Def2Vec: Learning Word Vectors from Definitions

Andrey Kurenkov
Department of Computer Science
Stanford University
Stanford, CA 94305
andreyk@stanford.edu

Tony Duan
Department of Computer Science
Stanford University
Stanford, CA 94305
tonyduan@stanford.edu

Abstract

Pre-trained distributed word representations ("word vectors") have contributed enormously to the advancement of NLP, but are challenged by the presence of out-of-vocabulary (OOV) words – that is, rare words with no corresponding pre-trained vectors. OOV words have been traditionally handled as `<UNK>` tokens, and recent approaches have suggested character-level models to account for morphology. However, for many applications such as translation and summarization, these solutions are not sufficient as they do not convey the semantic meaning of the OOV words. We address this problem with *Def2Vec*, a recurrent neural network that converts from word definitions to word vectors. We quantitatively and qualitatively demonstrate that Def2Vec is able to embed vectors into a space similar to GloVe's pre-trained word embeddings, and demonstrate the utility of Def2Vec in improving the performance of a Neural Machine Translation model when the vocabulary of pre-trained vectors is limited.

1 Introduction

Pre-trained distributed word representations ("word vectors") have contributed enormously to advances in NLP, due to their ability to capture the various semantic meanings of a given word better and more flexibly than other approaches. However, since the standard practice is to pretrain word vectors on a lot of data and with a lot of compute, downstream tasks such as translation or summarization that rely on these vectors have no way of generating new vectors beyond those that are in the pretrained vector vocabulary. The problem of handling words without associated pretrained vectors - out-of-vocabulary (OOV) words - is a serious and as of yet unresolved one. Such OOV words with have been traditionally handled as `<UNK>` tokens and more recently with character-level models [10] [4].

For many applications such as translation and summarization, these existing solutions are not sufficient as they either completely ignore the word itself (as with the `<UNK>` token approach) or capture only information related to its morphology and not its actual semantic meaning (character-level encodings). An intuitive and simple idea is to handle OOV words the same way people do: by looking up a definition (a sentence or set of sentences that convey the semantic meaning of the word) in a dictionary. Since both definitions and word vectors convey the semantic meaning of a given word, it makes intuitive sense that it should be possible to generate vectors directly from definitions.

Our contribution is a deep learning model that does exactly that – *Def2Vec*. We build upon prior work that explores definition-based word vectors by, (1) extending existing models with attention, which we show is crucial for improving performance, and (2) evaluating the utility of Def2Vec vectors in an extrinsic task, Neural Machine Translation. We quantitatively and qualitatively demonstrate that, by leveraging definitions alone, Def2Vec is able to embed words into a semantically meaningful space comparable to that of pre-trained GloVe embeddings. We also demonstrate the utility of Def2Vec in improving the performance of a Neural Machine Translation model when the pre-trained vectors vocabulary is limited.

2 Related Work

The closest to our work is [1], in which the authors leverage dictionary definitions and character-level morphology to construct neural models that embed word vectors. They evaluate performance on three extrinsic tasks: question answering, entailment prediction, and language modeling. We extend their work by introducing an attention mechanism to the model, and by evaluating performance in the neural machine translation.

There have been a number of other attempts toward deriving word vectors from dictionary definitions. Most salient is the neural model by [2] which takes the same approach as we do and shows success at the reverse lookup task, but evaluates performance on translation through a bilingual embedding instead of directly replacing word vectors. Work by [15] leverages dictionary definitions, but implements a skip-gram model based on sampling "positive" and "negative" pairs instead of directly embedding definitions through a recurrent model. Definition-derived word embeddings were combined with language modeling in [11], in which the authors demonstrate success at modeling the definition of a word given its embedding. Other recent attempts to include semantic knowledge into word embeddings include [18], [20], and [14].

More broadly, there have been several recent works addressing the out-of-vocabulary problem. Notably, character-level models [4] have seen much success. For machine translation specifically, a recently proposed approach for handling OOV words is [7], in which the authors combine a word-level neural machine translation model with a character-level model. Character-level models are certainly effective at capturing morphological meaning, but we believe that definition-based word embeddings can capture more nuance for semantically rich words. Other approaches for handling OOV words include pointer-sentinel networks [17] and pointer-sentinel mixture models [3].

