INTERSPEECH 2020
October 25–29, 2020, Shanghai, China

Improved Learning of Word Embeddings with Word Definitions and Semantic Injection

Yichi Zhang[1], *Yinpei Dai*[1], *Zhijian Ou*[1†], *Huixin Wang*[2], *Junlan Feng*[2]

[1]Speech Processing and Machine Intelligence (SPMI) Lab, Tsinghua University, China
[2]China Mobile Research Institute

{zhangyic17, dyp16}@mails.tsinghua.edu.cn, ozj@tsinghua.edu.cn

Abstract

Recently, two categories of linguistic knowledge sources, word definitions from monolingual dictionaries and linguistic relations (e.g. synonymy and antonymy), have been leveraged separately to improve the traditional co-occurrence based methods for learning word embeddings. In this paper, we investigate to leverage these two kinds of resources together. Specifically, we propose a new method for word embedding specialization, named Definition Autoencoder with Semantic Injection (DASI). In our experiments[1], DASI outperforms its single-knowledge-source counterparts on two semantic similarity benchmarks, and the improvements are further justified on a downstream task of dialog state tracking. We also show that DASI is superior over simple combinations of existing methods in incorporating the two knowledge sources.

Index Terms: word embedding specialization, word definitions, semantic injection, dialog state tracking

1. Introduction

Distributed representations of words, also known as word embeddings, have been successfully used in many natural language processing (NLP) tasks [1, 2]. More recently, contextualized embeddings like ELMO [3] or BERT [4] have been proposed, which are dynamically created based on a whole sentence. Though with improved performance in some NLP tasks, they suffer from large storage/time complexity in inference and may not be suitable to word-level tasks such as word similarity evaluations. Under these considerations, word embeddings still have their own significance in applications.

Traditional methods of learning word embeddings over unlabeled textual corpora are mainly based on the distributional hypothesis [5], which states that words occuring in similar contexts tend to be semantically close. These co-occurrence based methods are good at capturing relatedness between words, but usually lack in capturing similarity and distinguishing similarity from relatedness [6]. For example, "cup" and "coffee" often appear together - they are related but not similar. This drawback can hurt the performance of downstream language understanding tasks such as dialog state tracking [7], which aims at tracking users' preference over semantic slots expressed in user utterances, as shown in Table 1.

This weakness of the co-occurrence based methods can be alleviated by introducing linguistic supervision in the learning process. Two classes of external semantic knowledge sources have been leveraged separately. First, *word definitions* in monolingual dictionaries, which contain semantic descriptions about words, are used to enhance the learning of word embeddings, especially for capturing similarity between words [8, 9, 10]. Second, distributional word vectors can be refined by injecting semantic relations (e.g. synonymy and antonymy) from lexical resources such as WordNet [11] and Paraphrase Database [12], as shown in [13, 14, 15, 16]. We term this process *semantic injection*.

Restaurant price range	
Slot value	Synonyms
cheap	cheaper, inexpensive, bargain, ...
moderate	mid-price, affordable, medium, ...
expensive	costly, pricy, dear, ...

Table 1: *Examples of different expressions of users' preference over restaurant price ranges. Language understanding models need to classify synonymous expressions into the correct class (slot value) and distinguish antonymous ones such as "cheap" and "expensive".*

However, for learning with dictionary definitions or with semantic injection, either alone has its own limitation. For learning with dictionary definitions alone, semantic relations between words are not explicitly enforced in learning word embeddings, since a word's synonyms and antonyms do not always appeared in its definition. For learning with semantic injection alone, although semantic relations are directly injected, the descriptive information contained in dictionary definitions are not exploited. For example, the phrase "charging low prices" in the definition of word "cheap" clearly carries useful information for capturing similarity between words.

