Exploiting Domain Knowledge via Grouped Weight Sharing with Application to Text Categorization

Ye Zhang[1]   **Matthew Lease**[2]   **Byron C. Wallace**[3]

[1]Department of Computer Science, University of Texas at Austin
[2]School of Information, University of Texas at Austin
[3]College of Computer & Information Science, Northeastern University
`yezhang@utexas.edu, ml@utexas.edu, byron@ccs.neu.edu`

Abstract

A fundamental advantage of neural models for NLP is their ability to learn representations from scratch. However, in practice this often means ignoring existing external linguistic resources, e.g., Word-Net or domain specific ontologies such as the Unified Medical Language System (UMLS). We propose a general, novel method for exploiting such resources via *weight sharing*. Prior work on weight sharing in neural networks has considered it largely as a means of model compression. In contrast, we treat weight sharing as a flexible mechanism for incorporating prior knowledge into neural models. We show that this approach consistently yields improved performance on classification tasks compared to baseline strategies that do not exploit weight sharing.

1 Introduction

Neural models are powerful in part due to their ability to learn good representations of raw textual inputs, mitigating the need for extensive task-specific feature engineering (Collobert et al., 2011). However, a downside of learning from scratch is failing to capitalize on prior linguistic or semantic knowledge, often encoded in existing resources such as ontologies. Such prior knowledge can be particularly valuable when estimating highly flexible models. In this work, we address how to exploit known relationships between words when training neural models for NLP tasks.

We propose exploiting the *feature-hashing* trick, originally proposed as a means of neural network compression (Chen et al., 2015). Here we instead view the partial parameter sharing induced by feature hashing as a flexible mechanism for tying together network node weights that we believe
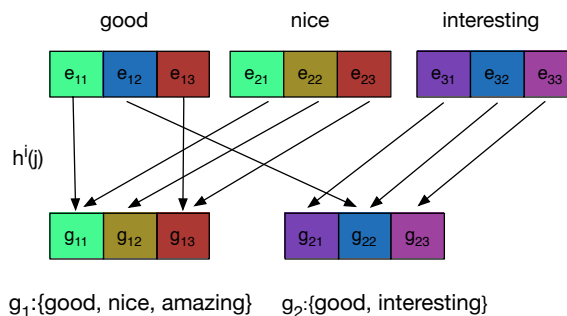

Figure 1: An example of grouped partial weight sharing. Here there are two groups. We stochastically select embedding weights to be shared between words belonging to the same group(s).

to be similar *a priori*. In effect, this acts as a regularizer that constrains the model to learn weights that agree with the domain knowledge codified in external resources like ontologies.

More specifically, as external resources we use Brown clusters (Brown et al., 1992), WordNet (Miller, 1995) and the Unified Medical Language System (UMLS) (Bodenreider, 2004). From these we derive groups of words with similar meaning. We then use feature hashing to share a subset of weights between the embeddings of words that belong to the same semantic group(s). This forces the model to respect prior domain knowledge, insofar as words similar under a given ontology are compelled to have similar embeddings.

Our contribution is a novel, simple and flexible method for injecting domain knowledge into neural models via stochastic weight sharing. Results on seven diverse classification tasks (three sentiment and four biomedical) show that our method consistently improves performance over (1) baselines that fail to capitalize on domain knowledge, and (2) an approach that uses *retrofitting* (Faruqui et al., 2014) as a preprocessing step to encode domain knowledge prior to training.

155

2 Grouped Weight Sharing

We incorporate similarity relations codified in existing resources (here derived from Brown clusters, SentiWordNet and the UMLS) as prior knowledge in a Convolutional Neural Network (CNN).[1] To achieve this we construct a shared embedding matrix such that words known *a priori* to be similar are constrained to share some fraction of embedding weights.

Concretely, suppose we have N groups of words derived from an external resource. Note that one could derive such groups in several ways; e.g., using the synsets in SentiWordNet. We denote groups by $\{g_1, g_2, ..., g_N\}$. Each group is associated with an embedding $\mathbf{g}_{g_i}$, which we initialize by averaging the pre-trained embeddings of each word in the group.

