A Self-Supervised Integration Method of Pretrained Language Models and Word Definitions

Hwiyeol Jo
NAVER Search US
hwiyeolj@gmail.com

Abstract

We investigate the representation of pretrained language models and humans, using the idea of word definition modeling–how well a word is represented by its definition, and vice versa. Our analysis shows that a word representation in pretrained language models does not successfully map its human-written definition and its usage in example sentences. We then present a simple method DefBERT that integrates pretrained models with word semantics in dictionaries. We show its benefits on newly-proposed tasks of definition ranking and definition sense disambiguation. Furthermore, we present the results on standard word similarity tasks and short text classification tasks where models are required to encode semantics with only a few words. The results demonstrate the effectiveness of integrating word definitions and pretrained language models.[1]

1 Introduction

A word embedding vector maps a word into a fixed-dimensional vector as a distributed representation. The word vectors are trained by looking at their context words and aggregating their representations in supervised ways (Turney, 2013) or unsupervised ways (Mikolov et al., 2013; Pennington et al., 2014). More recently, the representations have been learned as a form of pretrained language models (Peters et al., 2018; Devlin et al., 2019). The huge success of these pretrained language models on various NLP tasks is achieved by capturing a rich semantic representation of words from their context in huge data.

On the other hand, for centuries, lexicographers and linguists have created dictionaries that contain general definitions of words and examples of their usage. With these sophisticated data, there have been many applications for NLP tasks (e.g., machine translation (Hill et al., 2016), semantic relatedness classification (Bahdanau et al., 2017)). Some recent works have used WordNet (Miller, 1995) for fine-tuning BERT for word sense disambiguation (Huang et al., 2019; Guo et al., 2020), whereas our work uses up-to-date dictionary definitions and usage examples to fine-tune pretrained language models.

Distances between word '**love**' and its definitions:
1. An intense feeling of deep affection. (57.8)
A feeling of deep romantic or sexual attachment to someone. (139.8)
2. Affectionate greetings conveyed to someone on one's behalf. (126.6)
3. A formula for ending an affectionate letter. (64.9)
4. A personified figure of love, often represented as Cupid. (149.0)
5. A great interest and pleasure in something. (66.0)
6. A person or thing that one loves. (103.7)
7. A friendly form of address. (44.9)
8. Used in affectionate requests. (93.9)
(in tennis, squash, and some other sports) a score of zero; nil. (117.5)
9. Feel deep affection for (someone) (85.4)
10. Feel a deep romantic or sexual attachment to (someone) (191.5)
11. Like or enjoy very much. (71.3)

The closest definition to the word '**love**':
"**Several.**" (definition of word '**number**') (27.3)

Table 1: The mean squared distance between the word 'love' and its definitions in a dictionary (top; $|\mathtt{W}_i\text{-}\mathtt{D}_{w_i}|$), and the closest distance between the word and any definitions in our collected dictionary (bottom; $|\mathtt{W}_i\text{-}\mathtt{D}_{w_j}|$). Each word or definition is embedded by BERT (see §3).

In this work, we study the difference between *machine-learned definitions* and *human-written definitions*. Table 1 shows the mean squared distance between the vanilla BERT representation (the last hidden layer of [CLS]) for the word 'love' and the sentence representation (by [CLS]) for its definitions in dictionaries. The closest word of 'love' in the pretrained model is 'number' in our data collection. This indicates a potential risk of using pretrained representations as the only means to measure the semantic similarity between words

[1] https://github.com/hwiyeoljo/DefBERT

or short sentences, where the context words are insufficient to get good representations.

Furthermore, it is important to make general and self-indicated embeddings. For example, if we do not have pooling layers to the pretrained embedding, we need additional training data to fine-tune the pooling layer and the pretrained model. On the other hand, if we can do the same task by using the pretrained model only (without fine-tuning), this means good generalization.

Lastly, some researchers believe that target word token representation is better than `[CLS]` token when the input text is short. However, we do not know 'what the short text is' or 'when the model gets short text as inputs.' Thus, the fact that we can use `[CLS]` token for a single word or short text is beneficial in that we do not need to consider the input length. To do so, we attempt to inject word-definition-example (its usage) information into the model.

To overcome the deficiency and get such a generalized model, we propose a new joint representation that combines the human-written word definition with its usage example in a dictionary entry. We show the effectiveness of this new representation on several downstream tasks.

The main contributions are:

- Performed extensive analyses of how close the representations of pretrained language models are to the one of collected human-written definitions; our analyses show that the representations of BERT do not reflect the human-written definitions.
- Incorporated the dictionary definitions into the pretrained language models in embedding-level–As a new model called `DefBERT` (§4), showing significant performance improvements where tasks lack contextual information.
- Proposed two semantics-related ranking tasks: DefRank aims to find the correct definition given the word, and SenseRank is to find the proper sense from a word's definitions given the word's usage. Unsurprisingly but interestingly, `DefBERT` shows significant improvements in both tasks.