In this paper, we propose a new method for word embedding post-processing that specializes in similarity relation, named Definition Autoencoder with Semantic Injection (DASI), which leverages both knowledge sources. The connection and comparison of DASI with existing methods are detailed in Section 2. Two state-of-the-art specialization methods, the ATTRACT-REPEL (A-R) [16] and Consistency Penalized AutoEncoder (CPAE) [10], are used as the baselines of semantic injection and definition modeling respectively. It is found in our experiments that DASI outperforms its single-knowledge-source counterparts in three different vector spaces on two word similarity benchmarks: SimLex-999 [6] and SimVerb-3500 [17]. Furthermore, DASI improves the downstream dialog state tracking (DST) performance over two different DST models.

2. Related Work

Table 2 shows a brief review of existing methods for learning word embeddings, depending on the used external semantic knowledge sources. To the best of our knowledge, this paper

This work is supported by NSFC 61976122, Ministry of Education and China Mobile joint funding MCM20170301. † Corresponding author.

[1]The code is available at https://github.com/thu-spmi/DASI.

http://dx.doi.org/10.21437/Interspeech.2020-1702

External Resource		Semantic Injection	
		×	✓
Word Definitions	×	word2vec [19] GloVe [20] fastText [21]	retrofitting [13] PARAGRAM [14] counter-fitting [15] ATTRACT-REPEL [16]
	✓	Lexicographic [8] dict2vec [9] CPAE [10]	DASI (ours)

Table 2: *Categorization of different word embedding learning methods according to the use of external linguistic resources.*

represents the first exploration in incorporating both word definitions and semantic injection for word representation learning.

2.1. Learning with Word Definitions

There are two main ways to learn with word definitions. One is to use word definitions as an additional co-occurrence context [8, 9]. The other is to learn definition embeddings which can be used to generate or reconstruct the original word definitions [18, 10]. Our approach is similar to the latter in the spirit of using definition reconstruction to incorporate dictionary knowledge. Specifically, inspired from CPAE [10], DASI consists of encoding the sequence of words in a definition into a vector and trying to reconstruct/decode the definition. A consistency penalty is used to enforce that the input word embeddings it uses as inputs and the definition embeddings it produces as outputs are close to each other, as a result of the inherent recursivity of dictionaries. However, there are two important differences between DASI and CPAE, as will be detailed in Section 3.1 with experimental comparison in Section 5.

2.2. Learning with Semantic Injection

Semantic relations in semantic lexicons such as WordNet [11] and Paraphrase Database (PPDB) [12] have been shown to be useful in fine-tuning and specializing pre-trained word embeddings to better capture word similarity [13, 14, 15, 16]. Specifically, retrofitting [13] brings the vectors of semantically similar words close together. PARAGRAM [14] injects paraphrasing constraints from PPDB to the original skip-gram objective function [19] to fine-tune the word vectors. Counter-fitting [15] and ATTRACT-REPEL [16] leverage both synonymy and antonymy constraints as semantic injection sources, and use a hinge loss to pull synonymy pairs closer and push antonymy pairs away to reach the pre-defined similarity margin.

In DASI, we also use synonymy and antonymy constraints but with a different objective function so that we can naturally combine word definition modeling and semantic injection. In particular, synonyms are attracted by increasing the conditional likelihood of its synonyms for a given word, and antonyms are repelled in the opposite way (decreasing the conditional likelihood). This difference is detailed in Section 3.2 with experimental comparison in Section 5.

3. Definition Autoencoder with Semantic Injection

Denote by e_w the target embedding for word w, which is initially in a pre-trained word vector space (e.g. word2vec [19]). Suppose that we have access to a dictionary consisting of word definitions and a semantic lexicon consisting of synonyms and antonyms pairs.