3 Approach

3.1 Definition encoder

Our definition encoding model is implemented as a bidirectional GRU with attention (Figure 1). Given an input word, we look up its definition in the WordNet database [9]. We encode each word of the definition through an embedding layer, and input the embeddings into a bidirectional GRU. A weighted average of the outputs of the GRU is taken through a linear attention mechanism, and the result is passed through a fully-connected layer to predict the final output embedding. We define loss as the L_2 distance between our predicted embeddings and the ground-truth GloVe embeddings.

$$\text{loss} = ||\hat{v}_{\text{Def2Vec}} - v_{\text{GloVe}}||_2$$

The overall architecture is inspired by the model proposed in [1], with two changes: (1) the use of a GRU for recurrence instead of an LSTM, and (2) the implementation of an attended mean over GRU outputs instead of a straightforward mean.

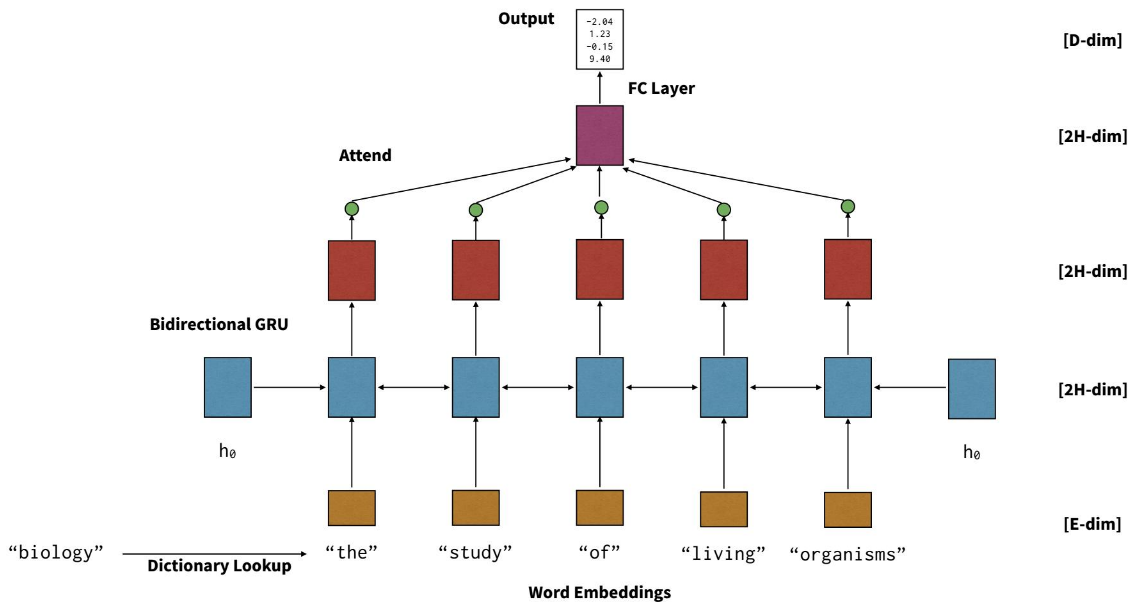


Figure 1: Architecture overview of Def2Vec encoder.

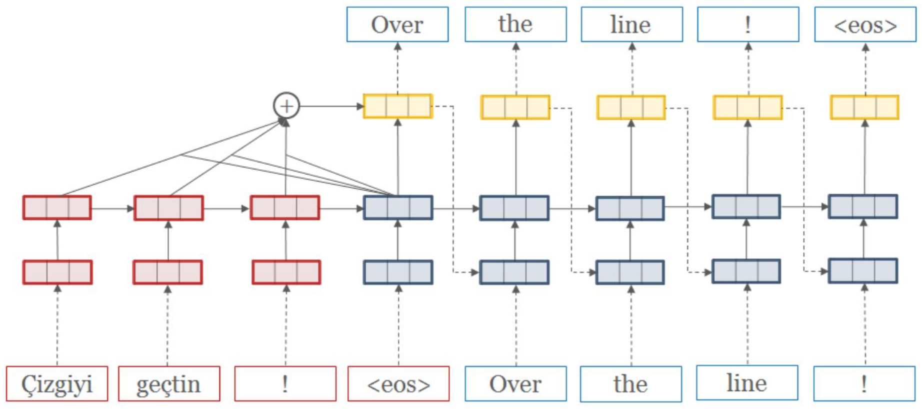


Figure 2: Architecture overview of the OpenNMT model, diagram derived from [6].