2 Related Work

Using dictionaries for NLP tasks. Dict2vec (Tissier et al., 2017) learned word embeddings through word-definition pairs. They designed strong and weak word pairs within dictionaries and made the word pairs close. Bahdanau et al. (2017) utilized dictionaries to solve out-of-vocabulary (OOV) problems by encoding the definitions of OOV words and generating the word's embeddings. Hill et al. (2016) suggested a dictionary-based learning task using neural networks. They also suggested reversed dictionary evaluation tasks that choose the most related word to a given description. Like dictionaries, WordNet (Miller, 1995) has been widely used to enrich word representations (Faruqui et al., 2015). However, the prior works were biased to inject relation knowledge, such as synonyms, rather than general word definitions.

More recently, GlossBERT (Huang et al., 2019) used definitions for disambiguation tasks, but the approach needs context-gloss pairs and a classifier even at inference. In this work, we attempt to build a generalized model which does not require additional classifiers.

Definition Modeling. The definition modeling task was proposed by Noraset et al. (2017) that generates a word definition from the word's embedding. The authors considered the definition modeling as a special case of language modeling and used it for word embedding evaluation. However, Gadetsky et al. (2018) found that the prior definition modeling tasks could not resolve word disambiguation because it is conditioned on only a single word. To address the issue, they also extended Noraset et al. (2017)'s model to process context.

Chang and Chen (2019) investigated whether contextual representations can capture word definitions. Unlike the prior works on definition modeling, they suggested a general framework that maps the contextualized representation into a definition embedding space and then selects top-N closest definitions. This retrieval-based approach can resolve the problems in the generative approach of definition modeling, such as the difficulty in evaluation.

The major differences between the prior works and our study are as follows: First, we compare representations from pretrained language models and definitions from a lexical dictionary at embedding-level. Second, we use word-definition pairs and definition-example pairs from the dictionary. The use of words in a sentence is similar to GlossBERT, but its objective is not to make definition-injected

	Chang and Chen (2019)	Oxford+ (Ours)
# Words (W)	31,889	30,533
# Definition (Def)	79,105	93,227
# Examples (Exam)	707,001	1,167,055
Avg./Max. # Def by Word	10.6/65	10.5/51
Avg./Max. # Exam by Def	17.8/46	18.0/85
Sense order	N	Y

Table 2: Comparison of dictionary datasets. We build on and augment the prior work. The differences in the number of words, definitions, and examples are due to updates.

representation. Rather it is to solve sense disambiguation tasks. The method also requires an additional classifier. Lastly, we propose two tasks that can measure the capability of model representations on human-written definitions (and examples): DefRank and SenseRank. Compared to other benchmark datasets that predict how similar two words or sentences are, we expect these tasks to be a straightforward benchmark.

3 Preliminary Analyses

The central motivation behind our analysis is to check whether a word representation in pretrained language models (in this work, BERT) can indicate the representation of its definition and vice versa.

3.1 Definition Dataset Collection: Oxford+

In prior work, Chang et al. (2018); Chang and Chen (2019) collected an online dictionary from lexico[2] (Oxford University Press, 2020). Since our work requires up-to-date definitions, we re-collected the dataset based on the vocabulary of the original work.

Table 2 shows the comparison and statistics of the dictionary data. The number of unique vocabulary is slightly different from the previous one. However, when considering the number of definitions and the number of examples increases, we think that the difference is due to the updates of lexico dictionary. Dictionaries usually order word senses by how frequently the senses are used, so the order information is important for investigating major versus minor definitions. Due to the more extensive coverage of usage and definitions, and additional information, we call our dataset Oxford+.

From Oxford+, we take two sets of pairs and calculate the distances: one is a pair between a word and its definition (`W-D`), and the other is a pair between a definition and its usage (`D-E`) where the pairs are embedded by pretrained language model.

3.2 Distance Measures

Embedding scheme. We use `bert-base-uncased` in HuggingFace (Wolf et al., 2019) as a backbone model.[3] Although there are several different ways to represent a word or sentence using BERT (e.g., averaging `[CLS]` in every hidden layer, concatenating `[CLS]`, etc.), we use the `[CLS]` token in the last hidden layer, as the original BERT paper proposed.

For all definition-example pairs, we first input the example through BERT and then use the target word tokens in the example instead of using the `[CLS]` token (see Figure 1). We average their vectors if the target word is tokenized by more than one token.

