Improving Skip-Gram Embeddings Using BERT

Yile Wang, Leyang Cui, and Yue Zhang

***Abstract*—Contextualized embeddings such as BERT and GPT have been shown to give significant improvement in NLP tasks. On the other hand, static embeddings such as skip-gram and GloVe still have desirable characteristics such as low computational cost, easy deployment and freedom from severe contextualized variation in representation. There has been some recent attempt enhancing the skip-gram model by adding syntactic information of context using GCN. We investigate the use of BERT embeddings instead for stronger context representation, which contains not only syntactic and surface features, but also rich knowledge from large-scale pre-training. Results show that BERT-enhanced skip-gram embeddings outperform GCN-enhanced embeddings on a range of tasks. Such embeddings also outperform recent effort distilling BERT embeddings into context-independent vectors.**

***Index Terms*—BERT, contextualized embeddings, skip-gram, word embedding.**

I. Introduction

WORD representation is a fundamental problem of NLP, attracting interest from both the linguistic perspective and the end-task perspective. Early research has considered matrix factorization methods, such as latent semantic analysis [1] and shallow window-based methods, such as [2]–[5]. With advances of neural networks, word embeddings have been investigated as a part of neural language model training [6], [7]. In particular, the skip-gram model [6] is a widely-used method that allows fast training by using noise contrastive estimation (NCE) over a simplified log-bilinear language model. The main idea is to train word embeddings by predicting target words using its context words.

Recently, contextualized word representations such as ELMo [8], GPT [9], [10], BERT [11] and XLNet [12] have been shown a more effective input representation method for downstream tasks. Compared with static word representation methods such as skip-gram [6] and GloVe [7], contextualized embeddings use a deep neural network to find a hidden-layer word representation given a sentence (or multi-sentence) level context, which can differ for the same word according to different contexts. Contextualized embeddings are correlated with syntactic [13], semantic [14] and factual knowledge [15], outperforming static embeddings on a wide range of tasks such as question answering [16], reading comprehension [17], commonsense reasoning [18] and natural language inference [19].

Despite the advantage of contextualized embeddings in task performance, they have several limitations. For instance, the models are large and complex neural networks containing billions of parameters. This can make them costly to use in terms of both computing resources and disk space, and adds to the cost for integrating them into NLP models compared to static embeddings. In addition, the vector representation of each word varies according to different model layers and context, and can represent context words more than the current word [20]. This results in large variation between representations of the same word across different sequence contexts, particularly for low-resource classes [21], [22]. Such variation is beyond what we can expect from the perspective of semantic polysemy. This is not only undesirable linguistically but also has implications on downstream tasks such as word alignment [23], [24].

In contrast, static embeddings are still valuable in at least two aspects. First, they are light-weight compared with contextualized embeddings in terms of both the model size and computation cost, without necessarily losing performance by a large margin [25]. Second, they are directly useful for tasks that are context-independent, which require a single fixed dense embedding for each given word. As a result, there has been recent attempt for deriving a static version of BERT embeddings. To this end, Bommasani *et al.* [26] take a relatively simple method, contextualized representations of words are calculated over a large corpus, from which the representations of a certain word over different network layers are averaged as a static representation of the word. We treat this method as one baseline.

Another recent work improves skip-gram embeddings by enriching the representation of context words when training target word embeddings. In particular, the skip-gram model uses static embeddings to represent both context words and target words. Instead, Vashishth *et al.* [27] apply a graph convolutional neural network (GCN) [28] for integrating rich syntactic and semantic features into the model, showing that such information from context improves the output target embeddings. We consider integrating BERT and skip-gram by using BERT instead of a GCN for enriching the context during the training of a skip-gram model. The advantage is at least two-fold. First, polysemous words are represented using BERT embeddings, thereby reducing word sense ambiguities [29]. Second, syntactic and semantic information over the entire sentence is integrated into the center word representation [13], [14], thereby providing richer features compared to a word window.

Manuscript received April 21, 2020; revised October 8, 2020 and January 8, 2021; accepted March 5, 2021. Date of publication March 11, 2021; date of current version April 8, 2021. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Eric Fosler-Lussier. *(Corresponding author: Yue Zhang.)*

Yile Wang and Leyang Cui are with Zhejiang University, Hangzhou 310027, China with the School of Engineering, Westlake University, Hangzhou 310027, China, and also with the Westlake Institute for Advanced Study, Hangzhou 310024, China (e-mail: wangyile@westlake.edu.cn; cuileyang@westlake.edu.cn).

Yue Zhang is with the School of Engineering, Westlake University, and the Westlake Institute for Advanced Study, Hangzhou 310024, China (e-mail: yue.zhang@wias.org.cn).

Digital Object Identifier 10.1109/TASLP.2021.3065201

Experiments over a range of intrinsic and extrinsic tasks show that our method outperforms the skip-gram model, averaging BERT, the method of integrating GCN and other static word embedding models, which demonstrate the advantage of leveraging contextualized embeddings to improve lexical semantics and downstream tasks. To our knowledge, we are the first to systematically integrate contextualized embeddings for improving the quality of static word embeddings. Our model and trained embeddings are released at https://github.com/ylwangy/bert2vec.

II. Related Work

Static Word Embeddings. Skip-gram (SG) and continuous-bag-of-words (CBOW) are two models based on distributed word-context pairs [6]. The former predicts the context words using a center word, while the latter predicts a center word using its context words. Wang *et al.* [30] claim that not all contexts are equal and considered word order in the skip-gram model. Hall *et al.* [31] and Levy *et al.* [32] further inject syntactic information by building word embeddings from the dependency parse trees over texts. GloVe [7] learns word embeddings by factorizing global word co-occurrence statistics. Our model follows the skip-gram framework. The main difference between our work and the above methods is that center words are represented using the contextualized information, rather than a lookup table statically.

Contextualized Word Embeddings. ELMo [8] provides deep word representations generated from LSTM based language modeling, GPT [9], [10] improves language model pretraining based on Transformer [33], BERT [11] investigates self-attention-network for deep bidirectional representations, RoBERTa [34] uses more data and optimizes the training of BERT, Longformer [35] is designed for modeling long documents based on BERT, XLNet [12] takes a generalized autoregressive pretraining model based on Transformer-XL [36]. The above models are designed to improve downstream tasks and outperform static embeddings in extrinsic evaluation. However, they are significantly larger in terms of the model size and slower in terms of runtime complexity. We consider using BERT for enhancing skip-gram.

Integrating Contextualized Information into Static Word Embeddings. There have been a few investigations into increasing the representation power of context embeddings in the training of static word vectors. Melamud *et al.* [37] use a bi-directional LSTM to replace the static context embedding table for the model. However, rather using the static embeddings, they used the contextualized LSTM embeddings for improving downstream tasks such as sentence completion, lexical substitution and word sense disambiguation tasks. This is similar to subsequent work in contextualized word vectors. While they did not report results on embedding evaluation benchmarks, we reimplement their method and compare it with our models.

SynGCN [27] use graph convolution network (GCN) to integrate syntactic context for learning context embeddings. Our work is similar in calculating word representations using sentential information. The main difference is that, while their model uses dependency parse trees and graph convolution network for better incorporating syntactic and semantic information, we directly model the sequential context by using BERT contextualized representation trained over large data.

Similar in spirit to our work, Rishi *et al.* [26] distill BERT representations across different encoding layers as static embeddings of words. They find that the distilled embeddings give comparable results to static embeddings in lexical semantic tasks. In contrast to their work, we try to better make use of the contextualized disambiguation power of contextualized embeddings for enhancing static embedding training algorithms, therefore building a model that more deeply integrates a contextualized representation and a distributed vector learning model.

III. Background

We take skip-gram [6] as our framework for training static word embeddings, considering that a similar architecture has been used by previous work [27], [37], which can serve as fair baselines. BERT [11] is used as the contextualized embeddings to replace the center word embeddings in our model. Below we introduce both models.

Skip-Gram. Given a sentence $s = w_1, w_2, \ldots, w_n (w_i \in D)$, we model each word w_i by using its context words w_{i-ws},..., w_{i-1}, w_{i+1},..., w_{i+ws}. The center word and context words are projected into two types of embeddings v_i and v'_{i+j} ($1 \leq |j| \leq ws$), respectively, as shown in Fig. 1(a). Given a training corpus with N sentences $C = \{s_c = w_1, w_2, \ldots, w_{n_c}\}|_{c=1}^{N}$, the training objective is to minimize:

$$L_{SG} = -\sum_{c=1}^{N}\sum_{i=1}^{n_c}\sum_{1\leq|j|\leq ws} \log f(v'_{i+j}, v_i) \tag{1}$$

herein $f(v'_{i+j}, v_i) = p(w_{i+j}|w_i)$ represents the concurrence probability of word w_{i+j} given the word w_i, which is estimated by:

$$p(w_{i+j}|w_i) = \frac{\exp({v'_{i+j}}^{\top} v_i)}{\sum_{w_k \in D} \exp({v'_k}^{\top} v_i)} \tag{2}$$

In practice, we use negative-sampling to avoid the computation cost of Eq. 2 summing over the whole vocabulary (see Eq. 15). During training, each word in the vocabulary uses the same embedding tables V and V' across sentences.