To exploit both grouped and independent word weights, we adopt a two-channel CNN model (Zhang et al., 2016b). The embedding matrix of the first channel is initialized with pre-trained word vectors. We denote this input by $\mathbf{E}^p \in \mathbb{R}^{V \times d}$ (V is the vocabulary size and d the dimension of the word embeddings). The second channel input matrix is initialized with our proposed weight-sharing embedding $\mathbf{E}^s \in \mathbb{R}^{V \times d}$. $\mathbf{E}^s$ is initialized by drawing from both $\mathbf{E}^p$ and the external resource following the process we describe below.

Given an input text sequence of length l, we construct sequence embedding representations $\mathbf{W}^p \in \mathbb{R}^{l \times d}$ and $\mathbf{W}^s \in \mathbb{R}^{l \times d}$ using the corresponding embedding matrices. We then apply independent sets of linear convolution filters on these two matrices. Each filter will generate a feature map vector $\mathbf{v} \in \mathbb{R}^{l-h+1}$ (h is the filter height). We perform 1-max pooling over each $\mathbf{v}$, extracting one scalar per feature map. Finally, we concatenate scalars from all of the feature maps (from both channels) into a feature vector which is fed to a softmax function to predict the label (Figure 2).

We initialize $\mathbf{E}^s$ as follows. Each row $\mathbf{e}_i \in \mathbb{R}^d$ of $\mathbf{E}_s$ is the embedding of word i. Words may belong to one or more groups. A mapping function $G(i)$ retrieves the groups that word i belongs to, i.e., $G(i)$ returns a subset of $\{g_1, g_2, ..., g_N\}$, which we denote by $\{g_1^{(i)}, g_2^{(i)}...g_K^{(i)}\}$, where K is the number of groups that contain word i. To initialize $\mathbf{E}^s$, for each dimension j of each word embedding $\mathbf{e}_i$, we use a hash function h^i to map
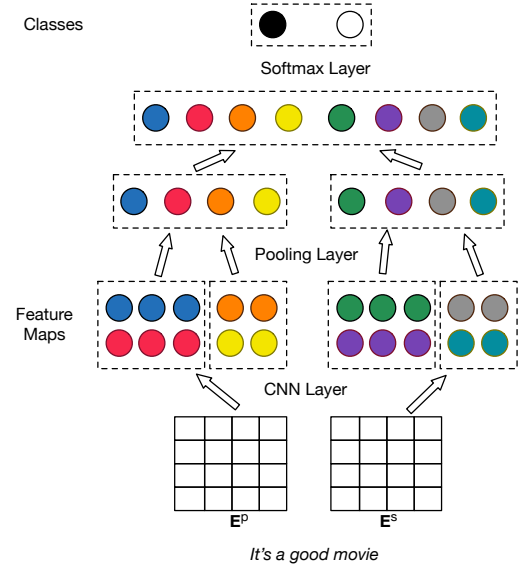

Figure 2: Proposed two-channel model. The first channel input is a standard pre-trained embedding matrix. The second channel receives a partially shared embedding matrix constructed using external linguistic resources.

(hash) the index j to one of the K group IDs: $h^i : \mathbb{N} \to \{g_1^{(i)}, g_2^{(i)}...g_K^{(i)}\}$. Following (Weinberger et al., 2009; Shi et al., 2009), we use a second hash function b to remove bias induced by hashing. This is a signing function, i.e., it maps (i, j) tuples to $\{+1, -1\}$ [2]. We then set $\mathbf{e}_{i,j}$ to the product of $\mathbf{g}_{h^i(j),j}$ and $b(i, j)$. h and b are both approximately uniform hash functions. **Algorithm 1** provides the full initialization procedure.

Algorithm 1 Initialization of $\mathbf{E}^s$

1: **for** i in $\{1, \ldots, V\}$ **do**
2: $\quad \{g_1^{(i)}, g_2^{(i)}, \ldots, g_K^{(i)}\} := G(i)$.
3: $\quad$ **for** $j \in \{1, \ldots, d\}$ **do**
4: $\qquad \mathbf{e}_{i,j} := \mathbf{g}_{h^i(j),j} \cdot b(i, j)$
5: $\quad$ **end for**
6: **end for**

For illustration, consider Figure 1. Here g_1 contains three words: *good*, *nice* and *amazing*, while g_2 has two words: *good* and *interesting*. The group embeddings $\mathbf{g}_{g_1}$, $\mathbf{g}_{g_2}$ are initialized as averages over the pre-trained embeddings of the words they comprise. Here, embedding parameters $\mathbf{e}_{1,1}$ and $\mathbf{e}_{2,1}$ are both mapped to $\mathbf{g}_{g_1,1}$, and thus share this value. Similarly, $\mathbf{e}_{1,3}$ and $\mathbf{e}_{2,3}$ will share value at $\mathbf{g}_{g_1,3}$. We have elided the second hash function b from this figure for simplicity.