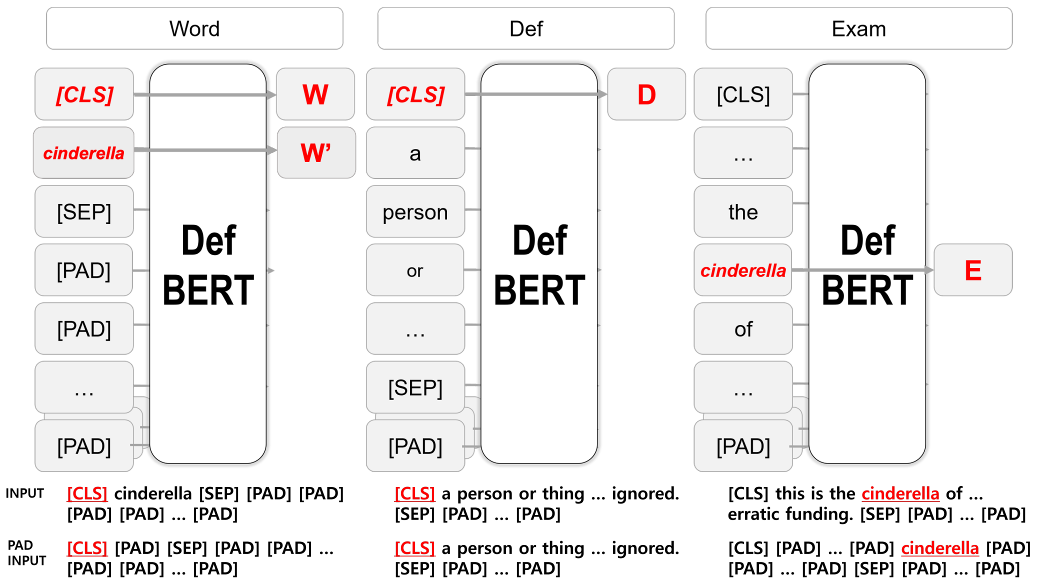
Let i be the word index, ij be the definition index of the j-th definition of the i-th word, and ijk be the index of the k-th example of the j-th definition for word i. Following our central motivation, the distance $|\mathtt{W}_i - \mathtt{D}_{ij}|$ between a word $\mathtt{W}_i$ and one of its definitions $\mathtt{D}_{ij}$ is calculated by mean squared distance. Likewise, the distance $|\mathtt{D}_{ij} - \mathtt{E}_{ijk}|$ between a word used in an example $\mathtt{E}_{ijk}$ and its definition $\mathtt{D}_{ij}$ is calculated by mean squared distance.[4]

In order to compare BERT's ability to capture human-written definitions, we need to control BERT's inputs and weights. We thus use (1) `[PAD]` masked inputs on the target word and (2) BERT with random weights. or example, suppose the embedding of empty input (`BERT([PAD] ... [PAD])`) is closer to the definition embedding (`BERT(D)`) than a single word embedding (`BERT(W)`)). In that case, BERT does not seem to capture definition information through the inputs. The controls by `[PAD]` will be denoted as $\mathtt{W}_{\mathtt{[PAD]}}$ for word and $\mathtt{E}_{\mathtt{[PAD]}}$ for usage example, respectively. The `[PAD]` controlled inputs are also illustrated in Figure 1. Likewise, if BERT with random weights performs better, BERT's pretrained weights do not have information about human-written definitions. We denote the controlled model as `Rand`. With this idea, we can define distance types.

[2]https://www.lexico.com but recently it is redirected to https://www.dictionary.com

[3]In the experiment (§5.1), further fine-tuning through masked language modeling loss over collected definition datasets improves model performances a little. However, we focus on the pretrained representation of vanilla BERT.

[4]Likewise, $|\mathtt{W}_i - \mathtt{E}_{ijk}|$ can be measured but it is out of our



INPUT [CLS] cinderella [SEP] [PAD] [PAD] [PAD] [PAD] ... [PAD] | [CLS] a person or thing ... ignored. [SEP] [PAD] ... [PAD] | [CLS] this is the cinderella of ... erratic funding. [SEP] [PAD] ... [PAD]

PAD INPUT [CLS] [PAD] [SEP] [PAD] [PAD] ... [PAD] [PAD] ... [PAD] | [CLS] a person or thing ... ignored. [SEP] [PAD] ... [PAD] | [CLS] [PAD] ... [PAD] cinderella [PAD] [PAD] ... [PAD] [SEP] [PAD] ... [PAD]

Figure 1: Illustration of word-definition distance and definition-example distance from embedded vectors of the word 'cinderella' (left), the definition "a person or thing ... ignored"(middle), and example "this is the cinderella of ... erratic funding." (right). The target token is underlined and highlighted with the boxes. `[PAD]` means an empty token to be considered meaningless. Best viewed in color.

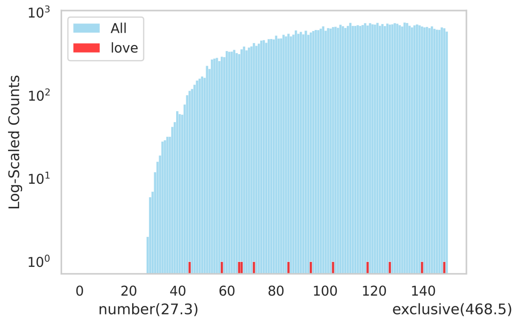


Figure 2: The distribution of $|\mathtt{W}_i - \mathtt{D}_{w_j}|$ where i is the index for word 'love'. The blue bars show log-scaled counts of all the definitions' distances from the target word. The red bars indicate the distance from the different senses of the target words' definitions.

Distance Types. For each word-definition pair and each definition-example pair,

- **|W-D|** : computes the distance between the original input vector (INPUT in Figure 1) and the definition vector for each layer.
- **|W$_{\text{[PAD]}}$-D|** : computes the distance between the padded input vector (PAD INPUT in Figure 1) and the definition vector for each layer.
- **|Rand W-D|** : is the same as |W-D|, but all the model weights are randomly initialized.
- **|D-E|** : computes the distance between the definition vector and the target word vector used in the example sentence.
- **|D-E$_{\text{[PAD]}}$|** : computes the distance between the definition vector and the padded target word vector in the example.
- **|Rand D-E|** : is the same as |D-E|, but all the model weights are randomly initialized.