BERT. BERT consists of a multi-layer bidirectional Transformer [33] encoder. The masked language model (MLM) objective is to predict certain masked words through its contextualized representation, as shown in Fig. 1(b).

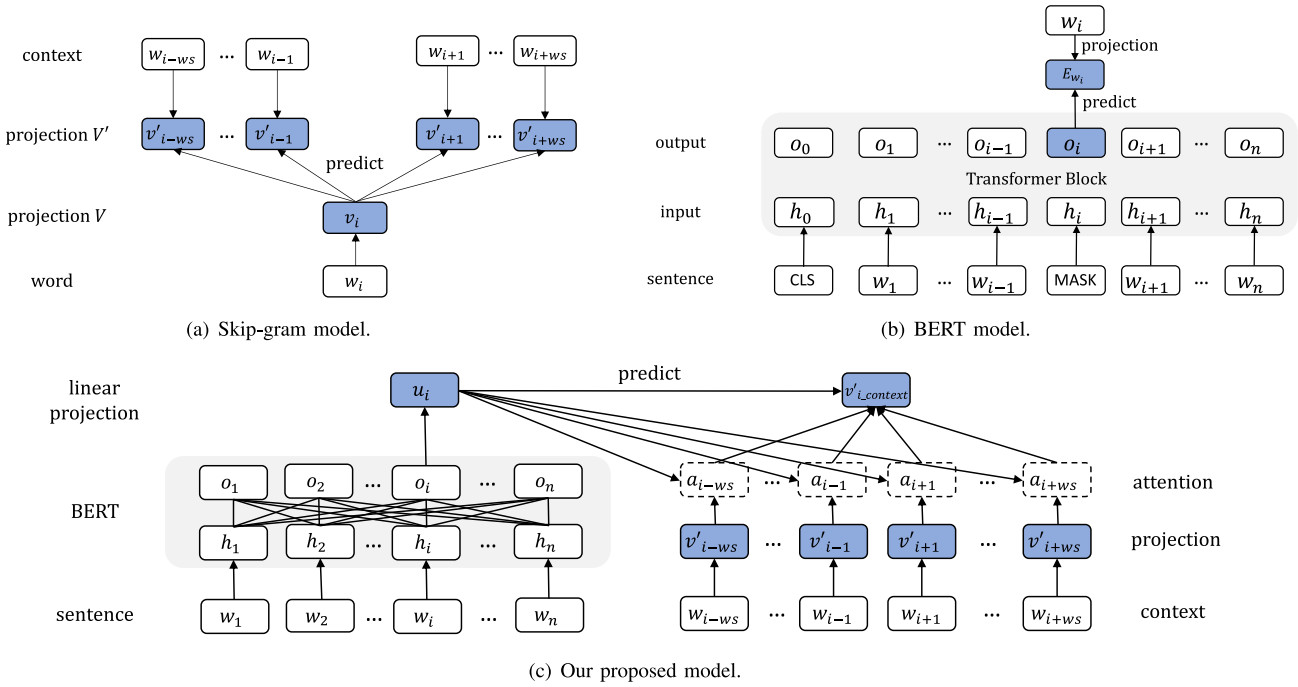
Formally, given a sentence $s = w_1, w_2, \ldots, w_n$, each w_i is transformed into input vector h_i by summing up the static WordPiece [38] token embeddings E_{w_i}, the segment embeddings SE_{w_i} and the position embeddings PE_{w_i}:

$$h_i = E_{w_i} + SE_{w_i} + PE_{w_i} \tag{3}$$

where SE_{w_i} distinguishes the sentence location and PE_{w_i} indicates character position.

The input vectors $H = \{h_1, \ldots, h_n\}$, $H \in \mathbb{R}^{n\times d}$ are then transformed into queries Q^m, keys K^m, and values V^m, $\{Q^m, K^m, V^m\} \in \mathbb{R}^{n\times d_k}$:

$$Q^m, K^m, V^m = HW_Q^m, HW_K^m, HW_V^m \tag{4}$$



(a) Skip-gram model.

(b) BERT model.

(c) Our proposed model.

Fig. 1. Skip-gram, BERT and our proposed model. The blue blocks denote the representation of words.

where $\{W_Q^m, W_K^m, W_V^m\} \in \mathbb{R}^{d \times d_k}$ are trainable parameters, $m \in \{1, \ldots, M\}$ represent the m-th attention head. M parallel attention functions are applied to produce M output states $\{O^1, \ldots, O^M\}$:

$$\begin{aligned} A^m &= softmax(\frac{Q^m {K^m}^\top}{\sqrt{d_k}}) \\ O^m &= A^m V^m \end{aligned} \tag{5}$$

A^m is the attention distribution for the m-th head and $\sqrt{d_k}$ is a scaling factor. Finally, each head for O_i are concatenated to obtain the final output of word w_i:

$$o_i = [O_i^1, \ldots, O_i^M] \tag{6}$$

Given a corpus $\{s_c = w_1, w_2, \ldots, w_{n_c}\}|_{c=1}^N$, the objective is to minimize the loss of predicting a randomly chosen masked word w_{mask_i} in $w_1, w_2, \ldots, w_{i-1}, \langle mask \rangle, w_{i+1}, \ldots, w_n$ by its output representation o_{mask_i} in Eq. 6:

$$L_{MLM} = -\sum_{c=1}^{N} \sum_{i=1}^{n_c} \log p(E_{w_{mask_i}} | o_{mask_i}) \tag{7}$$

where E is the token embedding in Eq. 3, $p(E_{w_{mask_i}} | o_{mask_i})$ is calculated as with Eq. 2:

$$p(E_{w_{mask_i}} | o_{mask_i}) = \frac{\exp({E_{w_{mask_i}}}^\top o_{mask_i})}{\sum_{w_k \in D} \exp({E_{w_k}}^\top o_{mask_i})} \tag{8}$$

IV. The Proposed Approach

Given a sentence $s = w_1, w_2, \ldots, w_n$, we model a center word w_i and its context words w_{i-ws},..., w_{i-1}, w_{i+1},..., w_{i+ws} as in the skip-gram model. To integrate contextualized embeddings, we use BERT to replace the center word embeddings v_i, so that each center word w_i is represented in a sentential context. To this end, a center word w_i is first transformed into h_i, which is the sum of the token embedding E_{w_i} and the position embedding PE_{w_i}:

$$h_i = E_{w_i} + PE_{w_i} \tag{9}$$

Then h_1, h_2,..., h_n are fed into a L-layer bidirectional Transformer block, as described in Eq. 4 and Eq. 5. In particular, we use a pre-trained BERT [11] model to generate the output representations o_i, where numbers of layers $L = 12$, attention heads $M = 12$ and model size $d = 768$.

A linear projection layer is used for transforming the output $o_i \in \mathbb{R}^d$ to $u_i \in \mathbb{R}^{d_{emb}}$:

$$u_i = U o_i \tag{10}$$

where $U \in \mathbb{R}^{d_{emb} \times d}$ are model parameters.

In practice, WordPiece tokenization may contain subwords, thus there exist multiple outputs for a certain complete word. We apply the mean pooling operation to these outputs from subwords to generate the final output for a single word.

To model co-occurrence between the center word w_i and its context words w_{i-ws},..., w_{i-1}, w_{i+1},..., w_{i+ws}, we maximize the probability of the context words $w_{i+j} (1 \leq |j| \leq ws)$ given the contextualized representation u_i of the center word:

$$p(w_{i+j} | w_i) = \frac{\exp({v'_{i+j}}^\top u_i)}{\sum_{w_k \in D} \exp({v'_k}^\top u_i)} \tag{11}$$

Similar to Eq. 2, v'_k are the context word embeddings for w_k by using a static embedding table. Again, we use negative-sampling (Eq. 15) to avoid the computational bottleneck.

Note that our model is not a direct adaptation of the skip-gram model by replacing one embedding table. The original skip-gram algorithm uses the center word embedding table as the

final output embeddings. However, to make the context words predictable and enable negative sampling from the vocabulary, we use BERT representation for the center word, and the context word embedding table as the final output static embeddings.