---

[1]The idea of sharing weights to reflect known similarity is general and could be applied with other neural architectures.

[2]Empirically, we found that using this signing function does not affect performance.

During training, we update $\mathbf{E}^p$ as usual using back-propagation (Rumelhart et al., 1986). We update $\mathbf{E}^s$ and group embeddings $\mathbf{g}$ in a manner similar to Chen et al. (2015). In the forward propagation before each training step (mini-batch), we derive the value of $\mathbf{e}_{i,j}$ from $\mathbf{g}$:

$$\mathbf{e}_{i,j} := \mathbf{g}_{h^i(j),j} * b(i,j) \qquad (1)$$

We use this newly updated $\mathbf{e}_{i,j}$ to perform forward propagation in the CNN.

During backward propagation, we first compute the gradient of $\mathbf{E}^s$, and then we use this to derive the gradient w.r.t $\mathbf{g}s$. To do this, for each dimension j in $\mathbf{g}_{g_k}$, we aggregate the gradients w.r.t $\mathbf{E}^s$ whose elements are mapped to this dimension:

$$\nabla \mathbf{g}_{g_k,j} := \sum_{(i,j)} \nabla \mathbf{E}^s_{i,j} \cdot \delta_{h^i(j)=g_k} \cdot b(i,j) \qquad (2)$$

where $\delta_{h^i(j)=g_k} = 1$ when $h^i(j) = g_k$, and 0 otherwise. Each training step involves executing Equations 1 and 2. Once the shared gradient is calculated, gradient descent proceeds as usual. We update all parameters aside from the shared weights in the standard way.

The number of parameters in our approach scales linearly with the number of channels. But the gradients can actually be back-propagated in a distributed way for each channel, since the convolutional and embedding layers are independent across these. Thus training time scales approximately linearly with the number of parameters in one channel (if the gradient is back-propagated in a distributed way).

3 Experimental Setup

3.1 Datasets

We use three sentiment datasets: a movie review (**MR**) dataset (Pang and Lee, 2005)[3]; a customer review (**CR**) dataset (Hu and Liu, 2004)[4]; and an opinion dataset (**MPQA**) (Wiebe et al., 2005)[5].

We also use four biomedical datasets, which concern *systematic reviews*. The task here is to classify published articles describing clinical trials as *relevant* or *not* to a well-specified clinical question. Articles deemed relevant are included in the corresponding review, which is a synthesis of all pertinent evidence (Wallace et al., 2010). We use data from reviews that concerned: clopidogrel (CL) for cardiovascular conditions (Dahabreh et al., 2013); biomarkers for assessing iron deficiency in anemia (AN) experienced by patients with kidney disease (Chung et al., 2012); statins (ST) (Cohen et al., 2006); and proton beam (PB) therapy (Terasawa et al., 2009).

3.2 Implementation Details and Baselines

We use SentiWordNet (Baccianella et al., 2010)[6] for the sentiment tasks. SentiWordNet assigns to each synset of wordnet three sentiment scores: positivity, negativity and objectivity, constrained to sum to 1. We keep only the synsets with positivity or negativity scores greater than 0, i.e., we remove synsets deemed objective. The synsets in SentiWordNet constitute our groups. We also use the Brown clustering algorithm[7] on the three sentiment datasets. We generate 1000 clusters and treat each as a group.

For the biomedical datasets, we use the Medical Subject Headings (MeSH) terms[8] attached to each abstract to classify them. Each MeSH term has a tree number indicating the path from the root in the UMLS. For example, 'Alagille Syndrome' has tree number 'C06.552.150.125'; periods denote tree splits, numbers are nodes. We induce groups comprising MeSH terms that share the same first three parent nodes, e.g., all terms with 'C06.552.150' as their tree number prefix constitute one group.

