lei et al., 2015). Our work applies the same tensor rank decompositions, and similar optimization algorithms, to the task of constructing a syntax-driven model for CDS. Although we focus on the Categorial framework, the low-rank decomposition methods are also applicable to other tensor-based semantic models including Van de Cruys (2010), Smolensky and Legendre (2006), and Blacoe et al. (2013).

2 Model

Tensor Models for Verbs We model each transitive verb as a bilinear function mapping subject and object noun vectors, each of dimensionality N, to a single sentence vector of dimensionality S (Coecke et al., 2011; Maillard et al., 2014) representing the composed subject-verb-object (SVO) triple. Each transitive verb has its own third-order tensor, which defines this bilinear function. Consider a verb V with associated tensor $\mathcal{V} \in \mathbb{R}^{S\times N\times N}$, and vectors $\mathbf{s} \in \mathbb{R}^N$, $\mathbf{o} \in \mathbb{R}^N$ for subject and object nouns, respectively. Then the compositional representation for the subject, verb, and object is a vector $V(\mathbf{s},\mathbf{o}) \in \mathbb{R}^S$, produced by applying *tensor contraction* (the higher-order analogue of matrix multiplication) to the verb tensor and two noun vectors. The l^{th} component of the vector for the SVO triple is given by

$$V(\mathbf{s},\mathbf{o})_l = \sum_{j,k} \mathcal{V}_{ljk}\mathbf{o}_k\mathbf{s}_j \qquad (1)$$

We aim to learn distributional vectors $\mathbf{s}$ and $\mathbf{o}$ for subjects and objects, and tensors $\mathcal{V}$ for verbs, such that the output vectors $V(\mathbf{s},\mathbf{o})$ are distributional representations of the entire SVO triple. While there are several possible definitions of the sentence space (Clark, 2013; Baroni et al., 2014), we follow previous work (Grefenstette et al., 2013) by using a contextual sentence space consisting of content words that occur within the same sentences as the SVO triple.

Low-Rank Tensor Representations Following Lei et al. (2014), we represent each verb's tensor using a low-rank *canonical polyadic (CP) decomposition* to reduce the numbers of parameters that must be learned during training. As a higher-order analogue to singular value decomposition for matrices, CP decomposition factors a tensor into a sum of R tensor products of vectors.[1] Given a third-order tensor $\mathcal{V} \in \mathbb{R}^{S\times N\times N}$, the CP decomposition of $\mathcal{V}$ is:

$$\mathcal{V} = \sum_{r=1}^{R} \mathbf{P}_r \otimes \mathbf{Q}_r \otimes \mathbf{R}_r \qquad (2)$$

where $\mathbf{P} \in \mathbb{R}^{R\times S}, \mathbf{Q} \in \mathbb{R}^{R\times N}, \mathbf{R} \in \mathbb{R}^{R\times N}$ are parameter matrices, $\mathbf{P}_r$ gives the rth row of matrix $\mathbf{P}$, and $\otimes$ is the tensor product.

The smallest R that allows the tensor to be expressed as this sum of outer products is the *rank* of the tensor (Kolda and Bader, 2009). By fixing a value for R that is sufficiently small compared to S and N (forcing the verb tensor to have rank of at most R), and directly learning the parameters of the low-rank approximation using gradient-based optimization, we learn a low-rank tensor requiring fewer parameters without ever having to store the full tensor.

In addition to reducing the number of parameters, representing tensors in this form allows us to formulate the verb tensor's action on noun vectors as matrix multiplication. For a tensor in the form of Eq. (2), the output SVO vector is given by

$$V(\mathbf{s},\mathbf{o}) = \mathbf{P}^\top(\mathbf{Qs} \odot \mathbf{Ro}) \qquad (3)$$

where $\odot$ is the elementwise vector product.

[1]However, unlike matrix singular value decomposition, the component vectors in the CP decomposition are not necessarily orthonormal.