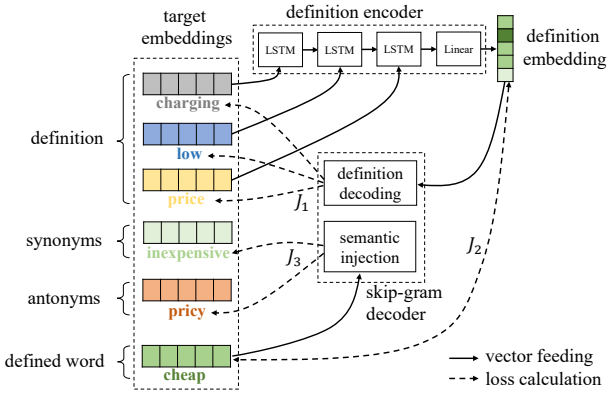

Figure 1: *Overview of the DASI model.*

- Denote by $\mathcal{D}$ the set of the words which have definitions. Let $D_w = \{D_{w,t}, t = 1, \cdots, |D_w|\}$ be the concatenated sequence of words from all the definitions for a word $w \in \mathcal{D}$, since a word may have multiple senses in the dictionary[2]. Denote by $\mathcal{E} \triangleq \{w' | w' \in D_w, w \in \mathcal{D}\}$ the set of words that are used to define words in $\mathcal{D}$.
- Denote by $\mathcal{M}$ the set of the words which have synonyms and/or antonyms. Let S_w and A_w be the set of synonyms and antonyms of word $w \in \mathcal{M}$ respectively.

Let $e_{\mathcal{E}}$, $e_{\mathcal{E}\cup\mathcal{D}}$ and $e_{\mathcal{M}}$ denote the embeddings of words in the corresponding set of words respectively. Let $\mathcal{V} \triangleq \mathcal{D} \cup \mathcal{E} \cup \mathcal{M}$ be the total vocabulary. Then we can improve $\{e_w | w \in \mathcal{V}\}$, by leveraging both the word definitions and the synonyms and antonyms, via DASI as shown in Fig.1.

3.1. Definition Modeling

Word definitions are incorporated by using an autoencoder model inspired from CAPE. For $w \in \mathcal{D}$, an LSTM based encoder runs over the word sequence D_w , where the words in D_w are represented by their target embeddings we want to improve. Then we apply a linear transformation to the last hidden state of LSTM to obtain the definition embedding d_w that is of the same dimension as e_w :

$$d_w = f_\theta(D_w) = W \cdot LSTM(D_w) + b$$

where θ denotes the encoder parameters, consisting of the LSTM parameters, and $\{W, b\}$. A skip-gram based decoder, which treats D_w as a bag-of-words, is introduced to define a reconstruction loss J_1:

$$J_1(e_{\mathcal{E}}, \theta, \phi) = -\sum_{w\in\mathcal{D}} \sum_{t=1}^{|D_w|} \log q_\phi(D_{w,t}|d_w)$$

Here $q_\phi(\cdot|\cdot)$ denotes the skip-gram probability, which can be generally defined for word $u \in \mathcal{V}$ appearing in the context of word $w \in \mathcal{D}$:

$$\log q_\phi(u|d_w) = \log \frac{\exp(W_u^{\mathrm{T}} d_w + b_u)}{\sum_{k\in\mathcal{V}} \exp(W_k^{\mathrm{T}} d_w + b_k)} \quad (1)$$

where W_k and b_k denote the weight and bias for word $k \in \mathcal{V}$, and ϕ denotes the decoder parameter $\{W_k, b_k | k \in \mathcal{V}\}$.

[2]With abuse of notation, D_w may also represent the set of words in the sequence D_w in this paper. $|D_w|$ denotes the length/size.

To incorporate the definition information compressed in d_w into e_w, we borrow the consistency penalty in [10], which is to make e_w and d_w be close to each other :

$$J_2(e_{\mathcal{D}\cup\mathcal{E}}, \theta) = \sum_{w\in\mathcal{D}} dist(e_w, d_w)$$

where $dist$ is a distance measurement and we choose the Euclidean distance in our experiment.

DASI draws inspiration from CPAE, but differs from CPAE in two important ways. First, DASI aims to improve the target embeddings, by feeding them into the LSTM and enforcing reconstruction and consistency to fine-tune the target embeddings. In contrast, CPAE adopts the definition embedding from the LSTM output as the target embedding, after training the LSTM from scratch. Fine-tuning is much faster than learning from scratch, and has the advantage of only making necessary changes to pre-trained vectors. Section 5 further provides experimental validation. Second, apart from used in definition decoding, we propose to use the skip-gram model to do semantic injection in DASI, which naturally integrates two knowledge sources and is found to be superior to other simple combinations of existing methods as shown in our experiments.