We compare our approach to several baselines. All use pre-trained embeddings to initialize $\mathbf{E}^p$, but we explore several approaches to exploiting $\mathbf{E}^s$: (1) randomly initialize $\mathbf{E}^s$; (2) initialize $\mathbf{E}^s$ to reflect the group embedding $\mathbf{g}$, but do not share weights during the training process, i.e., do not constrain their weights to be equal when we perform back-propagation; (3) use linguistic resources to *retro-fit* (Faruqui et al., 2014) the pre-trained embeddings, and use these to initialize $\mathbf{E}^s$. For *retro-fitting*, we first construct a graph derived from SentiWordNet. Then we run belief-propagation on the graph to encourage linked words to have similar vectors. This is a pre-processing step only; we do not impose weight sharing constraints during training.

---

[3] www.cs.cornell.edu/people/pabo/movie-review-data/
[4] www.cs.uic.edu/~liub/FBS/sentiment-analysis.html
[5] mpqa.cs.pitt.edu/corpora/mpqa_corpus/

[6] sentiwordnet.isti.cnr.it
[7] github.com/percyliang/brown-cluster
[8] www.nlm.nih.gov/bsd/disted/meshtutorial/

	total #instances	vocabulary size	#positive instances	#negative instances
MR	10662	18765	5331	5331
CR	3773	5340	2406	1367
MPQA	10604	6246	3311	7293
AN	5653	5554	653	5000
CL	8288	3684	768	7520
ST	3464	2965	173	3291
PB	4749	3086	243	4506

Table 1: Corpora statistics.

For the sentiment datasets we use three filter heights (3,4,5) for each of the two CNN channels. For the biomedical datasets, we use only one filter height (1), because the inputs are unstructured MeSH terms.[9] In both cases we use 100 filters of each unique height. For the sentiment datasets, we use Google word2vec (Mikolov et al., 2013)[10] to initialize $\mathbf{E}^p$. For the biomedical datasets, we use word2vec trained on biomedical texts (Moen and Ananiadou, 2013)[11] to initialize $\mathbf{E}^p$. For parameter estimation, we use Adadelta (Zeiler, 2012). Because the biomedical datasets are imbalanced, we use downsampling (Zhang et al., 2016a; Zhang and Wallace, 2015) to effectively train on balanced subsets of the data.

We developed our approach using the MR sentiment dataset, tuning our approach to constructing groups from the available resources – experiments on other sentiment datasets were run after we finalized the model and hyperparameters. Similarly, we used the anemia (AN) review as a development set for the biomedical tasks, especially w.r.t. constructing groups from MeSH terms using UMLS.

4 Results

We replicate each experiment five times (each is a 10-fold cross validation), and report the mean (min, max) across these replications. Results on the sentiment and biomedical corpora in are presented in Tables 2 and 3, respectively.[12] These exploit different external resources to induce the word groupings that in turn inform weight sharing. We report AUC for the biomedical datasets because these are highly imbalanced (see Table 1).

Our method improves performance compared to all relevant baselines (including an approach that also exploits external knowledge via retrofitting) in six of seven cases. Informing weight *initialization* using external resources improves performance independently, but additional gains are realized by also enforcing sharing during training.

We note that our aim here is not necessarily to achieve state-of-art results on any given dataset, but rather to evaluate the proposed method for incorporating external linguistic resources into neural models via weight sharing. We have therefore compared to baselines that enable us to assess this.

5 Related Work

Neural Models for NLP. Recently there has been enormous interest in neural models for NLP generally (Collobert et al., 2011; Goldberg, 2016). Most relevant to this work, simple CNN based models (which we have built on here) have proven extremely effective for text categorization (Kim, 2014; Zhang and Wallace, 2015).

Exploiting Linguistic Resources. A potential drawback to learning from scratch in end-to-end neural models is a failure to capitalize on existing knowledge sources. There have been efforts to exploit such resources specifically to induce better word vectors (Yu and Dredze, 2014; Faruqui et al., 2014; Yu et al., 2016; Xu et al., 2014). But these models do not attempt to exploit external resources jointly during training for a particular downstream task (which uses word embeddings as inputs), as we do here.