3.2 Neural Machine Translation

Our approach for machine translation is a standard Seq2Seq model with attention, implemented through Harvard's open-source OpenNMT project [6]. We use the default plain RNN encoder and decoder with attention and LSTM cells (Figure 2).

4 Experiments

4.1 Data

To train and assess the performance of Def2Vec we leverage datasets for word definitions, word vectors, and machine translation. For definitions, we employ the `nltk` library[1] which uses data from the WordNet database [9]. Because WordNet's number of words is limited at only 150k words, we also made limited use of the `vocabulary` library [2] to access the much larger Glosbe [3] dictionary. Our ground-truth embeddings are the corresponding 100-dimensional GloVe vectors [13]; we use the 400k vocabulary version trained on Wikimedia 2014 and Gigaword 6. Lastly, for the NMT task we make use of the Yandex 1M English-Russian Corpus which has one million aligned English and Russian sentences [19].

4.2 Training

Our GRU employs a hidden state of dimension 150 and has 2 layers, with a dropout probability of 0.1 between each layer. Our word embedding layer is initialized from 100-dimensional GloVe vectors and fine-tuned. Unknown words that occur in the definitions are represented by `<UNK>` tokens and an embedding for this token is trained as well. We implemented our model in PyTorch [12] and trained using the Adam [5] optimizer for 15 epochs with a learning rate of 0.0001 and a batch size of 64.

The full dataset that we train upon is the set of 400K pre-trained GloVe embeddings. For evaluation, we hold out 2034 words in the Stanford Rare Words dataset [8] as a test set. We also randomly sampled 2000 of the remaining words in GloVe to hold out as a validation set, on which we tuned hyper-parameters.

4.3 Intrinsic evaluation

Our first set of metrics are designed to evaluate the quality of our definition encoder model – how well does it reconstruct GloVe vectors given only the corresponding definitions? We evaluated on the L_2 loss in our test set constructed from the Stanford Rare Words dataset and performed an ablation analysis of architectural choices we made (Table 1). The ablation experiments compare performance of the "standard" model we have described thus far against versions with (1) bidirection removed, (2) attention removed, and (3) the GRU layer removed (limiting the model to attention only). Each experiment was run with the same set of hyper-parameters described.

We note that the average distance between any two random vectors in GloVe is 2.66, and a baseline approach of straightforwardly averaging the word embeddings comprising a given definition achieves an average distance of 4.00. All neural models we assessed yielded significantly better performance, with our "standard"

[1] `https://www.nltk.org/`

[2] `https://vocabulary.readthedocs.io/en/latest/introduction.html`

[3] `https://glosbe.com/`

Ablation	**Standard**	No bidirection	No GRU	No attention	Baseline-GloVe	Baseline-Avg
L_2 Distance	**0.180**	0.184	0.184	0.186	2.66	4.00

Table 1: Mean distances for Def2Vec and GloVe vectors for words in Stanford Rare Words Dataset [8].

model outperforming all others at a loss of 0.180. Surprisingly, the version of our model without a GRU component was able to perform almost as well as the full model. In fact, ablating the attention mechanism resulted in comparatively worse performance than ablating the GRU component, suggesting that attention is the most critical choice we made in model architecture.

Qualitatively, we verify through t-SNE visualizations [16] of the test set embedding space that our predicted Def2Vec embeddings are able to accurately reconstruct the corresponding ground-truth GloVe vectors for most words (Figure 4). Moreover, when we query Def2Vec with given definitions, the ground-truth vector is frequently among the nearest GloVe vectors in the embedded space (Figure 3). Lastly, we confirm that the attention mechanism is intuitively reasonable, as it weighs the most relevant words of a given definition most significantly.