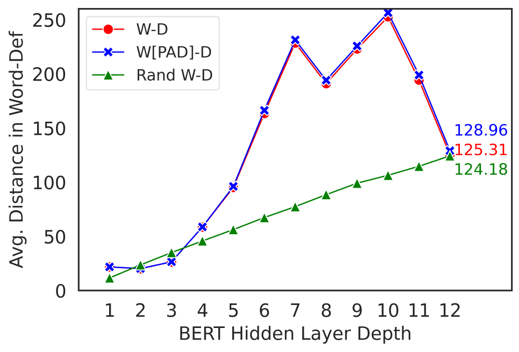
To sum up, the padded inputs and the randomized weights are used to contaminate the model representation. If the contaminated embeddings are closer to definitions than the vanilla input or model embeddings, the model representation is not meaningful.

3.3 Findings

Distribution of Distances. We visualize the distances between a target word 'love' and all definitions in OXFORD+ (Figure 2). As we showed in §1, the closest (or most similar) word to 'love' was 'number'. The definitions of 'love' are scattered over the distribution, indicating how BERT's representation of 'love' is far from its human-written definitions. We observe similar patterns in most words in the dictionary.

The pretrained representation of a word alone is not indicating its human-written definitions. Figure 3 (top) shows the averaged word-definition distances according to hidden layers. The distance of |W-D| is smaller than |W$_{\text{[PAD]}}$-D| distance across all the hidden layer depth. Since the difference between them is only the input, the word itself in-

motivation since the pair does not use definition.



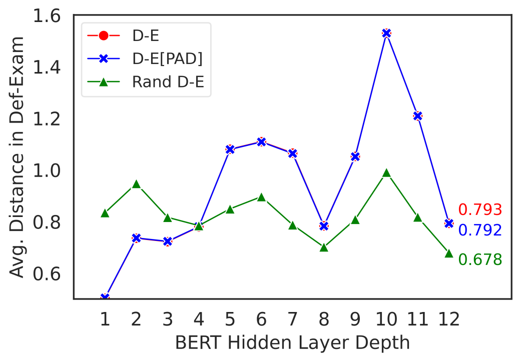


Figure 3: The average distances of |W-D| (top) and |D-E| (bottom) according to distance types (§3.2). Best viewed in color.

cludes information about the human-written definitions definition. In the same plot, however, the distance of randomized BERT |Rand W-D| is much lower than |W-D| and |$\mathtt{W}_{\mathtt{[PAD]}}$-D| at upper layers, which casts doubt about BERT's pretrained weights whether they can represent human-written definitions. We thus conjecture that using a word alone is not appropriate for a contextualized representation since a single word lacks context.

To provide more context for the model, we conduct a second experiment to compare the definition's representation to its usage in the example sentence where pretrained language models have shown strong performances.

BERT can self-indicate better by using surrounding words but it still fails to capture the human-written definitions. Figure 3 (bottom) shows the definition-example distances. The distances of |D-E| and |$\mathtt{D\text{-}E}_{\mathtt{[PAD]}}$| shows similar trends but |$\mathtt{D\text{-}E}_{\mathtt{[PAD]}}$| is smaller at the last hidden layer. The result shows the tokens are less self-indicated in the sentences, while the averaged distance of the randomized model is much smaller than in ordinary settings.

From this analysis, the pretrained language model (especially BERT) seems unable to encode human-written definitions, as |Rand W-D| and |Rand D-E| show lower distance than |W-D| and |D-E|, respectively. Also, the distances between the vanilla BERT and the padded models are small, which tells us that it might have potential benefits by adding semantic information.

4 DefBERT: Definition Induced BERT

Using lexical resources for fine-tuning word embeddings is a typical solution to take advantage of both lexical semantics and distributional semantics. However, as seen in §3, the lexical relations, such as antonyms and synonyms, are unnatural to be integrated with pretrained language models. On the other hand, dictionary definitions and examples are expressed as complete sentences, leading to better settings for optimizing the pretrained models.

Based on the analysis (§3), we present a simple yet effective method to integrate general definitions from a dictionary with pretrained representations while keeping the nature of contextualization. The setup of BERT for fine-tuning is the same as Figure 1; we then fine-tune BERT using the distances as a loss function.

By doing so, we optimize BERT's representation to be close to its human-written definitions (W-D) and its word representation used in the examples (D-E). The loss functions used for each pair are as follows:

$$\mathrm{L}_{\mathtt{W\text{-}D}} = \frac{1}{\#W \times \#D} \sum_i \sum_j \sqrt{(\mathtt{W}_i - \mathtt{D}_{ij})^2} \tag{1}$$

$$\mathrm{L}_{\mathtt{D\text{-}E}} = \frac{1}{\#W \times \#D \times \#E} \sum_i \sum_j \sum_k \sqrt{(\mathtt{D}_{ij} - \mathtt{E}_{ijk})^2} \tag{2}$$

where i is the word index, j is the definition index of the j-th definition of the i-th word, and ijk is the index of the k-th example of the j-th definition for word i. The number of words, definitions, and examples are denoted as $\#W$, $\#D$, and $\#E$, respectively. We use Adam optimizer (Kingma and Ba, 2014) with learning rate 5e-6, and 32 batch size. The maximum token length from our definition data is 191, including special tokens (e.g., [CLS] and [SEP]), but we utilize the model's maximum capacity, which is 512.