Past work on sparse linear models has shown the potential of exploiting linguistic knowledge in statistical NLP models. For example, Yogatama and Smith (2014) used external resources to inform structured, grouped regularization of log-linear text classification models, yielding improvements over standard regularization approaches. Elsewhere, Doshi-Velez et al. (2015) proposed a variant of LDA that exploits *a priori* known tree-

---

[9]For this work we are ignoring title and abstract texts.
[10]`code.google.com/archive/p/word2vec/`
[11]`bio.nlplab.org/`
[12]Sentiment task results are not directly comparable to prior work due to different preprocessing steps.

Method	MR	CR	MPQA
p only	81.02 (80.84,81.24)	84.34 (84.21,84.53)	89.41 (89.22,89.58)
p + r	81.25 (81.19,81.32)	84.33 (84.24,84.38)	89.63 (89.58,89.71)
p + retro	81.35 (81.23,81.51)	84.16 (84.09,84.28)	89.61 (89.48,89.77)
p + S (no sharing)	81.39 (81.32,81.43)	84.13 (84.06,84.21)	89.71 (89.67,89.75)
p + B (no sharing)	81.50 (81.29,81.63)	84.60 (84.53,84.66)	89.57 (89.52,89.61)
p + S (sharing)	81.69 (81.60,81.78)	84.34 (84.24,84.43)	89.84 (89.74,90.13)
p + B (sharing)	**81.83 (81.80,81.87)**	**84.68 (84.64,84.72)**	**89.97 (89.74,90.13)**

Table 2: Accuracy mean (min, max) on sentiment datasets. 'p': channel initialized with the pre-trained embeddings $\mathbf{E}^p$. 'r': channel randomly initialized. 'retro': initialized with retofitted embeddings. 'S/B (no sharing)': channel initialized with $\mathbf{E}^s$ (using SentiWordNet or Brown clusters), but weights are not shared during training. 'S/B (sharing)': proposed weight-sharing method.

Method	AN	CL	ST	PB
p only	86.63 (86.57,86.67)	88.73 (88.51,89.00)	67.15 (66.00, 67.91)	90.11 (89.46, 91.03)
p + r	85.67 (85.46,85.95)	88.87 (88.56,89.03)	67.72 (67.65,67.86)	90.12 (89.87,90.47)
p + retro	86.46 (86.32,86.65)	89.27 (88.89,90.01)	**67.78 (67.56,68.00)**	90.07 (89.92,90.20)
p + U	86.60 (86.32,87.01)	88.93 (88.67,89.13)	67.78 (67.71,67.85)	90.23 (89.84,90.47)
p + U(s)	**87.15 (87.00,87.29)**	**89.29 (89.09,89.51)**	67.73 (67.58,67.88)	**90.99 (90.46,91.59)**

Table 3: AUC mean (min, max) on the biomedical datasets. Abbreviations are as in Table 2, except here the external resource is the UMLS MeSH ontology ('U').'U(s)' is the proposed weight sharing method utilizing ULMS.

structured relations between tokens (e.g., derived from the UMLS) in topic modeling.

Weight-sharing in NNs. Recent work has considered stochastically sharing weights in neural models. Notably, Chen et al. (2015). proposed randomly sharing weights in neural networks. Elsewhere, Han et al. (2015) proposed quantized weight sharing as an intermediate step in their deep compression model. In these works, the primary motivation was *model compression*, whereas here we view the hashing trick as a mechanism to encode domain knowledge.

6 Conclusion

We have proposed a novel method for incorporating prior semantic knowledge into neural models via stochastic weight sharing. We have showed it generally improves text classification performance vs. model variants which do not exploit external resources and vs. an approach based on retrofitting prior to training. In future work, we will investigate generalizing our approach beyond classification, and to inform weight sharing using other varieties and sources of linguistic knowledge.

References

Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. 2010. Sentiwordnet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining. In *LREC*. volume 10, pages 2200–2204.

Olivier Bodenreider. 2004. The unified medical language system (umls): integrating biomedical terminology. *Nucleic acids research* 32(suppl 1):D267–D270.

Peter F Brown, Peter V Desouza, Robert L Mercer, Vincent J Della Pietra, and Jenifer C Lai. 1992. Class-based n-gram models of natural language. *Computational linguistics* 18(4):467–479.

Wenlin Chen, James T Wilson, Stephen Tyree, Kilian Q Weinberger, and Yixin Chen. 2015. Compressing neural networks with the hashing trick. In *ICML*. pages 2285–2294.