3 Training

We train the compositional model for verbs in three steps: extracting transitive verbs and their subject and object nouns from corpus data, producing distributional vectors for the nouns and the SVO triples, and then learning parameters of the verb functions, which map the nouns to the SVO triple vectors.

Corpus Data We extract SVO triples from an October 2013 download of Wikipedia, tokenized using Stanford CoreNLP (Manning et al., 2014), lemmatized with the Morpha lemmatizer (Minnen et al., 2001), and parsed using the C&C parser (Curran et al., 2007). We filter the SVO triples to a set containing 345 distinct verbs: the verbs from our test datasets, along with some additional high-frequency verbs included to produce more representative sentence spaces. For each verb, we selected up to 600 triples which occurred more than once and contained subject and object nouns that occurred at least 100 times (to allow sufficient context to produce a distributional representation for the triple). This resulted in approximately 150,000 SVO triples overall.

Distributional Vectors We produce two types of distributional vectors for nouns and SVO triples using the Wikipedia corpus. Since these methods for producing distributional vectors for the SVO triples require that the triples occur in a corpus of text, the methods are not a replacement for a compositional framework that can produce representations for previously unseen expressions. However, they can be used to generate data to train such a model, as we will describe.

1) Count vectors (SVD): we count the number of times each noun or SVO triple co-occurs with each of the 10,000 most frequent words (excluding stopwords) in the Wikipedia corpus, using sentences as context boundaries. If the verb in the SVO triple is itself a content word, we do not include it as context for the triple. This produces one set of context vectors for nouns and another for SVO triples. We weight entries in these vectors using the t-test weighting scheme (Curran, 2004; Polajnar and Clark, 2014), and then reduce the vectors to 100 dimensions via singular value decomposition (SVD), decomposing the noun vectors and SVO vectors separately.

2) Prediction vectors (PV): we train vector embeddings for nouns and SVO triples by adapting the Paragraph Vector distributed bag of words method of Le and Mikolov (2014), an extension of the skip-gram model of Mikolov et al. (2013). In our experiments, given an SVO triple, the model must predict contextual words sampled from all sentences containing that triple. In the process, the model learns vector embeddings for both the SVO triples and for the words in the sentences such that SVO vectors have a high dot product with their contextual word vectors. While previous work (Milajevs et al., 2014) has used prediction-based vectors for words in a tensor-based CDS model, ours uses prediction-based vectors for both words and phrases to train a tensor regression model.

We learn 100-dimensional vectors for nouns and SVO triples with a modified version of `word2vec`,[2] using the hierarchical sampling method with the default hyperparameters and 20 iterations through the training data.

Training Methods We learn the tensor $\mathcal{V}$ of parameters for a given verb V using multi-linear regression, treating the noun vectors $\mathbf{s}$ and $\mathbf{o}$ as input and the composed SVO triple vector $V(\mathbf{s}, \mathbf{o})$ as the regression output. Let M_V be the number of training instances for V, where the i^{th} instance is a triple of vectors $\left(\mathbf{s}^{(i)}, \mathbf{o}^{(i)}, \mathbf{t}^{(i)}\right)$, which are the distributional vectors for the subject noun, object noun, and the SVO triple, respectively. We aim to learn a verb tensor $\mathcal{V}$ (either in full or in decomposed, low-rank form) that minimizes the mean of the squared residuals between the predicted SVO vectors $V(\mathbf{s}^{(i)}, \mathbf{o}^{(i)})$ and those vectors obtained distributionally from the corpus, $\mathbf{t}^{(i)}$. Specifically, we attempt to minimize the following loss function:

$$L(V) = \frac{1}{M_V} \sum_{i=1}^{M_V} ||V(\mathbf{s}^{(i)}, \mathbf{o}^{(i)}) - \mathbf{t}^{(i)}||_2^2 \qquad (4)$$

$V(\mathbf{s}, \mathbf{o})$ is given by Eq. (1) for full tensors, and by Eq. (3) for tensors represented in low-rank form.