However, as we observed in the analysis (§3), the pretrained embeddings of source and target words (i.e., W, D, and E) might not be appropriate to be trained. Therefore, we additionally design loss functions, which utilize the other dictionary information: the distance between [CLS] token of W and

`Easy set`: target word "love"		B	D
C_1^*	affectionate greetings conveyed to someone on one's behalf.	4	**1**
C_2	persist in an activity or process.	**1**	3
C_3	a device for reducing mechanical vibration, in particular a shock absorber on a motor vehicle.	2	4
C_4	denoting popular black culture in general.	3	2
`Challenge set`: target word "love"		**B**	**D**
C_1^*	feelings of deep affection.	4	**1**
C_2	regarded with deep affection. [dear]	2	4
C_3	inspiring affection. [endearing]	**1**	3
C_4	deep love and respect. [adoration]	3	2
`Neologism set`: target word "ohana"		**B**	**D**
C_1^*	especially in hawaii: a family, including members of an extended family, as well as close friends and associates.	4	**1**
C_2	a trouser leg.	**1**	4
C_3	absence of difficulty or effort.	2	3
C_4	an estimation of the quality or worth of someone or something.	3	2

Table 3: Examples in DefRank easy (top), challenge (middle), and neologism (bottom) set. * indicates the gold definition. B and D mean the rank predicted by BERT and `DefBERT`, respectively.

`W` tokens itself (`W'`) to align the token embedding(s) to `[CLS]` token. Likewise, the distance between `W` and `E` is used.

$$L_{\texttt{W-W'}} = \frac{1}{\#W} \sum_i \sqrt{(\texttt{W}_i - \texttt{W'}_i)} \tag{3}$$

$$L_{\texttt{W-E}} = \frac{1}{\#W \times \#D \times \#E} \sum_i \sum_j \sum_k \sqrt{(\texttt{W}_i - \texttt{E}_{ijk})^2} \tag{4}$$

We use the additional loss functions for calibration of `DefBERT`. As a result, we can provide all the information in the dictionary self-supervised way.

In the training process, we prepare two BERT models in order to make the training fast and keep BERT's original properties; One of BERT model makes prediction and update its weights by the loss(es), while the other BERT model only makes prediction used for target embedding. The target BERT is copied in every epoch. After the training, the fine-tuned BERT is selected.

5 Experiments

5.1 DefRank: Definition Ranking Task

Setup. To evaluate the ability of pretrained word vectors to capture human-written definitions at embedding-level (i.e., without classifiers), we present a task called Definition Ranking (DefRank). Given a word, the model predicts the closest word definition among four candidate definitions. The main idea is similar to Chang and Chen (2019), but DefRank looks at only a word and does not require additional mapping function in the evaluation framework, which corresponds to our goal–get a general embedding model. We assign approximately 10% of data to test set[5].

DefRank has two sets based on task difficulty: `Easy` set and `Challenge` set. The candidate definitions in the easy set are randomly sampled from Oxford+. On the other hand, candidate definitions in the challenge set are selected by the closest three definitions except for the gold definitions. We use Sentence-BERT (Reimers and Gurevych, 2019) to choose similar and negative examples as an adversarial constraint. Therefore, models are supposed to capture the subtle differences in meaning among the definitions of words such as love, dear, endearing, and adoration. Table 3 (top) and Table 3 (middle) show the examples.

Furthermore, the easy set has a sub-set called `Neologism` set, which consists of a newly coined word or expression. Thus, we can evaluate the models' ability even when the words never appear in the (pre-)training data.

To collect neologisms, we refer to the update notes of Oxford Dictionary and consider 'new word entries' as neologisms. We then process them by removing words that require a subscription to see the full definition and references in definitions to other similar words (e.g., See, Cf. and explanations after ';'). The number of collected neologisms is 345. Table 3 (bottom) presents the example of neologism.

We compare BERT variations, such as BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), Sentence-BERT (Reimers and Gurevych, 2019), and GlossBERT (Huang et al., 2019). Besides, we report the performance fine-tuned by masked language modeling on the definition data. In the masked language training, we set an artificial template that "the definition of `W` is `D` and its example is `E`." As we mentioned in §4, `W-W'` pairs and `W-E` pairs are used for model calibration, denoted as `[+W']` and `[+E]`, respectively.

We also empirically find the optimized pair selection of `DefBERT`, which shows the best performances in DefRank, denoted as `BestSelect`[6]).

[5]After we post-process to clean the test data, the ratio becomes approximately 9%

[6]The best sequence of training is `[+E]+W-D+D-E+[+W']`.

Model	Easy	Chal.	Neo
Randomized BERT	29.11	26.52	31.01
BERT-base	32.41	25.81	36.52
BERT-base(MLM-FT)	36.32	26.04	29.28
BERT-large	33.91	25.79	36.81
RoBERTa-base	26.07	25.84	62.98
Sentence-BERT	75.08	*30.45	65.22
GlossBERT	49.58	26.93	52.17
ConceptNet	83.88	32.58	35.36
DefBERT(W-D)	60.11	27.92	51.59
DefBERT(D-E)	74.28	31.11	**70.72**
DefBERT([+W'])	61.65	29.51	49.28
DefBERT([+E])	78.55	31.53	68.12
DefBERT([+W']W-D)	74.22	30.59	60.58
DefBERT([+E]W-D)	83.27	32.29	69.28
DefBERT([+W']D-E)	79.04	32.32	68.99
DefBERT([+E]D-E)	80.73	32.54	67.25
DefBERT(BestSelect)	**84.67**	**33.76**	70.43

Table 4: DefRank performance (top-1 accuracy) in the test set. [+] denotes the model calibrated with the dataset. *We acknowledge that Sentence-BERT is not a fair comparison in the challenge set, but see its performance on the other sub-tasks (easy and neo).