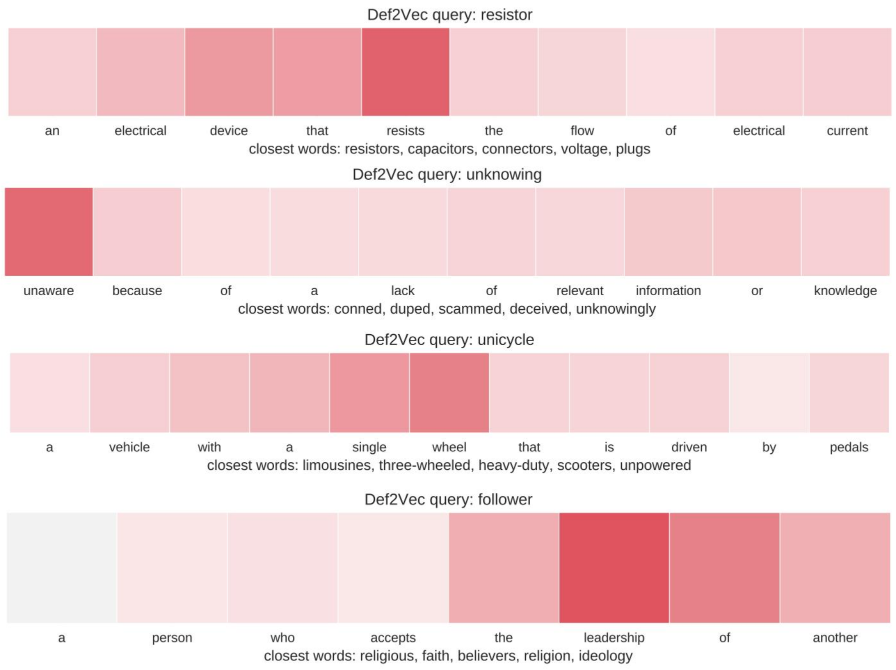
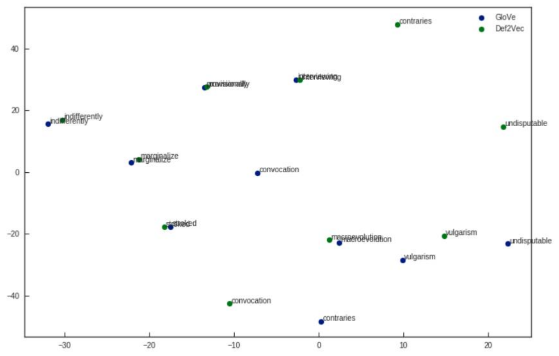


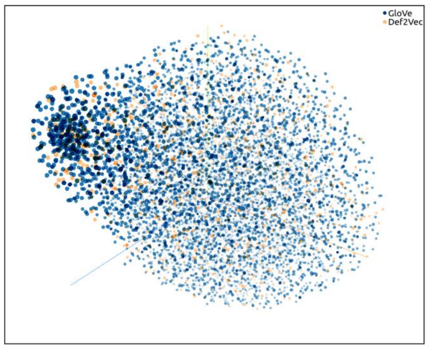
Figure 3: Qualitative depiction of attention mechanism for queries on our test set. We find that Def2Vec is often able to successfully reconstruct the target vector from given definitions, with intuitive attention scores.

4.4 Extrinsic Evaluation

Since the intrinsic evaluation only shows how well our model replicates GloVe vectors, we also do additional extrinsic evaluation to verify how useful the Def2Vec vectors are for downstream tasks. We train our



(a) 2D t-SNE.



(b) 3D t-SNE.

Figure 4: In (a), we examine the embeddings of 10 words in the test set and find that the reconstructed Def2Vec embeddings are very close to the ground-truth GloVe embeddings. In (b), 5k GloVe and 5k Def2Vec embeddings from the training set are plotted to show their distributions are similar but not the same, showing that we are not overfitting.