3.2. Semantic Injection

We inject the semantic relation constraint by increasing the conditional log-probability of synonyms and decreasing the conditional log-probability of antonyms for $w \in \mathcal{M}$ as follows:

$$J_3(e_{\mathcal{M}}, \phi) = -\sum_{w\in\mathcal{M}}\left(\sum_{s\in S_w}\log q_\phi(s|e_w) - \sum_{a\in A_w}\log q_\phi(a|e_w)\right)$$

Here we propose to reuse the skip-gram model $q_\phi(\cdot|\cdot)$ in Eq. (1) to calculate the probability of synonyms/antonyms appearing in the context of word w, since e_w and d_w are of the same dimension. The semantic injection penalty pulls the defined word embedding e_w close to the embedding of synonyms e_s where $s \in S_w$, and pushes e_w away from the embedding of antonyms e_a where $a \in A_w$. The motivation of reusing the skip-gram decoder for semantic injection is to incorporate word definitions and semantic relations in a consistent manner.

The overall objective function is given by the weighted sum of the three losses described above:

$$J(e_{\mathcal{V}}, \theta, \phi) = J_1(e_{\mathcal{E}}, \theta, \phi) + \alpha J_2(e_{\mathcal{D}\cup\mathcal{E}}, \theta) + \beta J_3(e_{\mathcal{M}}, \phi)$$

where α and β are hyperparamters to control the balance of each loss.

4. Experiments

We compare the proposed method with two state-of-the-art models, CPAE [10] and ATTRACT-REPEL (shortened as A-R) [16], using either word definitions or semantic relations. We conduct experiments for both intrinsic and downstream evaluations. Details of training and evaluation settings are described as follows.

4.1. Training Settings

We use word definitions and semantic relations in WordNet [11]. WordNet is a large human-constructed semantic lexicon for English words. It groups words into sets of synonyms called *synsets*, provides their definitions, and records the various semantic relations between synsets. There are 117,597 synsets and 207,016 semantic relation pairs. We use the definitions in WordNet as the resource of dictionary definitions, and the annotated synonymy and antonymy as the resource of semantic injection to train our models. For words that have more than one sense (included in more than one synsets in WordNet), their definitions are the concatenation of definitions in all senses and the synonym set and antonym set are also the union of each in all senses respectively.

Pre-trained Embedding	Specialization Method	Similarity Scores SV-dev	SV-test	SL
word2vec	-	39.20	35.78	44.09
	A-R	59.60	54.61	65.87
	CPAE	44.59	41.73	44.63
	DASI	**63.77**	**60.75**	**67.59**
GloVe	-	26.92	22.20	36.89
	A-R	51.46	44.48	59.88
	CPAE	35.13	28.68	40.20
	DASI	**59.56**	**54.88**	**64.03**
Paragram-SL999	-	52.80	54.21	68.41
	A-R	60.57	60.02	72.99
	CPAE	60.63	58.35	68.12
	DASI	**66.81**	**65.54**	**73.95**

Table 3: *Results of different post-processing methods for three kinds of embeddings on SimVerb-3500 (SV) and SimLex-999 (SL). Best results are in bold.*

We use three kinds of pre-trained word embeddings in our experiments. The word2vec[3] [19] and GloVe [4] [20] which are distributed word vectors trained on large text corpus, and Paragram-SL999 [5] [14] which is a specialization of word2vec using the Paraphrase Database [12]. All of the embeddings are 300 dimensions.

The implementations of CPAE and ATTREP-REPEL are based on the open-source code of the corresponding papers. We use a batch size of 256 and employ the Adam optimizer for all the methods. For DASI, we set $\alpha = 25, \beta = 1.0$ and use a learning rate of 0.001. We set $\lambda = 32$ and a learning rate of 0.001 for CPAE and $\delta_{sim} = 1.0$, $\delta_{ant} = 0.0$, $\lambda_{reg} = 10^{-9}$ and a learning rate of 0.05 for ATTRACT-REPEL. All the hyperparameters are chosen by grid search. We use the standard development set of SimVerb-3500 [17] for validation check, and training early stops when no improvement is observed within 100 updating steps. The model with the best validation score is used for intrinsic and downstream evaluations. The vocabulary size is 20,000 in our experiments, which includes almost all the words in the word similarity benchmarks and the downstream dialog dataset.