Finally, we will report the BestSelect model performance.

Results. Table 4 shows the performance on the DefRank task. Considering the high performance of Sentence-BERT, our tasks are well-designed to examine the semantics incorporated in model representations. The results show that fine-tuning by masked language modeling is ineffective in the performances. Besides, GlossBERT does not perform well on these tasks, which implies that the word disambiguation model largely depends on the classifiers at the end of the architecture.

Our variations of DefBERT show much better performance since we train models with a similar distribution. However, it is interesting that D-E pairs increase the model performances more, even though W-D pairs are directly related to the tasks.

The performance gaps between baselines and our variations are small for the challenge set. Therefore, the challenge set is very hard to distinguish the subtle variation of semantics, requiring a deeper understanding of definitions.

Lastly, we can find several properties of definition pairs. For example, calibrations with only [+W'] or [+E] make significant improvements to the model. The models starting with calibration perform much better than the models without calibrations. We guess that BERT's self-attention successfully normalize the model. Moreover, DefBERT

	Input example "their love for their country" for target word "love"	B	D
C_1^*	an intense feeling of deep affection.	3	**1**
C_2	a great interest and pleasure in something.	2	2
C_3	affectionate greetings conveyed to someone on one's behalf.	4	3
C_4	a formula for ending an affectionate letter.	**1**	4

Table 5: Examples in SenseRank task. * means the gold definition. B and D mean the rank predicted by BERT and DefBERT, respectively.

proves to be effective in neologisms. We conjecture that DefBERT learns unseen words (and their tokens) through other words' definitions. We also report the performance of ConceptNet vector (Speer et al., 2017). The representation is a strong baseline since the embeddings are fine-tuned and specialized in a number of tasks regarding word semantics. When evaluation, the sentence vectors are made by averaging the word vectors. ConceptNet shows good performance on the easy set and the challenge set, which also tells us that DefRank correlates with word semantics tasks while hardly being correct in neologism. The combination of various types of lexical resources (e.g., dictionary, relation, WordNet) remains an interesting direction for future work.

5.2 SenseRank: Sense Disambiguation Task

Setup. Extending from DefRank, we propose another task SenseRank that distinguishes the different senses of definitions for the same word. In this setting, we provide a word and its usage, an example sentence. Then, models select the most appropriate sense of definitions among the word's definitions. Compared to Chang and Chen (2019), SenseRank has to choose a gold definition among the candidate definitions from the same target word. Therefore, the task can be used to measure the model's ability to do fine-grained sense disambiguation.

Table 5 shows four definitions for the target word 'love'. Given an example sentence, DefBERT correctly predicts the most similar sense of the definitions, while BERT fails. Similar to the challenge set of DefRank, the candidate definitions in SenseRank are semantically very similar (i.e., the variation of their senses), but this task has more contexts than DefRank.

We filter out the words for which the number of definitions is fewer than four. We then sample 10% (115,849) as a test set.

Model	SenseRank
BERT-base	54.83
BERT-base(MLM-FT)	41.33
BERT-large	27.78
RoBERTa-base	43.23
Sentence-BERT	**86.59**
GlossBERT	52.25
ConceptNet	39.38
DefBERT(W-D)	74.94
DefBERT(D-E)	**97.54**
DefBERT([+W'])	90.02
DefBERT([+E])	93.76
DefBERT([+W']W-D)	92.67
DefBERT([+E]W-D)	96.24
DefBERT([+W']D-E)	97.02
DefBERT([+E]D-E)	96.51
DefBERT(BestSelect)	**97.27**

Table 6: SenseRank performance in the test set.

Results. Table 6 shows the performances on SenseRank. Similar to the DefRank, the accuracies from BERT variants are relatively low except for Sentence-BERT, which is good at encoding semantics. Apart from D-E pairs that is closely related to SenseRank, other types of data pairs (i.e., W-D), and +W' and +E for calibration) increase the model performances. Also, DefBERT with best selection shows the largest improvement. The results indicate that the setup of DefBERT learns the sense-specific patterns between definitions and examples. Moreover, ConceptNet performs worse than most of the BERT-variants, showing that context is an important factor in this task.

5.3 Downstream Task 1: Word-Similarity

Setup. Word similarity tasks can be used to evaluate word representations. They make use of Spearmann correlations to assess agreement between human ratings and computational representations of the similarity between word pairs. We use the evaluation tasks–WordSim (Finkelstein et al., 2001; Agirre et al., 2009), RareWord (Luong et al., 2013), MEN (Bruni et al., 2012), SemEval (Camacho-Collados et al., 2017), SimLex (Hill et al., 2015), and SimVerb (Gerz et al., 2016). For DefBERT, we choose the best selection model in DefRank. Note that there is no additional training on the word similarity datasets.