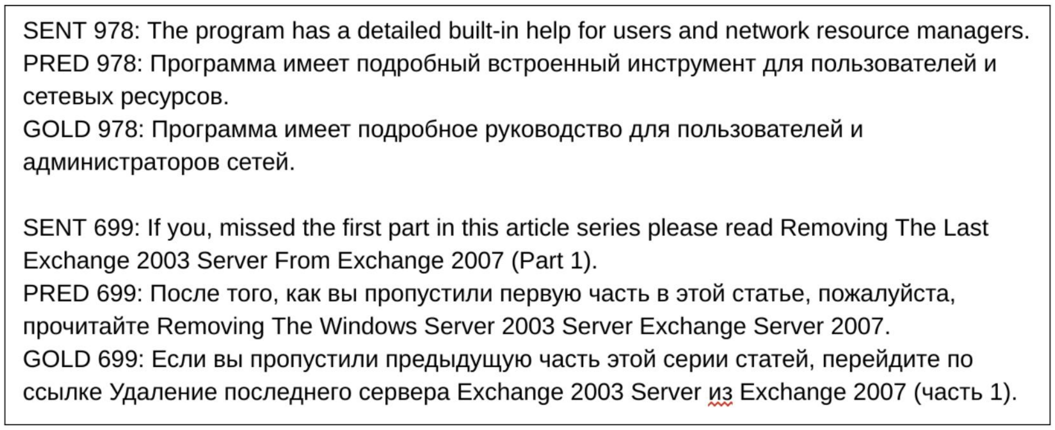
translation model derived from OpenNMT with 2 layers, a hidden state of size 200, and frozen encoder vectors. We run two sets of experiments: (1) a set with only a subset of the most frequent GloVe vectors, and (2) a set with a combination of the most frequent GloVe vectors and our vectors embedded by Def2Vec. We implemented our model in PyTorch [12] and trained using the Adam [5] optimizer for 12 epochs with a learning rate of 0.0001 and a batch size of 64.

Quantitative results are presented in (Table 2). We find that if we train the NMT model with only 1k of the most frequent words in GloVe, performance is noticably inferior. However, if we augment these words with the vectors imputed by Def2Vec, performance improves considerably – in fact, it is almost on par with perplexities reported using the top 10k most frequent words in GloVe. We make the same observation when we train the NMT model with 10k of the most frequent words in GloVe – the addition of Word2Vec embeddings improves the BLEU score. However, when we reach a vocabulary of 50k GloVe vectors as a baseline, the addition of our Def2Vec embeddings actually causes a slight decrease in performance of the model. This indicates that there are diminishing marginal returns to the addition of Def2Vec embeddings into the NMT system; it is most helpful when the original vocabulary is small and as we look up definitions for rarer words, the overall boost in performance is less significant.

We also present qualitative results in (Figure 5). Qualitatively, our model is able to generate reasonably accurate translations much of the time. We found it especially surprising given that we had a hidden size of only 200 with little tuning, and trained on a relatively small dataset.

Vocabulary	400k	50k + Def2Vec	10k + Def2Vec	1k + Def2Vec	50k	10k	1k
Perplexities	38.15	39.19	39.27	45.23	39.05	41.82	59.64
BLEU	5.55	5.09	4.61	–	5.15	4.44	2.82

Table 2: Quantitative evaluation of NMT models evaluated on test set sentences. The vocabulary labels indicate whether only a subset of the most common GloVe vectors were used, or whether a combination of that subset was combined with Def2Vec vectors. The missing entry is due to the fact that we ran out of time to generate the last 1k + Def2Vec BLEU score.



SENT 978: The program has a detailed built-in help for users and network resource managers.
PRED 978: Программа имеет подробный встроенный инструмент для пользователей и сетевых ресурсов.
GOLD 978: Программа имеет подробное руководство для пользователей и администраторов сетей.

SENT 699: If you, missed the first part in this article series please read Removing The Last Exchange 2003 Server From Exchange 2007 (Part 1).
PRED 699: После того, как вы пропустили первую часть в этой статье, пожалуйста, прочитайте Removing The Windows Server 2003 Server Exchange Server 2007.
GOLD 699: Если вы пропустили предыдущую часть этой серии статей, перейдите по ссылке Удаление последнего сервера Exchange 2003 Server из Exchange 2007 (часть 1).

Figure 5: Examples of NMT outputs on test set with 50k + Def2Vec model. Sentece 978 is translated well, whereas the more complicated sentence 699 is translated less accurately.