Attention Aggregation. Not all context words contribute equally to deciding the representation for a word. For example, predicting the stop words (e.g., *"the"*, *"a"*) is less informative than more meaningful words. One method to solve this problem is sub-sampling [39]. A word w_i is discarded with a probability by:

$$P(w_i) = 1 - \sqrt{\frac{t}{f(w_i)}} \tag{12}$$

where $f(w_i)$ is the frequency of word w_i in the training corpus and t is a chosen threshold, typically around 10^{-5}.

Sub-sampling is used in the skip-gram model. However, it cannot be directly used in our method because contextualized representation can be undermined with words being removed from a sentence. We choose instead to select more indicative context words automatically while keeping the training sentence complete. Formally, we apply the attention mechanism to aggregate context words for each center word w_j by using u_i as the query vector and v'_j as the key vectors:

$$a_j = \text{ATT}(u_i, v'_j) \tag{13}$$

where $\text{ATT}(\cdot)$ denotes the dot-product attention operation [40].

The context embeddings are then combined using the corresponding attention coefficient:

$$v'_{i\_context} = \sum_{1 \le |j| \le ws} a_{i+j} v'_{i+j} \tag{14}$$

Training. The skip-gram models in Eq. 2 and Eq. 11 approximate the original training objectives in the cross-entropy for w by using noise contrastive estimating. Given $\{s_c = w_1, w_2, \ldots, w_{n_c}\}|_{c=1}^{N}$, the new objective is to minimize a noise contrastive estimation loss function with negative sampling:

$$L = -\sum_{c=1}^{N}\sum_{i=1}^{n_c}\left(\log\sigma(v'^{\top}_{i\_context}u_i) + \sum_{m=1}^{k}\mathbb{E}_{w_{neg_m}\sim P(w)}[\log\sigma(-v'^{\top}_{neg_m}u_i)]\right) \tag{15}$$

where σ is the sigmoid function, w_{neg_m} denotes a negative sample, k is the number of negative samples and $P(w)$ is the noise distribution set as the unigram distribution $U(w)$ raised to the 3/4 power (i.e., $P(w) = U(w)^{3/4}/Z$).

The final embeddings v' are optimized through stochastic gradient descent. It has been shown that when $k \to \infty$ the gradient of the NCE loss equal the gradients of the cross-entropy loss.

Testing. Following [27], the trained embeddings are tested for lexical semantics tasks. First, the similarity score between two words are calculated based on the cosine similarity between their embeddings:

$$\text{score}_{word} = \cos(x, y) = \frac{x^{\top}y}{||x|| \cdot ||y||} \tag{16}$$

Second, the word analogy task investigates relations of the form "x is to y as x^* is to y^*," where y^* can be predicted given the word vectors of x, y, and x^* by 3CosAdd [41]:

$$y^* = \underset{y' \in V, y' \neq x^*, y, x}{\arg\max} \cos((x^* + y - x), y') \tag{17}$$

The relation similarity score between x to y and x^* to y^* is computed as:

$$\text{score}_{relation} = \cos((y - x), (y^* - x^*)) \tag{18}$$

Third, we can use the standard agglomerative clustering algorithm over the resulting word vectors for solving the word clustering task.

V. Experiments

We compare the effectiveness of our method with both the skip-gram baselines, the syntactic GCN method [27] and distilled embeddings from pre-trained model [26]. In addition, our methods are also compared with the state-of-the-art methods on standard benchmarks.

A. Experimental Settings

Datasets.

Training Corpus. Following Vashishth *et al.* [27], the Wikipedia dump[1] corpus is used for training static embeddings, which consist of 57 million sentences with 1.1 billion tokens. Sentences with a length between 10 to 40 are selected, the final average length of sentences is 20.2.

Intrinsic Tasks. We perform word similarity tasks on the WordSim-353 [42], SimLex-999 [43], Rare Word (RW) [44], MEN-3K [45], and RG-65 [46] datasets, computing the Spearman's rank correlation between the word similarity score_{word} and human judgments.

For word analogy, we compare the analogy prediction accuracy on the Google [6] datasets. We also compare the Spearman's rank correlation between relation similarity $\text{score}_{relation}$ and human judgments on the SemEval-2012 [47] dataset.

There has been a line of work discussing the limitations of word similarity and word analogy [48]–[51]. Following [52], [53], we also perform word concept categorization tasks which involves grouping nominal concepts into natural categories. For instance, *eat*, *breathe* and *drink* should belong to *BodyAction* class. In our experiments, we evaluate on AP [54], Battig [55] and ESSLI (including both Noun and Verb subsets) [56] datasets. Cluster purity is used as the evaluation metric.

Extrinsic Tasks. We conduct three downstream tasks by using different word embeddings, including chunking, POS tagging and NER. To directly compare the difference of word vectors, we do not fine-tune the embeddings during training.

The chunking task is evaluated on a CONLL-2000 shared task [57]. Following Reimers and Gurevych [58], we use the sections 15-18 for training, 19 for development and 20 for testing. F1-score is used as the evaluation metric.

We use OntoNotes 4.0 [59] as the named entity recognition dataset. F1-score is used as the evaluation metric.

[1] https://dumps.wikimedia.org/enwiki/20 180 301/

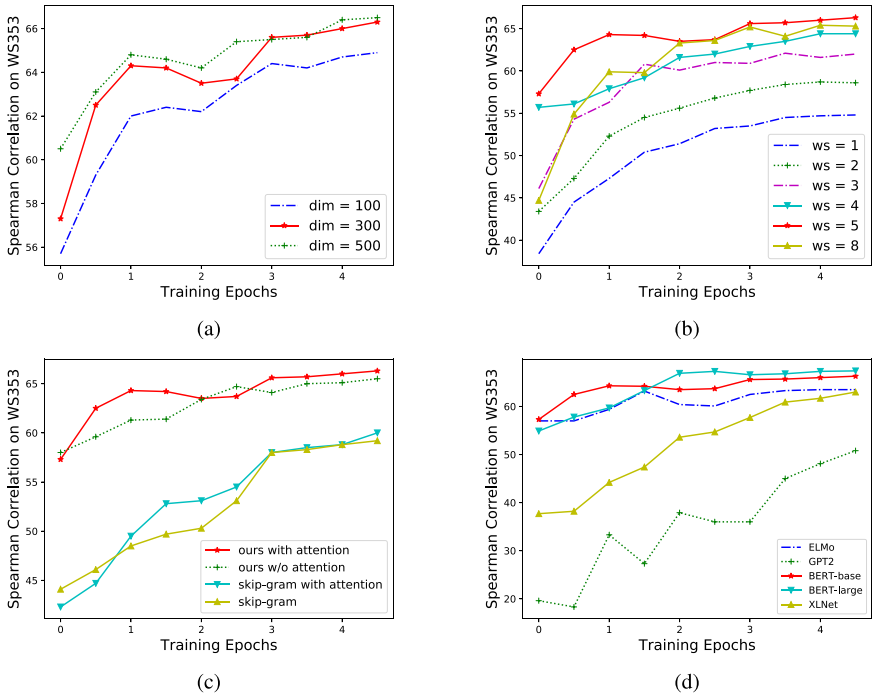


Fig. 2. Development experiments: (a) embedding dimension, (b) window size, (c) attention aggregation and (d) base models.

We use the WSJ portion of Penn Treebank [60] for POS tagging, adopting the standard splits by using sections 0-18 as the training set, 19-21 as the development set and 22-24 as the test set. Token-level accuracy is used to evaluate the performance.

For the external tasks, we adopt the BiLSTM-CRF model [61]–[63], which has been shown a strong baseline for sequence labeling.

Hyper-Parameters Settings. The dimension of word embedding vectors d_{emb} is 300, the window size for context words ws is set as 5, the number of negative samples k is 5, which are selected using development set, the initial learning rate for SGD is 0.08 and gradients are clipped at norm 5.

B. Development Experiments

We select one million sentences from the Wikipedia corpus for development experiments, investigating the effect of embedding dimension, context window size, attention aggregation and the base models for generating center word embedding.

Embedding Dimension. Fig. 2(a) shows the results for different word embedding dimension d_{emb}. The model with 100 dimensional embeddings gives a lower result, which is likely because the model underfits with too few dimensions. The model with 500 dimensions gives similar final results compared with 300 dimensions, while having more parameters and taking more training and testing time. We thus select the dimension of 300, which is the same as most existing work.