4.2. Evaluation Settings

We conduct the intrinsic evaluation on two word similarity datasets: SimLex-999 [6] and SimVerb-3500 [17], which contain human annotated similarity ratings for 999 and 3500 word pairs respectively. For embedding evaluation, we compute the cosine similarity for each word vector pair and measure the *Spearman correlation* ρ between predicted score ranking and ground truth ranking.

We expect that the improvement of word similarity capturing can benefit downstream tasks such as language understanding, and we choose the dialog state tracking (DST) task for evaluation. DST aims to capture user goals, which are expressed by slot-value pairs such as *food=India* or *area=North*, given user utterances. The performance are measured by *joint goal accuracy* and *request accuracy*, which represent the proportion of turns with all the slot-value pairs correctly classified and all the requests correctly answered respectively.

[3] https://code.google.com/archive/p/word2vec/
[4] https://nlp.stanford.edu/projects/glove/
[5] https://cogcomp.org/page/resource_view/106

Model	Embedding	Joint goal	Request
NBT [22]	Paragram-SL999	81.94	93.52
	+ A-R	83.08	**93.88**
	+ CPAE	81.80	93.76
	+ DASI	**84.32**	93.34
GLAD [23]	Paragram-SL999	88.71	97.11
	+ A-R	86.41	96.94
	+ CPAE	88.96	97.15
	+ DASI	**89.78**	**97.40**
BERT-DST [24]		87.7	-
COMER [25]		88.6	-
DADST [26]		89.9	-

Table 4: *Joint goal accuracy and request accuracy of two models with different input embeddings on WOZ 2.0. Paragram-SL999 is chosen as the baseline embedding and three post-processing methods are compared. Best results are in bold.*

Method	Epoch	SV-dev	SV-test	SL
CPAE (input)	**24.5**	44.59	**44.63**	**41.73**
CPAE (definition)	34.6	**45.56**	41.72	38.76
DASI (input)	**33.2**	**63.77**	**60.75**	**67.59**
DASI (definition)	47.1	57.35	54.17	54.96

Table 5: *Comparison of applying input embeddings versus definition embeddings in evaluating definition autoencoder based models. Epoch denotes the average epoch number when training early stops. Best results are in bold.*

The evaluation is based on WOZ 2.0 dataset [22], which consists of 600/200/400 dialogs in training/development/testing sets respectively in the restaurant domain. The system is required to track user's preference on price range, restaurant location and food type, where many variants (southern for south, cheaper for cheap etc.) and synonyms (costly for expensive etc.) are used in user expressions. We use two DST models, the Neural Belief Tracker (NBT) [22] and the Global-Locally Self-Attentive Dialogue State Tracker (GLAD) [23]. Both of them use fixed word embeddings as input which allows a fair comparison among different input embeddings. We train NBT and GLAD following their default settings.

5. Results and Analysis

We report the Spearman's correlation coefficient $\rho \times 100$ on SimVerb-3500 and SimLex-999 datasets as metric of word similarity capturing. A-R in the tables denotes the shortcut of ATTRACT-REPEL. All the scores in each table are the average of 5 runs with different random seeds. Results are organized to show three conclusions:

DASI outperforms its single-knowledge-source counterparts under various cases. In Table 3, it can be seen that both pre-trained distributed embeddings (word2vec and GloVe) and specialized embeddings (Paragram-SL999) are improved through injecting external linguistic knowledge. We find that DASI significantly outperforms CPAE and ATTRACT-REPEL in all cases, which shows that word definitions and semantic relations can work together and make a further improvement of embedding qualities. For downstream evaluation, we only report the results of Paragram-SL999 with different post-processing methods since they are better than results of word2vec and GloVe. As shown in Table 4, the significant increase of joint goal accuracy indicates that better capturing of word similarities benefits the tracking of user goals. Compared to state-of-the-art BERT-based models where the *BERTbase* [4] model (110M parameters) is used to produce contextualized embeddings, GLAD with Paragram-SL999 + DASI obtains a close joint goal accuracy with only 17M parameters in total. The consistent improvement of DASI over multiple pre-trained embeddings and under different DST models suggest that DASI generalizes well.