In both the low-rank and full-rank tensor learning, we use mini-batch ADADELTA optimization (Zeiler, 2012) up to a maximum of 500 iterations through the training data, which we found to be sufficient for convergence for every verb. Rather than placing a regularization penalty on the tensor parameters, we use early stopping if the loss

[2]`https://groups.google.com/d/msg/word2vec-toolkit/Q49FIrNOQRo/J6KG8mUj45sJ`

increases on a validation set consisting of 10% of the available SVO triples for each verb.

For low-rank tensors, we compare seven different maximal ranks: R=1, 5, 10, 20, 30, 40 and 50. To learn the parameters of the low-rank tensors, we use an alternating optimization method (Kolda and Bader, 2009; Lei et al., 2014): performing gradient descent on one of the parameter matrices (for example $\mathbf{P}$) to minimize the loss function while holding the other two fixed ($\mathbf{Q}$ and $\mathbf{R}$), then repeating for the other parameter matrices in turn. The parameter matrices are randomly initialized.[3]

4 Evaluation

We compare the performance of the low-rank tensors against full tensors on two tasks. Both tasks require the model to rank pairs of sentences each consisting of a subject, transitive verb, and object by the semantic similarity of the sentences in the pair. The gold standard ranking is given by similarity scores provided by human evaluators and the scores are not averaged among the annotators. The model ranking is evaluated against the ranking from the gold standard similarity judgements using Spearman's ρ.

The verb disambiguation task (GS11) (Grefenstette and Sadrzadeh, 2011) involves distinguishing between senses of an ambiguous verb, given subject and object nouns as context. The dataset consists of 200 sentence pairs, where the two sentences in each pair have the same subject and object but differ in the verb. Each of these pairs was ranked by human evaluators on a 1-7 similarity scale so that properly disambiguated pairs (e.g. *author write book – author publish book*) have higher similarity scores than improperly disambiguated pairs (e.g. *author write book – author spell book*).

The transitive sentence similarity dataset (Kartsaklis and Sadrzadeh, 2014) consists of 72 subject-verb-object sentences arranged into 108 sentence pairs. As in GS11, each pair has a gold standard semantic similarity score on a 1-7 scale. For example, the pair *medication achieve result – drug produce effect* has a high similarity rating, while *author write book – delegate buy land* has a low rating. In this dataset, however, the two sentences in each pair have no lexical overlap: neither subjects, objects, nor verbs are shared.

[3]Since the low-rank tensor loss is non-convex, we suspect that parameter initialization may produce better results.

	GS11		KS14		# tensor
	SVD	PV	SVD	PV	params.
Add.	0.13	0.14	**0.55**	**0.56**	–
Mult.	0.13	0.14	0.09	0.27	–
R=1	0.10	0.05	0.18	0.30	300
R=5	0.26	0.30	0.28	0.40	1.5K
R=10	0.29	0.32	0.26	0.45	3K
R=20	0.31	0.34	0.39	0.44	6K
R=30	0.28	0.33	0.32	0.46	9K
R=40	0.32	0.30	0.31	**0.52**	12K
R=50	**0.34**	0.32	**0.42**	0.51	15K
Full	0.29	**0.36**	0.41	**0.52**	1M

Table 1: Model performance on the verb disambiguation (GS11) and sentence similarity (KS14) tasks, given by Spearman's ρ, and the number of parameters needed to represent each verb's tensor. We show the highest tensor result for each task and vector set in bold (and also bold the baseline when it outperforms the tensor method).

5 Results

Table 1 displays correlations between the systems' scores and human SVO similarity judgements on the verb disambiguation (GS11) and sentence similarity (KS14) tasks, for both the count (SVD) and prediction vectors (PV). We also give results for simple composition of word vectors using elementwise addition and multiplication (Mitchell and Lapata, 2008) (using verb vectors produced in the same manner as for nouns). As is consistent with prior work, the tensor-based models are surpassed by vector addition on the KS14 dataset (Milajevs et al., 2014), but perform better than both addition and multiplication on the GS11 dataset.[4]