Window Size. The window size ws affects the amount of information used for explicit co-occurrence-based training, but with BERT context embeddings, each context word hidden vector contain information over the whole sentence. We compare the effect of different window sizes ranging from 1 to 8. The results are shown on Fig. 2(b). When ws is 1, we only consider the relationship between the center word embedding and its two neighbor hidden vectors. The performance is 54.8. As the window size increases, the model gives better results. However, when the window size is 8, the model costs twice as much training time but does not give further improvements compared with a window size of 5. Therefore we set the window size to 5, which is the same as skip-gram.

Attention Aggregation. Fig. 2(c) shows the results of skip-gram and our model with or without attention aggregation. Our model stably outperforms skip-gram. This shows the advantage of using BERT to represent context. Without attention aggregation, our model treats all context words equally. It gives slower convergence with a best development result of 65.5, lower than 66.3 with attention aggregation. This shows the effectiveness of differentiating context words [39].

Base Models. Fig. 2(d) shows the results of different base models for center word representation generation. Also, we compare the BERT model with different model sizes (BERT$_{base}$ (110 M) *v.s* BERT$_{large}$ (340 M)). We can find that GPT2 performs worse than the other models, which may be because it only use the unidirectional information. The BERT, XLNet and ELMo models do not show significant difference. BERT$_{large}$ performs slightly better than BERT$_{base}$ with much more parameters but costs more training time. Overall, considering both model performance and training efficiency, we use BERT$_{base}$ in our experiments.

C. Baselines

• **SG, CBOW** are the skip-gram and continuous-bag-of-words models by Mikolov *et al.* [6]. For skip-gram, a common practice is to use the center word embedding as the primary output. For fair comparison against our model, we also show the

TABLE I
MAIN RESULTS ON WORD SIMILARITY AND ANALOGY TASKS. THE ELMO, GPT2, BERT AND XLNET MODELS USE 512, 768, 768 AND 768 DIMENSIONAL EMBEDDINGS, RESPECTIVELY, WHILE OTHERS USE 300 DIMENSIONAL VECTORS. † AND ‡ INDICATE STATISTICAL SIGNIFICANCE COMPARED TO BOTH SG AND BERT$_{avg}$ MODELS ACCORDING TO [48], [64] WITH $p < 0.01$ AND $p < 0.05$, RESPECTIVELY

Types	Models	Word Similarity								Analogy	
		WS353	WS353S	WS353R	SimLex	RW	MEN	RG65	*Avg*	Google	SemEval
Static	SG	61.0	68.9	53.7	34.9	34.5	67.0	75.2	56.4	43.5	19.1
	SG(context)	53.2	60.9	43.5	32.0	28.0	58.8	69.3	49.4	40.6	16.7
	CBOW	62.7	70.7	53.9	38.0	30.0	68.6	72.7	56.6	58.4	18.9
	GloVe	54.2	64.3	50.2	31.6	29.9	68.3	61.8	51.4	45.3	18.7
	FASTTEXT	68.3	74.6	61.6	38.2	37.3	74.8	80.8	62.2	72.7	19.5
	Deps	60.6	73.1	46.8	39.6	33.0	60.5	77.1	55.8	36.0	22.9
Contextualized	ELMo$_{token}$	54.1	69.1	39.2	41.7	42.1	57.7	69.6	53.3	39.8	19.3
	GPT2$_{token}$	65.5	71.5	55.7	48.4	31.6	69.8	63.2	57.9	33.1	21.3
	BERT$_{token}$	57.8	67.3	42.5	48.9	29.5	54.8	66.1	52.4	31.7	22.0
	XLNet$_{token}$	62.4	74.4	53.2	48.1	34.0	66.3	68.3	58.1	32.6	22.2
	ELMo$_{word}$	45.5	62.1	32.4	40.6	34.6	57.2	60.9	47.6	36.4	22.6
	GPT2$_{word}$	30.7	31.4	27.6	26.4	22.5	26.2	10.6	25.1	19.9	12.5
	BERT$_{word}$	24.0	31.0	14.1	13.4	10.8	22.0	18.5	19.1	25.2	10.1
	XLNet$_{word}$	62.8	69.8	55.5	49.0	29.7	61.7	63.4	56.0	31.9	22.5
	ELMo$_{avg}$	58.3	71.3	47.4	43.6	38.4	65.5	66.8	55.9	49.1	21.2
	GPT2$_{avg}$	64.5	72.1	59.7	46.9	29.1	68.6	80.0	60.1	37.2	21.9
	BERT$_{avg}$	59.4	67.0	49.9	46.8	30.8	66.3	**81.2**	57.3	59.4	20.8
	XLNet$_{avg}$	64.9	72.3	58.0	47.3	27.7	64.1	69.7	57.1	30.8	23.2
LSTM + Static	Context2vec	63.5	66.6	57.3	39.3	23.1	66.4	72.6	55.6	60.7	20.0
GCN + Static	SynGCN	60.9	73.2	45.7	45.5	33.7	71.0	79.6	58.5	50.7	**23.4**
BERT + Static	Ours	**72.8**†	**75.3**†	**66.7**†	**49.4**‡	**42.3**†	**76.2**†	78.6	**65.9**†	**75.8**†	20.2

results from the context word embedding, which is denoted as SG(context).

• **GloVe** is a log-bilinear regression model which leverages global co-occurrence statistics of corpus [7].

• **FASTTEXT** takes into account subword information by incorporating character n-grams into the skip-gram model [65].

• **Deps** modifies the skip-gram model using dependency parse trees to replace sequential contexts [32].

• **BERT**. We investigate three ways to distill BERT [11] into static embeddings for lexical semantics tasks. The first method, called BERT$_{token}$, ignores the contextualized parameters and uses the mean pooled subword token embeddings from E in Eq. 3 as a set of static embeddings. The second method, called BERT$_{word}$, takes each single word as a complete sentence and output its word representation as a static embeddings. The third method, called BERT$_{avg}$, takes the average of output o_i in Eq. 6 over training corpus. The methods are similar to [26] but the results directly comparable with our method due to the use of the same training corpus.

• **ELMo, GPT2 and XLNet**. Similar to BERT, we also investigate the token embeddings, word embeddings and the average of output representation from ELMo [8], GPT2 [10] and XLNet [12] models. The baselines are ELMo$_{token}$, ELMo$_{word}$, ELMo$_{avg}$, GPT2$_{token}$, GPT2$_{word}$, GPT2$_{avg}$, XLNet$_{token}$, XLNet$_{word}$ and XLNet$_{avg}$, respectively.

• **Context2vec**. Melamud *et al.* learns the context embedding by using single layer bi-directional LSTM [37], the original model was trained on the two-million-word ukWaC corpus [66]. We reimplement their method and train on the same Wikipedia corpus for fair comparison.

• **SynGCN**. Given a training sentence, Vashishth *et al.* [27] use GCN to calculate context word embeddings based on the syntax structure.

The above baselines can be categorized into three classes, as shown in the first column in Table I. In particular, the first category of methods are static embeddings, where word vectors come from a lookup table. In the second category, static embeddings from contextualized word embedding models are used. In the last category, contextualized information is integrated in the training of skip-gram embeddings.

D. Results

Table I shows the main results on word similarity and analogy tasks. The models that we compare are all evaluated on the same dataset under the same settings, which means that they can be slightly different from the results reported in the original papers. In general, the models that integrate contextualized information into static embeddings performs better than the others. However, compared with LSTM and GCN, our model with BERT representation gives the best results. Overall, our model gives the best performance on 5 out of 7 datasets. In particular, it outperforms the best performing baselines by a large margin on WS353 and Google datasets, obtaining 4.5% and 3.1% absolute improvements, respectively.

We find that within WS353, our model gives much more improvements on the WS353R (relatedness) subset than the WS353S (similarity) subset [67]. In the WS353S dataset, the word pairs belong to the same semantic category. Examples include ⟨*dog*, *cat*⟩ and ⟨*money*, *dollar*⟩. We find that such words typically share the same context words explicitly, and therefore the baseline SG model can more easily differentiate them. In contrast, in the WS353R dataset, the word pairs has semantic relations rather than being synonyms. Examples include ⟨*computer*, *keyboard*⟩ and ⟨*baby*, *mother*⟩. Such words may not have the same context explicitly, although the context may have similar themes and meanings. Therefore, using BERT to represent the surrounding words gives the model better disambiguation power.