Mei Chung, Denish Moorthy, Nira Hadar, Priyanka Salvi, Ramon C Iovin, and Joseph Lau. 2012. *Biomarkers for Assessing and Managing Iron Deficiency Anemia in Late-Stage Chronic Kidney Disease*. AHRQ Comparative Effectiveness Reviews. Agency for Healthcare Research and Quality (US), Rockville (MD).

Aaron M Cohen, William R Hersh, K Peterson, and Po-Yin Yen. 2006. Reducing workload in systematic review preparation using automated citation classification. *Journal of the American Medical Informatics Association* 13(2):206–219.

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(Aug):2493–2537.

Issa J Dahabreh, Denish Moorthy, Jenny L Lamont, Minghua L Chen, David M Kent, and Joseph Lau. 2013. Testing of cyp2c19 variants and platelet reactivity for guiding antiplatelet treatment .

Finale Doshi-Velez, Byron C Wallace, and Ryan Adams. 2015. Graph-sparse lda: A topic model with structured sparsity. In *AAAI Conference on Artificial Intelligence*. pages 2575–2581.

Manaal Faruqui, Jesse Dodge, Sujay K Jauhar, Chris Dyer, Eduard Hovy, and Noah A Smith. 2014. Retrofitting word vectors to semantic lexicons. *arXiv preprint arXiv:1411.4166 .*

Yoav Goldberg. 2016. A primer on neural network models for natural language processing. *Journal of Artificial Intelligence Research* 57:345–420.

Song Han, Huizi Mao, and William J Dally. 2015. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149 .*

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the 10th ACM SIGKDD international conference on Knowledge discovery and data mining*. pages 168–177.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882 .*

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

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM* 38(11):39–41.

SPFGH Moen and Tapio Salakoski2 Sophia Ananiadou. 2013. Distributional semantics resources for biomedical text processing.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proc. of the ACL.*

D.E. Rumelhart, G.E. Hintont, and R.J. Williams. 1986. Learning representations by back-propagating errors. *Nature* 323(6088):533–536.

Qinfeng Shi, James Petterson, Gideon Dror, John Langford, Alex Smola, and SVN Vishwanathan. 2009. Hash kernels for structured data. *Journal of Machine Learning Research* 10(Nov):2615–2637.

T. Terasawa, T. Dvorak, S. Ip, G. Raman, J. Lau, and T. A. Trikalinos. 2009. Charged Particle Radiation Therapy for Cancer: A Systematic Review. *Ann. Intern. Med. .*

Byron C Wallace, Thomas A Trikalinos, Joseph Lau, Carla Brodley, and Christopher H Schmid. 2010. Semi-automated screening of biomedical citations for systematic reviews. *BMC bioinformatics* 11(1):55.

Kilian Weinberger, Anirban Dasgupta, John Langford, Alex Smola, and Josh Attenberg. 2009. Feature hashing for large scale multitask learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*. pages 1113–1120.

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

Chang Xu, Yalong Bai, Jiang Bian, Bin Gao, Gang Wang, Xiaoguang Liu, and Tie-Yan Liu. 2014. Rc-net: A general framework for incorporating knowledge into word representations. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*. ACM, pages 1219–1228.

Dani Yogatama and Noah A Smith. 2014. Linguistic structured sparsity in text categorization. In *Meeting of the Association for Computational Linguistics*. pages 786–796.

Mo Yu and Mark Dredze. 2014. Improving lexical embeddings with semantic knowledge. In *ACL*. pages 545–550.

Zhiguo Yu, Trevor Cohen, Elmer V Bernstam, and Byron C Wallace. 2016. Retrofitting word vectors of mesh terms to improve semantic similarity measures. *Intl. Workshop on Health Text Mining and Information Analysis at EMNLP* pages 43–51.

Matthew D. Zeiler. 2012. Adadelta: An adaptive learning rate method.

Ye Zhang, Iain Marshall, and Byron C Wallace. 2016a. Rationale-augmented convolutional neural networks for text classification. *arXiv preprint arXiv:1605.04469 .*

Ye Zhang, Stephen Roller, and Byron C Wallace. 2016b. MGNC-CNN: A simple approach to exploiting multiple word embeddings for sentence classification pages 1522–1527.

Ye Zhang and Byron C Wallace. 2015. A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification. *arXiv preprint arXiv:1510.03820 .*