Method	SV-dev	SV-test	SL
original	39.20	35.78	44.09
^{a}A-R $\rightarrow$ CPAE	59.31	54.45	65.74
^{b}CPAE $\rightarrow$ A-R	59.67	56.53	64.17
^{c}CPAE + A-R loss	43.01	38.90	47.14
^{d}DASI (w/o weight tying)	62.73	**60.79**	66.55
DASI	**63.77**^{abcd}	60.75^{abc}	**67.59**^{abcd}

Table 6: *Comparison of different methods for fusing word definitions and semantic relations. A-R→CPAE denotes post-processing by ATTRECT-REPEL first and CPAE later, while A-R→CPAE the inverse. CPAE+A-R loss denotes the model that replaces the J_3 loss in DASI with the pair-wise hinge loss in ATTRACT-REPEL. DASI without weight tying uses independent definition decoder and semantic injection model. Superscripts in the last line denote statistical significance ($p<0.01$) over the four baselines a, b, c and d.*

The effect of adopting different embeddings in incorporating definition knowledge shows the superiority of DASI over CPAE. Table 5 shows that using the input embeddings (as proposed in DASI) rather than the output definition embeddings from the definition autoencoder (as proposed in CPAE) has better similarity scores and faster convergence speed. This indicates that fine-tuning on the original pre-trained vectors is more efficient than learning new vectors by adjusting the LSTM encoder parameters from scratch, since the model can make only necessary changes to the pre-trained vectors.

DASI outperforms simple combinations of existing methods in fusing both sources of external knowledge. We compare DASI with several baseline models incorporating knowledge from both word definitions and semantic relations. As shown in Table 6, simultaneously utilizing multiple external knowledge resources within a single training process outperforms successive utilizations. For DASI, using the skip-gram model obtains a significant better result than directly adding the loss of ATTRACT-REPEL to CPAE, presumably because of the consistent manner in DASI as discussed in Section 3.2. Sharing the parameters in the definition decoder and semantic injection model for DASI results in better similarity scores and a more compact model.

6. Conclusion and Future Work

This work represents a first exploration of incorporating two kinds of external linguistic knowledge resources, the word definitions and semantic relations together to do specialization of word vectors. We develop a new DASI method, which shows significant improvement over its single-knowledge-source counterparts on both intrinsic and downstream evaluations, across various pre-trained embeddings and DST models. We also analyze and compare different fusion approaches to show that DASI is a better solution for knowledge fusion.

There are some interesting future works. First, since only the words contained in dictionary or with semantic relations are specialized in DASI, how to leverage recent progress in learning a global specialization function [27, 28] to overcome this limitation is interesting. Second, while DASI makes use of two external linguistic knowledge resources, other resources such as the example sentences in dictionaries which might be helpful as well are worthwhile to exploration.

7. References

[1] J. Turian, L. Ratinov, and Y. Bengio, "Word representations: a simple and general method for semi-supervised learning," in *Proceedings of the 48th annual meeting of the association for computational linguistics*. Association for Computational Linguistics, 2010, pp. 384–394.

[2] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural language processing (almost) from scratch," *Journal of machine learning research*, vol. 12, no. Aug, pp. 2493–2537, 2011.

[3] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," *arXiv preprint arXiv:1802.05365*, 2018.

[4] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 2019, pp. 4171–4186.

[5] Z. S. Harris, "Distributional structure," *Word*, vol. 10, no. 2-3, pp. 146–162, 1954.

[6] F. Hill, R. Reichart, and A. Korhonen, "Simlex-999: Evaluating semantic models with (genuine) similarity estimation," *Computational Linguistics*, vol. 41, no. 4, pp. 665–695, 2015.