Unsurprisingly, the rank-1 tensor has lowest performance for both tasks and vector sets, and performance generally increases as we increase the maximal rank R. The full tensor achieves the best, or tied for the best, performance on both tasks when using the PV vectors. However, for the SVD vectors, low-rank tensors surpass the performance of the full-rank tensor for R=40 and R=50

[4]The results in this table are not directly comparable with Milajevs et al. (2014), who compare against *averaged* annotator scores. Comparing against averaged annotator scores, our best result on GS11 is 0.47 for the full-rank tensor with PV vectors, and our best non-addition result on KS14 is 0.68 for the K=40 tensor with PV vectors (the best result is addition with PV vectors, which achieves 0.71). These results exceed the scores reported for tensor-based models by Milajevs et al. (2014).

on GS11, and R=50 on KS14.

On GS11, the SVD and PV vectors have varying but mostly comparable performance, with PV having higher performance on 5 out of 8 models. However, on KS14, the PV vectors have better performance than the SVD vectors for every model by at least 0.05 points, which is consistent with prior work comparing count and predict vectors on these datasets (Milajevs et al., 2014).

The low-rank tensor models are also at least twice as fast to train as the full tensors: on a single core, training a rank-1 tensor takes about 5 seconds for each verb on average, ranks 5-50 each take between 1 and 2 minutes, and the full tensors each take about 4 minutes. Since a separate tensor is trained for each verb, this allows a substantial amount of time to be saved even when using the constrained vocabulary of 345 verbs.

6 Conclusion

We find that low-rank tensors for verbs achieve comparable or better performance than full-rank tensors on both verb disambiguation and sentence similarity tasks, while reducing the number of parameters that must be learned and stored for each verb by at least two orders of magnitude, and cutting training time in half.

While in our experiments the prediction-based vectors outperform the count-based vectors on both tasks for most models, Levy et al. (2015) indicate that tuning hyperparameters of the count-based vectors may be able to produce comparable performance. Regardless, we show that the low-rank tensors are able to achieve performance comparable to the full rank for both types of vectors. This is important for extending the model to many more grammatical types (including those with corresponding tensors of higher order than investigated here) to build a wide-coverage tensor-based semantic system using, for example, the CCG parser of Curran et al. (2007).

Acknowledgments

Daniel Fried is supported by a Churchill Scholarship. Tamara Polajnar is supported by ERC Starting Grant DisCoTex (306920). Stephen Clark is supported by ERC Starting Grant DisCoTex (306920) and EPSRC grant EP/I037512/1. We would like to thank Laura Rimell and the anonymous reviewers for their comments.

References

Marco Baroni and Roberto Zamparelli. 2010. Nouns are vectors, adjectives are matrices: Representing adjective-noun constructions in semantic space. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing (EMNLP 2010)*, Cambridge, Massachusetts.

Marco Baroni, Raffaela Bernardi, and Roberto Zamparelli. 2014. Frege in space: A program of compositional distributional semantics. *Linguistic Issues in Language Technology*, 9.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. *Journal of Machine Learning Research*, (3):1137–1155.

William Blacoe, Elham Kashefi, and Mirella Lapata. 2013. A quantum-theoretic approach to distributional semantics. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2013)*, Atlanta, Georgia.

Stephen Clark. 2013. Type-driven syntax and semantics for composing meaning vectors. *Quantum Physics and Linguistics: A Compositional, Diagrammatic Discourse*, pages 359–377.

Bob Coecke, Mehrnoosh Sadrzadeh, and Stephen Clark. 2011. Mathematical foundations for a compositional distributional model of meaning. *Linguistic Analysis*, 36(1-4):345–384.

R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12:2493–2537.

James R Curran, Stephen Clark, and Johan Bos. 2007. Linguistically motivated large-scale NLP with C&C and Boxer. In *Proceedings of the Demonstration Session of the 45th Annual Meeting of the Association for Computational Linguistics (ACL 2007)*, Prague, Czech Republic.

James R. Curran. 2004. *From distributional to semantic similarity*. Ph.D. thesis, University of Edinburgh.