Results. Table 7 shows performances on the word similarity tasks. The other embeddings, except for DefBERT show poor performances. Additional masked language modeling fine-tuning increases the performance only a little. We conjecture that word similarity/relatedness tasks are very challenging for pretrained and contextualized models because no context is given (see §6 for further discussion). The result is the same as what we found in our preliminary distance analysis on word-definition pairs. On the other hand, DefBERT largely closes the gaps among word, definition, and usage, which leads to significant improvements from BERT in all the datasets.

$\rho \times 100$	W-S	W-R	RW	MEN	SEM	SL	SV	Avg
BERT	23.1	1.8	5.3	19.1	10.8	7.2	0.8	9.7
BERT(FT)	30.8	13.0	6.5	17.7	10.5	5.6	2.5	12.4
Sent-BERT	33.1	23.2	40.6	60.6	49.3	**61.9**	**49.9**	45.5
GlossBERT	26.6	-3.6	25.7	30.8	30.7	28.3	15.0	21.9
DefBERT	**71.6**	**51.8**	**46.7**	**76.5**	**58.7**	53.2	41.1	**57.1**

Table 7: Model performances on word similarity tasks. WordSim dataset is categorized into semantics (W-S) and relation (W-R).

	TREC	SST2	IMDB
BERT	97.1(.3)	**92.7(.2)**	93.4(.1)
BERT(MLM-FT)	97.3(.3)	91.4(.4)	**93.5(.1)**
Sent-BERT	**97.3(.2)**	91.6(.3)	93.4(.1)
GlossBERT	96.8(.4)	91.3(.3)	92.9(.1)
DefBERT	**97.3(.2)**	**92.7(.4)**	93.3(.1)

Table 8: We reported five times averaged performance on the text classification datasets. The dataset is ordered by its average word length in instances, which is approximately 10, 19, and 234, respectively.

5.4 Downstream Task 2: Short Text Classification

Setup. As we mentioned in §1 and showed in the previous experiments §5, BERT embedding for a word or short text did not make good representations. In order to generalize the effect of our integration, we employ text classification datasets–TREC (Hovy et al., 2001), SST-2 (Socher et al., 2013), and IMDB (Maas et al., 2011). All the datasets are relatively small, and the text length is short in TREC and SST-2, whereas IMDB is rather long. We report IMDB performance to show the performance of long text.

As the original paper did, we use a [CLS] token at the last hidden layers. The hyperparameters are 2e-5 for learning rate, 32 for mini-batch size. We use Adam optimizer (Kingma and Ba, 2014). If the dataset does not have a validation set, we assign 15% of the training set and use them for early-stopping. The maximum length of tokens is 512.

Results. We present the performance of text classification in Table 8. Compared to other methods, `DefBERT` shows comparable performance with other baselines. Although the performance gap is small (we guess that the baseline is already strong), `DefBERT` shows the best performance on the shortest dataset TREC, which has only maximum 37 words (by space). On the other hand, IMDB has approximately maximum 3000 words. Though GlossBERT is also fine-tuned by external data (specifically, gloss), the result indicates that word disambiguation tasks are not related to representing a single word or short sentence.

6 Conclusion and Further Discussion

We present a novel way of combining pretrained contextualized representations and human-written definitions from a dictionary. We first collect definitions and examples from an online dictionary OXFORD+. Our analyses with the dictionary show that BERT's representations do not incorporate human-written definitions. Motivated by the findings, we develop a new representation `DefBERT`, by constraining BERT to human-written definitions in the dictionary. In the experiments, we first proposed definition ranking (DefRank) and sense disambiguation tasks (SenseRank) and `DefBERT` outperforms other baselines. We also presented the effectiveness of `DefBERT` in downstream tasks: word similarity benchmark and short text classification tasks.

One of the contributions of this paper is to make researchers revisit the old and traditional resource, dictionaries. While resources, including synonyms, antonyms, and other relations, are widely used to improve models as a constraint, dictionaries are less frequently used. However, the dictionary is the basic form of word semantics and is a relatively objective resource compared to relational resources.

Furthermore, word-related resources are hard to align with pretrained language models because the weights are dynamic according to contexts. Therefore, pouring resources can occur catastrophic forgetting that the information previously trained disappears. On this problem, we suggest a potential approach to enhance semantics on the pretrained weights, maintaining the nature of contextualized encoder.

7 Limitations

The performances except for the proposed tasks. We presented the result of neologism and the performances on two downstream tasks (i.e., word similarity task and short text classification), which are closely related to the understanding of word semantics. The selected downstream tasks are challenging for the contextualized models; they can use only a few contexts to make a representation.

The performance in general benchmarks (e.g., GLUE) is almost the same as the vanilla BERT because our model suffers catastrophic forgetting while learning definition information. Sophisticated modeling and training processes to overcome the problem could be interesting future work.

The use of other models Other pretrained models like RoBERTa could be a base model of our method (e.g., DefRoBERTa). However, we think that BBPE tokens scarcely have semantic meanings, which makes it hard to find appropriate tokens to inject definition information. Therefore, integrating human-written definitions with other types of tokens (e.g., Byte-Pair Encoding and Byte-level BPE) is also a future direction.