Among the static word embedding baselines, the skip-gram and CBOW models give relatively similar results. The

TABLE II
WORD CONCEPT CATEGORIZATION RESULTS

Models	AP	Batting	ESSLI(N)	ESSLI(V)	*Avg*
SG	63.4	42.8	75.0	62.2	60.8
SG(context)	57.4	41.6	72.5	66.6	59.5
CBOW	63.2	43.3	75.0	64.4	61.4
GloVe	58.0	41.3	72.5	60.0	58.0
FASTTEXT	63.4	**44.4**	75.0	62.2	61.2
Deps	61.8	41.7	77.5	**68.8**	62.4
BERT_{avg}	55.7	34.7	70.0	64.0	56.1
SynGCN	63.4	42.8	**82.5**	62.2	62.7
Ours	**64.1**	43.8	77.5	66.6	**63.0**

context embeddings perform worse than the center embeddings in skip-gram model. The FASTTEXT model gives the best result for word similarity tasks by leveraging subword information. The syntax-based embeddings Deps outperforms other static embeddings on the SemEval-2012 dataset. The reason can be that the syntax-based embedding encodes functional similarity rather than topical similarity [68], which is more suitable for the relation similarity tasks, including relation classes such as "part-whole" (e.g., ⟨*car*, *engine*⟩ is more similar to ⟨*hand*, *finger*⟩ than ⟨*bottle*, *water*⟩) and "cause-purpose" (e.g., ⟨*anesthetic*, *numbness*⟩ is more similar to ⟨*joke*, *laughter*⟩ than ⟨*smile*, *friendship*⟩).

With regard to contextualized embedding models, the static token embeddings (e.g., BERT_{token}) and the average of output representations (e.g., BERT_{avg}) outperform static methods in general. In particular, BERT_{avg} gets the best result on RG65 dataset. This shows the effectiveness of sentential information. Our method outperforms these methods for using contextualized embeddings, showing the usefulness of integrating contextualized embeddings into static embedding training based on the distributional hypothesis. The single word-based representation methods performs relatively worse than the token-based or average-based methods, especially for GPT2 and BERT models, which indicates that treating each words as a sentence is not effective for obtaining static embeddings.

Table II shows the results on word concept categorization. Similar to the results of word similarity and analogy, our model gives significant improvement over the SG and BERT_{avg} baselines, obtaining competitive results against other methods. Overall, our model gives the best averaged performance, showing the advantages of integrating both contextualized and word co-occurrence information.

Note that the results that we obtain are not the absolute best results on many datasets. For example, Rescki *et al.* [69] obtain a Spearman's rank correlation of 76.0 on the SimLex-999 dataset, and Pilehvar *et al.* [70] obtain a Spearman's rank correlation of 92.0 on the RG65 dataset. These state-of-the-art results are obtained by using different external resources, such as WORDNET and 4LANG concept dictionary for Rescki *et al.* [69] and WIKTIONARY for Pilehvar *et al.* [70], which makes direct comparison with the methods unfair. Therefore, we did not include all state-of-the-art results in Table I and II.

E. Extrinsic Results

There has been debate about the correlation between intrinsic evaluation and extrinsic evaluation [71], [72]. The general conclusion is that contextualized embeddings significantly outperform static embeddings [8], [11], although there has been recent argument that given sufficient training data static embeddings do not necessarily underperform contextualized embeddings by a large margin [25]. We conduct extrinsic evaluation on chunking, NER and POS-tagging. The results are shown in Table III. The baseline model skip-gram, the syntactic GCN baseline and the BERT_{avg} are compared. From the table it can be seen that the methods give relatively similar results, with our method outperforming the baselines in general, giving the best results on NER task, although the improvement are not significant as in the intrinsic evaluation.

TABLE III
RESULTS ON EXTRINSIC TASKS. THE BERT_{avg} MODEL USE 768 DIMENSIONAL EMBEDDINGS. SG, SYNGCN AND OURS USE 300 DIMENSIONAL EMBEDDINGS

Models	CHUNK	NER	POS	*Avg*
SG	94.29	88.76	94.66	92.57
BERT_{avg}	94.28	87.98	**95.60**	92.62
SynGCN	**94.51**	88.42	95.04	92.65
Ours	94.32	**89.02**	94.76	**92.69**

VI. ANALYSIS

Below we investigate the main reason behind the effectiveness of our method.

Fine-grained Result. Table IV shows the word similarity results of some representative word pairs. BERT_{token} and BERT_{word} do not capture the relatedness of ⟨*dividend*, *payment*⟩ and ⟨*murder*, *manslaughter*⟩ due to lack of consideration of context information, showing discrepancy between human judgement and model scoring. BERT_{avg} improves the performance. Almost all the models give better results for word pairs that have higher co-occurrence frequencies. For example, the phrase "*benchmark index*" and "*board recommendation*" appear 8 and 29 times in corpus, respectively. In addition, the same neighboring words appearing in more sentences may have more similar averaged contextualized representations, thus resulting in the fact that BERT_{avg} gives higher similarity scores compared with human judgement. SynGCN tends to underestimate the relationship between word pairs compared with other models, which shows negative influence of differentiating syntactic contexts. Overall, the results of our model are closer to human judgement.

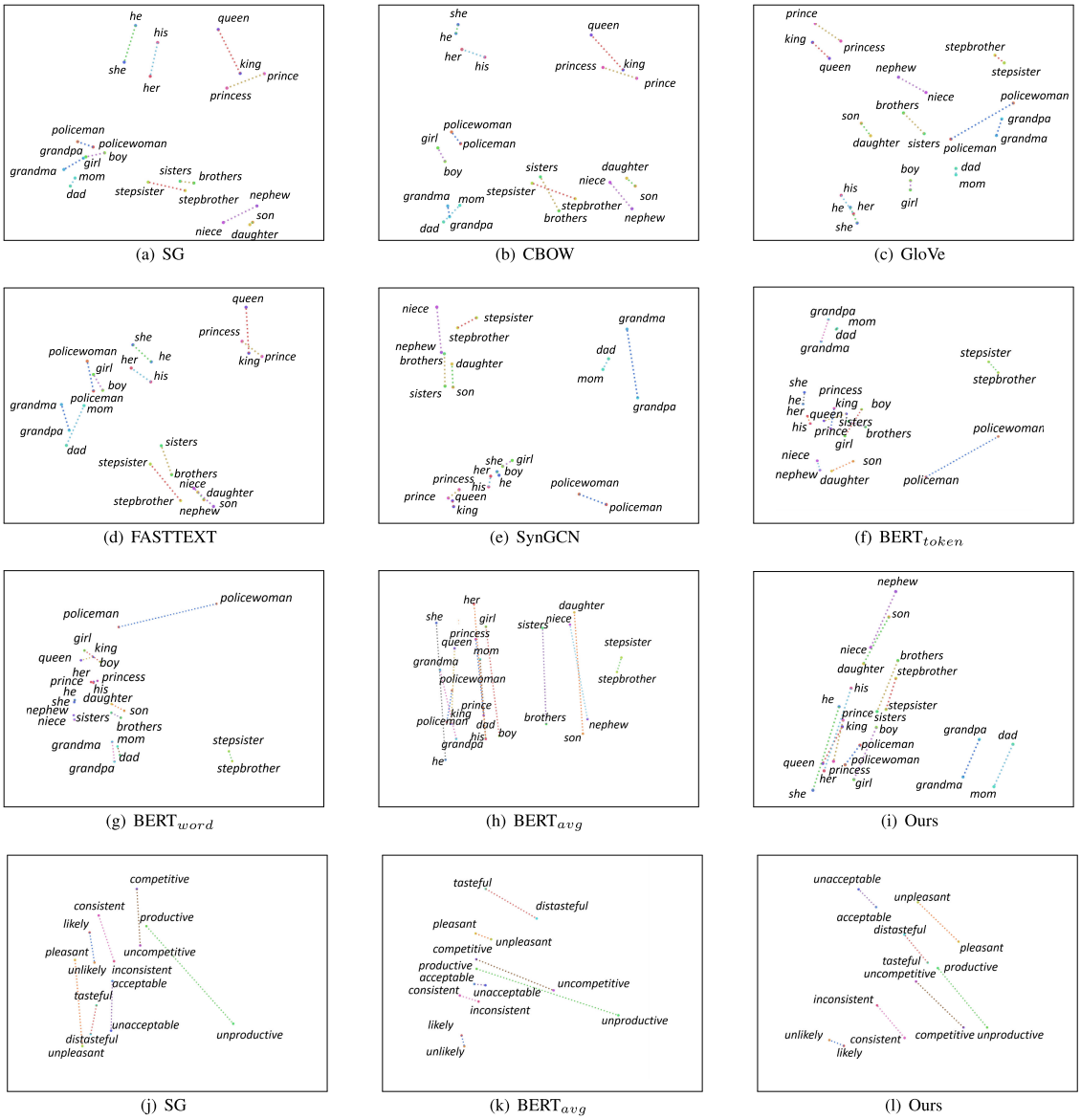
For word analogy, we compare the performance of models according to different types of word pairs. Table V shows the results. BERT_{token} and BERT_{word} perform relatively worse on "*capital-country*" and "*city-state*" compared to skip-gram because it does not model context information. BERT_{avg} improves the results by a large margin, giving comparable results on grammatical related word analogy such as "*plural*" due to the use of sentential information. SynGCN performs relatively well on grammatically related word pairs by using syntax structures. However, it does not perform as well on "*capital-country*" and "*nationality-adjective*" compared with the sequential context based skip-gram model. In contrast, our model takes the advantages of both syntactic and semantic patterns by using BERT, and gives the best overall performance.