5 Conclusion

Motivated as an intuitive approach for handling out-of-vocabulary words, we have shown that our Def2Vec model can effectively embed word vectors given only the definition of a query word. Our experiments show that the approach is intrinsically successful at reconstructing word vectors, and can be used as a reverse lookup tool. Moreover, we verify that the resulting word embeddings have utility in the extrinsic neural machine translation task, especially when the original vocabulary is relatively small. We believe that dictionary-derived word embeddings can be especially useful for highly technical words, slang, and other languages – any instance in which a rich dataset of definitions is available.

There are a few important limitations in our work and possible extensions toward further work. First, we currently handle out-of-vocabulary tokens in given definitions with an `<UNK>` token – an inelegant approach for which we'd like to explore a recursive approach in the future, looking up dictionary definitions for unknown entries in a definition. Second, we currently include definitions in our training process that may reference semantically identical words to the target – for example, "transparency: the condition of being transparent". While we decided that it was reasonable to keep these definitions in for now, we would want to explore the impact of removing such definitions in the future. Finally, we are interested in exploring datasets beyond definitions, including Wikipedia articles, parse trees, and dependency trees.

References

[1] Dzmitry Bahdanau, Tom Bosc, Stanislaw Jastrzebski, Edward Grefenstette, Pascal Vincent, and Yoshua Bengio. Learning to compute word embeddings on the fly. *CoRR*, abs/1706.00286, 2017.

[2] Felix Hill, Kyunghyun Cho, Anna Korhonen, and Yoshua Bengio. Learning to understand phrases by embedding the dictionary. *CoRR*, abs/1504.00548, 2015.

[3] Sébastien Jean, Kyunghyun Cho, Roland Memisevic, and Yoshua Bengio. On using very large target vocabulary for neural machine translation. *CoRR*, abs/1412.2007, 2014.

[4] Yoon Kim, Yacine Jernite, David Sontag, and Alexander M. Rush. Character-aware neural language models. *CoRR*, abs/1508.06615, 2015.

[5] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

[6] Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander M. Rush. Opennmt: Open-source toolkit for neural machine translation. In *Proc. ACL*, 2017.

[7] Minh-Thang Luong and Christopher D. Manning. Achieving open vocabulary neural machine translation with hybrid word-character models. *CoRR*, abs/1604.00788, 2016.

[8] Minh-Thang Luong, Richard Socher, and Christopher D. Manning. Better word representations with recursive neural networks for morphology. In *CoNLL*, Sofia, Bulgaria, 2013.

[9] George A. Miller. Wordnet: A lexical database for english. *Commun. ACM*, 38(11):39–41, November 1995.

[10] Yasumasa Miyamoto and Kyunghyun Cho. Gated word-character recurrent language model. *CoRR*, abs/1606.01700, 2016.

[11] Thanapon Noraset, Chen Liang, Larry Birnbaum, and Doug Downey. Definition modeling: Learning to define word embeddings in natural language. *CoRR*, abs/1612.00394, 2016.

[12] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

[13] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014.

[14] Sascha Rothe and Hinrich Schütze. Autoextend: Extending word embeddings to embeddings for synsets and lexemes. *CoRR*, abs/1507.01127, 2015.

[15] Julien Tissier, Christophe Gravier, and Amaury Habrard. Dict2vec : Learning Word Embeddings using Lexical Dictionaries. In *Conference on Empirical Methods in Natural Language Processing (EMNLP 2017)*, pages 254–263, Copenhague, Denmark, September 2017.

[16] L.J.P. van der Maaten and G.E. Hinton. Visualizing high-dimensional data using t-sne. 2008.

[17] O. Vinyals, M. Fortunato, and N. Jaitly. Pointer Networks. *ArXiv e-prints*, June 2015.

[18] Chang Xu, Yalong Bai, Jiang Bian, Bin Gao, Gang Wang, Xiaoguang Liu, and Tie-Yan Liu. 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*, CIKM '14, pages 1219–1228, New York, NY, USA, 2014. ACM.

[19] Yandex. Yandex 1m en-ru dataset, 2018.

[20] Chunting Zhou, Chonglin Sun, Zhiyuan Liu, and Francis C. M. Lau. Category enhanced word embedding. *CoRR*, abs/1511.08629, 2015.