Edward Grefenstette and Mehrnoosh Sadrzadeh. 2011. Experimenting with transitive verbs in a DisCoCat. In *Proceedings of the 2011 Workshop on Geometrical Models of Natural Language Semantics (GEMS 2011)*, Edinburgh, Scotland.

Edward Grefenstette, Georgiana Dinu, Yao-Zhong Zhang, Mehrnoosh Sadrzadeh, and Marco Baroni. 2013. Multi-step regression learning for compositional distributional semantics. In *Proceedings of the 10th International Conference on Computational Semantics (IWCS 2013)*, Pottsdam, Germany.

Dimitri Kartsaklis and Mehrnoosh Sadrzadeh. 2014. A study of entanglement in a categorical framework of natural language. In *Proceedings of the 11th Workshop on Quantum Physics and Logic (QPL 2014)*, Kyoto, Japan, June.

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

Quoc V. Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on Machine Learning (ICML 2014)*, Beijing, China.

Tao Lei, Yu Xin, Yuan Zhang, Regina Barzilay, and Tommi Jaakkola. 2014. Low-rank tensors for scoring dependency structures. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL 2014)*, Baltimore, Maryland.

Tao Lei, Yuan Zhang, Lluis Marquez, Alessandro Moschitti, and Regina Barzilay. 2015. High-order low-rank tensors for semantic role labeling. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics – Human Language Technologies (NAACL-HLT 2015)*, Denver, Colorado.

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.

Jean Maillard, Stephen Clark, and Edward Grefenstette. 2014. A type-driven tensor-based semantics for CCG. In *Proceedings of the EACL 2014 Type Theory and Natural Language Semantics Workshop (TTNLS)*, Gothenburg, Sweden.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics (ACL 2014): System Demonstrations*, pages 55–60, Baltimore, Maryland.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Neural Information Processing Systems (NIPS 2013)*.

Dmitrijs Milajevs, Dimitri Kartsaklis, Mehrnoosh Sadrzadeh, and Matthew Purver. 2014. Evaluating neural word representations in tensor-based compositional settings. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014)*.

Guido Minnen, John Carroll, and Darren Pearce. 2001. Applied morphological processing of English. *Natural Language Engineering*, 7(03):207–223.

Jeff Mitchell and Mirella Lapata. 2008. Vector-based models of semantic composition. In *Proceedings of the 46th Annual Meeting of the Assocation for Computational Linguistics: Human Language Technologies (ACL-08: HLT)*, Columbus, Ohio.

Denis Paperno, Nghia The Pham, and Marco Baroni. 2014. A practical and linguistically-motivated approach to compositional distributional semantics. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL 2014)*, Baltimore, Maryland.

Tamara Polajnar and Stephen Clark. 2014. Improving distributional semantic vectors through context selection and normalisation. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2014)*, Gothenburg, Sweden.

Tamara Polajnar, Luana Fagarasan, and Stephen Clark. 2014. Reducing dimensions of tensors in type-driven distributional semantics. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014)*, Doha, Qatar.

Hinrich Schütze. 1998. Automatic word sense discrimination. *Computational Linguistics*, 24(1):97–124.

Paul Smolensky and Geraldine Legendre. 2006. *The Harmonic Mind: from neural computation to optimality-theoretic grammar*. MIT Press, Cambridge, MA.

Richard Socher, Brody Huval, Christopher D. Manning, and Andrew Y. Ng. 2012. Semantic Compositionality Through Recursive Matrix-Vector Spaces. In *Proceedings of the 2012 Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2012)*, Jeju Island, Korea.

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

Tim Van de Cruys. 2010. A non-negative tensor factorization model for selectional preference induction. *Journal of Natural Language Engineering*, 16(4):417–437.

Fabio M Zanzotto and Lorenzo Dell'Arciprete. 2012. Distributed tree kernels. In *Proceedings of the 29th International Conference on Machine Learning (ICML 2012)*, Edinburgh, Scotland.

Matthew D Zeiler. 2012. ADADELTA: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*.