The use of all the loss function & Collect more definition data. Presenting more experiments with other models, other collections of definition data, and other loss functions will further support our idea. Nevertheless, we want to show the performances with the widely-used basic model of pretrained language models (i.e., BERT), using definition data from the previous work, with various loss functions (e.g., `W-D, D-E, [+W'], [+E]`) as many as possible. A fine-grained combination of all the loss functions could make further improvements.

Acknowledgement

The author would like to thank previous co-workers who discussed this idea long ago, including reviewers in several rounds of submission. Also, PAUST gave helpful advice on experimental techniques and distributed software engineering. Lastly, I am grateful to Alice Lee for help in writing this work.

References

Eneko Agirre, Enrique Alfonseca, Keith Hall, Jana Kravalová, Marius Pasca, and Aitor Soroa. 2009. A study on similarity and relatedness using distributional and wordnet-based approaches. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 19–27.

Dzmitry Bahdanau, Tom Bosc, Stanisław Jastrzębski, Edward Grefenstette, Pascal Vincent, and Yoshua Bengio. 2017. Learning to compute word embeddings on the fly. *arXiv preprint arXiv:1706.00286*.

Elia Bruni, Gemma Boleda, Marco Baroni, and Nam-Khanh Tran. 2012. Distributional semantics in technicolor. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 136–145.

Jose Camacho-Collados, Mohammad Taher Pilehvar, Nigel Collier, and Roberto Navigli. 2017. SemEval-2017 task 2: Multilingual and cross-lingual semantic word similarity. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 15–26, Vancouver, Canada. Association for Computational Linguistics.

Ting-Yun Chang and Yun-Nung Chen. 2019. What does this word mean? explaining contextualized embeddings with natural language definition. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6066–6072.

Ting-Yun Chang, Ta-Chung Chi, Shang-Chi Tsai, and Yun-Nung Chen. 2018. xsense: Learning sense-separated sparse representations and textual definitions for explainable word sense networks. *arXiv preprint arXiv:1809.03348*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. 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)*, pages 4171–4186.

Manaal Faruqui, Jesse Dodge, Sujay Kumar Jauhar, Chris Dyer, Eduard Hovy, and Noah A. Smith. 2015. 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*, pages 1606–1615, Denver, Colorado. Association for Computational Linguistics.

Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. 2001. Placing search in context: The concept revisited. In *Proceedings of the 10th international conference on World Wide Web*, pages 406–414.

Artyom Gadetsky, Ilya Yakubovskiy, and Dmitry Vetrov. 2018. Conditional generators of words definitions. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 266–271.

Daniela Gerz, Ivan Vulić, Felix Hill, Roi Reichart, and Anna Korhonen. 2016. Simverb-3500: A large-scale evaluation set of verb similarity. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2173–2182.

Ping Guo, Yue Hu, and Yunpeng Li. 2020. Mg-bert: A multi-glosses bert model for word sense disambiguation. In *International Conference on Knowledge Science, Engineering and Management*, pages 263–275. Springer.

Felix Hill, KyungHyun Cho, Anna Korhonen, and Yoshua Bengio. 2016. Learning to understand phrases by embedding the dictionary. *Transactions of the Association for Computational Linguistics*, 4:17–30.

Felix Hill, Roi Reichart, and Anna Korhonen. 2015. Simlex-999: Evaluating semantic models with (genuine) similarity estimation. *Computational Linguistics*, 41(4):665–695.

Eduard Hovy, Laurie Gerber, Ulf Hermjakob, Chin-Yew Lin, and Deepak Ravichandran. 2001. Toward semantics-based answer pinpointing. In *Proceedings of the First International Conference on Human Language Technology Research*.

Luyao Huang, Chi Sun, Xipeng Qiu, and Xuan-Jing Huang. 2019. Glossbert: Bert for word sense disambiguation with gloss knowledge. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3500–3505.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Minh-Thang Luong, Richard Socher, and Christopher D Manning. 2013. Better word representations with recursive neural networks for morphology. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 104–113.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Tomas Mikolov, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781.

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

Thanapon Noraset, Chen Liang, Lawrence A Birnbaum, and Douglas C Downey. 2017. Definition modeling: Learning to define word embeddings in natural language. In *31st AAAI Conference on Artificial Intelligence, AAAI 2017*.

Oxford University Press. 2020. a new collaboration between dictionary.com and oxford university press (oup). http://lexico.com/.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*, pages 1532–1543.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of NAACL-HLT*, pages 2227–2237.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3973–3983.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: an open multilingual graph of general knowledge. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, pages 4444–4451.

Julien Tissier, Christophe Gravier, and Amaury Habrard. 2017. Dict2vec: Learning word embeddings using lexical dictionaries. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 254–263.

Peter D. Turney. 2013. Distributional semantics beyond words: Supervised learning of analogy and paraphrase. *TACL*, 1:353–366.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
7

☒ A2. Did you discuss any potential risks of your work?
Overall process has no potential risk

☑ A3. Do the abstract and introduction summarize the paper's main claims?
1

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☒ Did you use or create scientific artifacts?

Left blank.

☐ B1. Did you cite the creators of artifacts you used?
No response.

☐ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
No response.

☐ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
No response.

☐ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
No response.

☐ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
No response.

☐ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
No response.

C ☑ Did you run computational experiments?

5

☒ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
We used well-known BERT.

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
5

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
5

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
5

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?
Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
No response.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
No response.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
No response.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
No response.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
No response.