TABLE IV
WORD SIMILARITY COMPARISON BETWEEN HUMAN AND MODELS. THE SCORES OF HUMAN ARE NORMALIZED TO (0,1). THE NUMBERS IN THE PARENTHESES DENOTE THE DIFFERENCE OF COSINE VALUES AND HUMAN JUDGEMENTS

Word Pairs	Human	SG	BERT$_{token}$	BERT$_{word}$	BERT$_{avg}$	SynGCN	Ours
dividend, *payment*	0.763	0.464 (-0.29)	0.347 (-0.41)	0.551 (-0.21)	0.503 (-0.26)	0.431 (-0.33)	**0.566 (-0.19)**
murder, *manslaughter*	0.853	0.600 (-0.25)	0.369 (-0.48)	0.287 (-0.56)	0.672 (-0.18)	0.516 (-0.33)	**0.712 (-0.14)**
shower, *thunderstorm*	0.631	0.401 (-0.23)	0.344 (-0.28)	0.771 (+0.14)	0.483 (-0.14)	0.398 (-0.23)	**0.496 (-0.13)**
board, *recommendation*	0.447	0.259 (-0.18)	0.299 (-0.14)	0.857 (+0.41)	0.583 (+0.13)	0.092 (-0.35)	**0.342 (-0.10)**
benchmark, *index*	0.425	0.305 (-0.12)	0.247 (-0.17)	0.759 (+0.33)	0.569 (+0.14)	0.255 (-0.17)	**0.435 (-0.01)**

TABLE V
WORD ANALOGY PREDICTION ACCURACY ON GOOGLE DATASETS ACCORDING TO DIFFERENT TYPES OF WORD PAIRS

Types	Example	SG	BERT$_{token}$	BERT$_{word}$	BERT$_{avg}$	SynGCN	Ours
capital-country	*Berlin* to *Germany* is *Ottawa* to *Canada*	59.7	17.2	31.3	45.3	51.3	**86.7**
city-state	*Phoenix* to *Arizona* is *Dallas* to *Texas*	39.2	16.2	22.3	36.2	38.4	**70.8**
nationality-adj	*Austria* to *Austrian* is *Spain* to *Spanish*	67.3	69.3	44.9	87.9	40.1	**90.3**
family	*son* to *daughter* is *uncle* to *aunt*	63.6	41.5	24.5	76.6	69.5	**86.7**
comparative	*good* to *better* is *easy* to *easier*	53.4	55.2	19.2	80.4	78.6	**91.7**
superlative	*fast* to *fastest* is *bad* to *worst*	23.8	41.6	19.4	58.0	45.5	**85.9**
plural	*dog* to *dogs* is *mouse* to *mice*	38.5	28.3	38.8	90.6	74.7	**92.2**

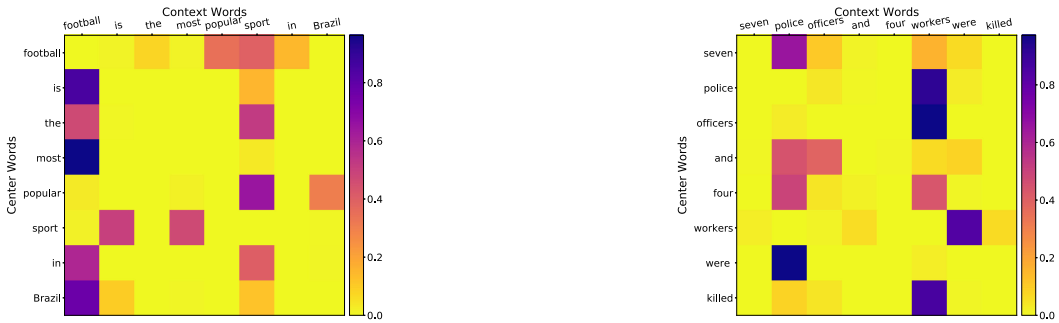


(a) SG (b) CBOW (c) GloVe
(d) FASTTEXT (e) SynGCN (f) BERT$_{token}$
(g) BERT$_{word}$ (h) BERT$_{avg}$ (i) Ours
(j) SG (k) BERT$_{avg}$ (l) Ours

Fig. 3. Visualization of word pairs with the *male-female* (a-i) and *positive-negative* (j-l) relationship.

TABLE VI
NEAREST NEIGHBORS FOR WORDS *"LIGHT"* AND *"WHILE"*

Models	Nearest Neighbors for Word *"light"*	Nearest Neighbors for Word *"while"*
SG	*uv*, *bioluminescence*, *fluorescent*, *glare*, *sunlight*, *illumination*	*whilst*, *recuperating*, *pursuing*, *preparing*, *attempting*, *fending*
CBOW	*stevenson*, *intimidation*, *earle*, *yellowing*, *row*, *kizer*	*whilst*, *when*, *still*, *although*, *and*, *but*
GloVe	*excluding*, *justify*, *orestes*, *generation*, *energy*, *frieze*	*both*, *taking*, *','*, *up*, *but*, *after*
FASTTEXT	*sculpts*, *baha'i*, *kinghorn*, *lick*, *inputs*, *minimize*	*whilst*, *still*, *and*, *meanwhile*, *instead*, *though*
SynGCN	*search*, *prostějov*, *preceding*, *forearms*, *freewheel*, *naxos*	*whilst*, *time*, *when*, *years*, *months*, *tenures*
Ours	*lights*, *dark*, *lighter*, *illumination*, *glow*, *illuminating*	*whilst*, *whereas*, *although*, *conversely*, *though*, *meanwhile*



(a) football is the most popular sport in Brazil

(b) seven police officers and four workers were killed

Fig. 4. Attention distribution visualization of the sentences.

Word Pairs Visualization. Fig. 3(a)-(i) shows the t-SNE [73] visualization results for word pairs with the *male-female* relationship. For example, the pronoun pair $\langle he, she \rangle$, the occupation pair $\langle policeman, policewoman \rangle$ and the family relation pair $\langle grandpa, grandma \rangle$ all differ only by the gender characteristics. In particular, the skip-gram, CBOW, GloVe, FASTTEXT and SynGCN baselines all capture the gender analogy through vector space topology to some extent. However, inconsistency exists between different word pairs. For BERT based vectors, BERT_{avg} performs better than the others, which shows the importance of contextualized information. Overall, the outputs of our method are highly consistent, better demonstrating the algebraic motivation behind skip-gram embeddings compared with the fully-static skip-gram algorithm. This demonstrates the effect of contextualized embeddings in better representing semantic information.

Given that different words can occur with different frequency in the corpus, we also show more results for words with *positive-negative* relationship in Fig. 3(j)-(l). Similar to the findings in previous word pairs, the results become better form SG to BERT_{avg} and our model, which shows the advantages of our model again.

Nearest Neighbors. Table VI shows the nearest neighbors to the words *"light"* and *"while"* according to cosine similarity. In particular, for the noun "*light*," static embeddings yield many noise and unrelated words such as "*uv*," "*stevenson*," "*excluding*," "*baha'i*" and "*prostějov*," which may occur in the context of "*light*". In contrast, our method captures the main meaning and generates cleaner results.

For word "*while*," static methods yield words that tend to co-occur with the word "*while*," such as "*preparing*," "*still*," "*taking*" and "*instead*". In contrast, SynGCN returns words that are semantically similar, namely those that are related to time. In contrast with the baselines, our method returns multiple conjunctions that have similar meanings to "*while*," such as "*whilst*," "*whereas*" and "*although*," which better conforms to the intuition, demonstrating the advantage of using contextualized to address word sense ambiguities.