[7] J. Williams, A. Raux, D. Ramachandran, and A. Black, "The dialog state tracking challenge," in *Proceedings of the SIGDIAL 2013 Conference*, 2013, pp. 404–413.

[8] T. Wang, A. Mohamed, and G. Hirst, "Learning lexical embeddings with syntactic and lexicographic knowledge," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, vol. 2, 2015, pp. 458–463.

[9] J. Tissier, C. Gravier, and A. Habrard, "Dict2vec: Learning word embeddings using lexical dictionaries," in *Conference on Empirical Methods in Natural Language Processing (EMNLP 2017)*, 2017, pp. 254–263.

[10] T. Bosc and P. Vincent, "Auto-encoding dictionary definitions into consistent word embeddings," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018, pp. 1522–1532.

[11] G. A. Miller, "Wordnet: a lexical database for english," *Communications of the ACM*, vol. 38, no. 11, pp. 39–41, 1995.

[12] J. Ganitkevitch, B. Van Durme, and C. Callison-Burch, "Ppdb: The paraphrase database," in *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2013, pp. 758–764.

[13] M. Faruqui, J. Dodge, S. K. Jauhar, C. Dyer, E. Hovy, and N. A. Smith, "Retrofitting word vectors to semantic lexicons," in *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2015, pp. 1606–1615.

[14] J. Wieting, M. Bansal, K. Gimpel, and K. Livescu, "From paraphrase database to compositional paraphrase model and back," *Transactions of the Association for Computational Linguistics*, vol. 3, pp. 345–358, 2015.

[15] N. Mrkšic, D. OSéaghdha, B. Thomson, M. Gašic, L. Rojas-Barahona, P.-H. Su, D. Vandyke, T.-H. Wen, and S. Young, "Counter-fitting word vectors to linguistic constraints," in *Proceedings of NAACL-HLT*, 2016, pp. 142–148.

[16] N. Mrkšić, I. Vulić, D. Ó. Séaghdha, I. Leviant, R. Reichart, M. Gašić, A. Korhonen, and S. Young, "Semantic specialization of distributional word vector spaces using monolingual and cross-lingual constraints," *Transactions of the Association for Computational Linguistics*, vol. 5, pp. 309–324, 2017.

[17] D. Gerz, I. Vulić, F. Hill, R. Reichart, and A. Korhonen, "Simverb-3500: A large-scale evaluation set of verb similarity," in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 2016, pp. 2173–2182.

[18] T. Noraset, C. Liang, L. Birnbaum, and D. Downey, "Definition modeling: Learning to define word embeddings in natural language," in *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[19] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[20] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.

[21] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *TACL*, vol. 5, pp. 135–146, 2017. [Online]. Available: https://transacl.org/ojs/index.php/tacl/article/view/999

[22] N. Mrkšić, D. Ó. Séaghdha, T.-H. Wen, B. Thomson, and S. Young, "Neural belief tracker: Data-driven dialogue state tracking," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2017, pp. 1777–1788.

[23] V. Zhong, C. Xiong, and R. Socher, "Global-locally self-attentive dialogue state tracker," *arXiv preprint arXiv:1805.09655*, 2018.

[24] G.-L. Chao and I. Lane, "Bert-dst: Scalable end-to-end dialogue state tracking with bidirectional encoder representations from transformer," in *Interspeech 2019*, 2019.

[25] L. Ren, J. Ni, and J. McAuley, "Scalable and accurate dialogue state tracking via hierarchical sequence generation," in *2019 Conference on Empirical Methods in Natural Language Processing*, 2019, pp. 1876–1885.

[26] V. Balaraman and B. Magnini, "Domain-aware dialogue state tracker for multi-domain dialogue systems." *arXiv preprint arXiv:2001.07526*, 2020.

[27] G. Glavaš and I. Vulić, "Explicit retrofitting of distributional word vectors," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2018, pp. 34–45.

[28] E. M. Ponti, I. Vulić, G. Glavaš, N. Mrkšić, and A. Korhonen, "Adversarial propagation and zero-shot cross-lingual transfer of word vector specialization," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018, pp. 282–293.