Attention Distribution Visualization. Fig. 4(a) and Fig. 4(b) show the attention weights in Eq. 13 when different words are used as the center word for the sentences "*football is the most popular sport in Brazil*" and "*seven police officers and four workers were killed*". As expected, for each center word, the most relevant context word receives relatively more attention. For example, the word "*football*" is more associated with the words "*popular*" and "*sport*," the word "*the*" is more associated with nouns, and the nouns "*police officers*" and "*workers*" get the most attentions. No word pays attention to the word "*the*" in the context words, which is a stop word, and the words "*seven*," "*and*" and "*four*" get the least attentions, which are numeral-measure words and conjunction.

VII. CONCLUSION

We investigated how to make use of BERT embedding for better training skip-gram embeddings. Compared with recent work using GCN to the same end, we show that BERT serving as context embeddings give a balance between syntactic and surface features, selecting useful context more effectively. Our method gives the best results on a range of benchmarks. Future work includes the investigation of sense embeddings and syntactic embeddings under our framework.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers, in particular the reviewer 3 in the first round, for the detailed

and constructive suggestions. They were highly motivating for reaching a final version of this article.

REFERENCES

[1] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman, "Indexing by latent semantic analysis," *J. Amer. Soc. Inf. Sci.*, vol. 41, no. 6, pp. 391–407, Sep. 1990.

[2] Y. Bengio, R. Ducharme, P. Vincent, and C. Janvin, "A neural probabilistic language model," *J. Mach. Learn. Res.*, vol. 3, pp. 1137–1155, Mar. 2003.

[3] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in *Proc. 25th Int. Conf. Mach. Learn.*, 2008, pp. 160–167.

[4] A. Mnih and G. Hinton, "A scalable hierarchical distributed language model," in *Proc. 21st Int. Conf. Neural Inf. Process. Syst.*, 2008, pp. 1081–1088.

[5] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural language processing (almost) from scratch," *J. Mach. Learn. Res.*, vol. 12, pp. 2493–2537, Nov. 2011.

[6] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," [Online]. Available: http://dblp.uni-trier.de/db/journals/corr/corr1301.html#abs-1301-3781

[7] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2014, pp. 1532–1543.

[8] M. Peters *et al.*, "Deep contextualized word representations," in *NAACL*, New Orleans, LA, USA, Jun. 2018, pp. 2227–2237.

[9] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving language understanding by generative pre-training," 2018.

[10] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," 2019. [Online]. Available: https://github.com/openai/gpt-2/issues/79

[11] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *NAACL*, Minneapolis, MN, USA, Jun. 2019, pp. 4171–4186.

[12] Z. Yang *et al.*, "XlNet: Generalized autoregressive pretraining for language understanding," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, vol. 32.

[13] Y. Goldberg, "Assessing BERT's syntactic abilities," 2019, *arXiv:1901.05287*.

[14] G. Jawahar, B. Sagot, and D. Seddah, "What does BERT learn about the structure of language?," in *ACL*, Florence, Italy, Jul. 2019, pp. 3651–3657.

[15] F. Petroni *et al.*, "Language models as knowledge bases?," in *EMNLP-IJCNLP*, 2019, pp. 2463–2473.

[16] E. Choi *et al.*, "QuAC: Question answering in context," in *Proc. Conf. Empirical Methods Natural Lang. Process.* Brussels, Belgium, Oct./Nov. 2018, pp. 2174–2184.

[17] H. Xu, B. Liu, L. Shu, and P. Yu, "BERT post-training for review reading comprehension and aspect-based sentiment analysis," in *NAACL*, Jun. 2019, pp. 2324–2335.

[18] B. Y. Lin, X. Chen, J. Chen, and X. Ren, "KagNet: Knowledge-aware graph networks for commonsense reasoning," in *Proc. Conf. Empirical Methods Natural Lang. Process. 9th Int. Joint Conf. Natural Lang. Process.*, Hong Kong, China, Nov. 2019, pp. 2822–2832.

[19] N. Jiang and M.-C. De Marneffe, "Evaluating BERT for natural language inference: A case study on the Commitment Bank," in *9th Int. Joint Conf. Nat. Lang. Process.*, Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 6086–6091.

[20] K. Ethayarajh, "How contextual are contextualized word representations? Comparing the geometry of BERT, ELMo, and GPT-2 embeddings," in *Proc. 9th Int. Joint Conf. Natural Lang. Process.*, Hong Kong, China, Nov. 2019, pp. 55–65.

[21] R. T. McCoy, J. Min, and T. Linzen, "BERTs of a feather do not generalize together: Large variability in generalization across models with similar test set performance," in *Proc. 3rd BlackboxNLP Workshop Analyzing Interpreting Neural Netw. NLP*, Nov. 2020, pp. 217–227.

[22] A. Stiff, Q. Song, and E. Fosler-Lussier, "How self-attention improves rare class performance in a question-answering dialogue agent," in *Proc. 21th Annu. Meeting Special Int. Group Discourse Dialogue*, Jul. 2020, pp. 196–202.

[23] X. Li, G. Li, L. Liu, M. Meng, and S. Shi, "On the word alignment from neural machine translation," in *Proc. 57th Annu. Meeting Assoc. Comput. Linguistics*. Florence, Italy, Jul. 2019, pp. 1293–1303.

[24] S. Ding, H. Xu, and P. Koehn, "Saliency-driven word alignment interpretation for neural machine translation," in *Proc. 4th Conf. Mach. Trans., Volume 1: Res. Papers*, Florence, Italy, Aug. 2019, pp. 1–12.

[25] S. Arora, A. May, J. Zhang, and C. Ré, "Contextual embeddings: When are they worth it?," in *Proc. 58th Annu. Meeting Assoc. Comput. Linguistics*. Online: Association for Computational Linguistics, Jul. 2020, pp. 2650–2663.

[26] R. Bommasani, K. Davis, and C. Cardie, "Interpreting pretrained contextualized representations via reductions to static embeddings," in *Proc. 58th Annu. Meeting Assoc. Comput. Linguistics*, Jul. 2020, pp. 4758–4781.

[27] S. Vashishth, M. Bhandari, P. Yadav, P. Rai, C. Bhattacharyya, and P. Talukdar, "Incorporating syntactic and semantic information in word embeddings using graph convolutional networks," in *ACL*, Florence, Italy, Jul. 2019, pp. 3308–3318.

[28] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. Int. Conf. Learn. Representations*, 2017.

[29] E. Reif, "Visualizing and measuring the geometry of BERT, " in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, vol. 32.

[30] W. Ling, C. Dyer, A. W. Black, and I. Trancoso, "Two/too simple adaptations of Word2Vec for syntax problems," in *NAACL*, Denver, CO, USA, May/Jun. 2015, pp. 1299–1304.

[31] D. Hall, G. Durrett, and D. Klein, "Less grammar, more features," in *ACL*, Baltimore, MD, USA, Jun. 2014, pp. 228–237.

[32] O. Levy and Y. Goldberg, "Dependency-based word embeddings," in *ACL*, Baltimore, MD, USA, Jun. 2014, pp. 302–308.

[33] A. Vaswani *et al.*, "Attention is all you need," in *NeurIPS*, 2017, pp. 5998–6008.

[34] Y. Liu *et al.*, "RoBERTa: A robustly optimized BERT pretraining approach," 2019, *arXiv:1907.11692*.

[35] I. Beltagy, M. E. Peters, and A. Cohan, "Longformer: The long-document transformer," 2020, *arXiv:2004.05150*.

[36] Z. Dai, Z. Yang, Y. Yang, J. Carbonell, Q. Le, and R. Salakhutdinov, "Transformer-XL: Attentive language models beyond a fixed-length context," in *ACL*, Florence, Italy, Jul. 2019, pp. 2978–2988.

[37] O. Melamud, J. Goldberger, and I. Dagan, "Context2Vec: Learning generic context embedding with bidirectional LSTM," in *Proc. 20th SIGNLL Conf. Comput. Natural Lang. Learn.*, Berlin, Germany, Aug. 2016, pp. 51–61.

[38] Y. Wu *et al.*, "Google's neural machine translation system: Bridging the gap between human and machine translation," 2016, *arXiv:1609.08144*.

[39] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *NeurIPS*, C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, Eds., 2013, pp. 3111–3119.

[40] T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, Lisbon, Portugal, Sep. 2015, pp. 1412–1421.

[41] O. Levy and Y. Goldberg, "Linguistic regularities in sparse and explicit word representations," in *Proc. 18th Conf. Comput. Natural Lang. Learn.* Ann Arbor, MI, USA: Jun. 2014, pp. 171–180.

[42] L. Finkelstein *et al.*, "Placing search in context: The concept revisited," *TOIS*, vol. 20, pp. 406–414, 2001.

[43] D. Kiela, F. Hill, and S. Clark, "Specializing word embeddings for similarity or relatedness," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, Lisbon, Portugal, Sep. 2015, pp. 2044–2048.

[44] T. Luong, R. Socher, and C. Manning, "Better word representations with recursive neural networks for morphology," in *Proc. 18th Conf. Comput. Natural Lang. Learn.*, Sofia, Bulgaria: Aug. 2013, pp. 104–113.

[45] E. Bruni, G. Boleda, M. Baroni, and N.-K. Tran, "Distributional semantics in technicolor," in *ACL*, Jeju Island, Korea, Jul. 2012, pp. 136–145.

[46] H. Rubenstein and J. Goodenough, "Contextual correlates of synonymy," *Commun. ACM*, vol. 8, no. 10, pp. 627–633, 1965.

[47] D. Jurgens, S. Mohammad, P. Turney, and K. Holyoak, "SemEval-2012 task 2: Measuring degrees of relational similarity," in *SEMEVAL*, Montréal, Canada, 7–8 Jun. 2012, pp. 356–364.

[48] M. Faruqui, Y. Tsvetkov, P. Rastogi, and C. Dyer, "Problems with evaluation of word embeddings using word similarity tasks," in *Proc. 1st Workshop Evaluating Vector-Space Representations NLP*, Berlin, Germany, Aug. 2016, pp. 30–35.

[49] A. Gladkova, A. Drozd, and S. Matsuoka, "Analogy-based detection of morphological and semantic relations with word embeddings: What works and what doesn't," in *Proc. NAACL Student Res. Workshop*, San Diego, CA, USA, Jun. 2016, pp. 8–15.

[50] A. Rogers, A. Drozd, and B. Li, "The (too many) problems of analogical reasoning with word vectors," in *Proc. 6th Joint Conf. Lexical Comput. Semantics*, Vancouver, Canada, Aug. 2017, pp. 135–148.
[51] D. Newman-Griffis, A. Lai, and E. Fosler-Lussier, "Insights into analogy completion from the biomedical domain," in *BioNLP*, Vancouver, Canada, Aug. 2017, pp. 19–28.
[52] A. Rogers, S. H. Ananthakrishna, and A. Rumshisky, "What's in your embedding, and how it predicts task performance," in *Proc. 27th Int. Conf. Comput. Linguistics*, Santa Fe, NM, USA, Aug. 2018, pp. 2690–2703.
[53] B. Whitaker, D. Newman-Griffis, A. Haldar, H. Ferhatosmanoglu, and E. Fosler-Lussier, "Characterizing the impact of geometric properties of word embeddings on task performance," in *Proc. 3rd Workshop Evaluating Vector Space Representations NLP*, 2019, pp. 8–17.
[54] A. Almuhareb, "Attributes in lexical acquisition," Univ. Essex, 2006. [Online]. Available: https://books.google.com.hk/books?id=_I0YHAAACAAJ
[55] M. Baroni and A. Lenci, "Distributional memory: A general framework for corpus-based semantics," *Comput. Linguistics*, vol. 36, no. 4, pp. 673–721, 2010.
[56] M. Baroni, S. Evert, and A. Lenci, "Esslli workshop on distributional lexical semantics bridging the gap between semantic theory and computational simulations," in *Assoc. Logic, Lang. Inf.*, 2008.
[57] E. F. T. K. Sang and S. Buchholz, "Introduction to the CoNLL-2000 shared task chunking," in *Proc. 18th Conf. Comput. Natural Lang. Learn.*, 2000. [Online]. Available: https://www.aclweb.org/anthology/W00-0726
[58] N. Reimers and I. Gurevych, "Reporting score distributions makes a difference: Performance study of LSTM-networks for sequence tagging," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, Copenhagen, Denmark, Sep. 2017, pp. 338–348.
[59] R. Weischedel *et al.*, "Ontonotes release 4.0," LDC2011T03, Philadelphia, PA, USA, 2011.
[60] M. P. Marcus, M. A. Marcinkiewicz, and B. Santorini, "Building a large annotated corpus of English: The penn treebank," *Comput. Linguistics*, vol. 19, no. 2, pp. 313–330, Jun. 1993.
[61] G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, and C. Dyer, "Neural architectures for named entity recognition," in *NAACL*, San Diego, CA, USA, Jun. 2016, pp. 260–270.
[62] X. Ma and E. Hovy, "End-to-end sequence labeling via bi-directional lstm-cnns-crf," in *ACL*, Berlin, Germany, Aug. 2016, pp. 1064–1074.
[63] J. Yang, S. Liang, and Y. Zhang, "Design challenges and misconceptions in neural sequence labeling," in *COLING*, 2018, pp. 3879–3889. [Online]. Available: https://aclanthology.info/papers/C18-1327/c18-1327
[64] P. Rastogi, B. Van Durme, and R. Arora, "Multiview LSA: Representation learning via generalized CCA," in *Proc. 2015 Conf. North Amer. Chapter Assoc. Comput. Linguistics: Hum. Lang. Technol.*, Denver, CO, USA, May/Jun. 2015, pp. 556–566.
[65] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *Trans. Assoc. Comput. Linguistics*, vol. 5, pp. 135–146, 2017.
[66] A. Ferraresi, E. Zanchetta, M. Baroni, and S. Bernardini, "Introducing and evaluating ukwac, a very large web-derived corpus of English," in *Proc. 4th Web Corpus Workshop (WAC-4) Can we beat Google*, 2008, pp. 47–54.
[67] E. Agirre, E. Alfonseca, K. Hall, J. Kravalova, M. Paşca, and A. Soroa, "A study on similarity and relatedness using distributional and WordNet-based approaches," in *Proc. Hum. Lang. Technol.: 2009 Annu. Conf. North Amer. Chapter Assoc. Comput. Linguistics*, Boulder, CO, USA, Jun. 2009, pp. 19–27.
[68] A. Komninos and S. Manandhar, "Dependency based embeddings for sentence classification tasks," in *NAACL*, San Diego, CA, USA, Jun. 2016, pp. 1490–1500.
[69] G. Recski, E. Iklódi, K. Pajkossy, and A. Kornai, "Measuring semantic similarity of words using concept networks," in *Proc. 1st Workshop Representation Learn. NLP*, Berlin, Germany, Aug. 2016, pp. 193–200.
[70] M. T. Pilehvar and R. Navigli, "From senses to texts: An all-in-one graph-based approach for measuring semantic similarity," *Artif. Intell.*, vol. 228, pp. 95–128, 2015.
[71] B. Chiu, A. Korhonen, and S. Pyysalo, "Intrinsic evaluation of word vectors fails to predict extrinsic performance," in *Proc. 1st Workshop Evaluating Vector-Space Representations NLP*, Berlin, Germany, Aug. 2016, pp. 1–6.
[72] A. Conneau, G. Kruszewski, G. Lample, L. Barrault, and M. Baroni, "What you can cram into a single $&!#* vector: Probing sentence embeddings for linguistic properties," in *Proc. 56th Annu. Meeting Assoc. Comput. Linguistics Vol. 1, Long Papers*, Melbourne, Australia, Jul. 2018, pp. 2126–2136.
[73] L. V. D. Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, no. 86, pp. 2579–2605, 2008.

Yile Wang received the bachelor's and the master's degree in optical engineering from Zhejiang University, Hangzhou, China, in 2013 and 2016, respectively. He is currently working toward the Ph.D. degree with Westlake University, Hangzhou, China. From 2016 to 2018, he was a Software Development Engineer with Huawei, Hangzhou, China. His current research interests include language modeling, sequence labeling, and machine learning.

Leyang Cui received the bachelor's degree from Sichuan University, Chengdu, China, in 2017 and the master's degree from the National University of Singapore, Singapore, in 2018. He is currently working toward the Ph.D. degree with Westlake University, Hangzhou, China. His current research interests include language modeling, sequence labeling, and open domain chatbot.

Yue Zhang is currently an Associate Professor with Westlake University, Hangzhou, China. He has been working on statistical parsing, text synthesis, natural language synthesis, machine translation, information extraction, sentiment analysis, and stock market analysis. His research interests include natural language processing and computational finance. He was the recipient of the Best Paper Awards of IALP 2017 and COLING 2018. He is on the editorial board, as the Action Editor of the *Transaction of Association of Computational Linguistics*, as an Associate Editor for *ACM Transactions on Asian and Low Resource Language Information Processing* and IEEE TRANSACTIONS ON BIG DATA, and as the Area Chairs of COLING 2014/18, NAACL 2015/19, EMNLP 2015/17/19, and ACL 2017/18/19. He gave conference tutorials at NAACL 2010, ACL 2014, and EMNLP 2016/18.