KG-BERT: BERT for Knowledge Graph Completion

Liang Yao, Chengsheng Mao, Yuan Luo*
Northwestern University
Chicago IL 60611
{liang.yao, chengsheng.mao, yuan.luo}@northwestern.edu

Abstract

Knowledge graphs are important resources for many artificial intelligence tasks but often suffer from incompleteness. In this work, we propose to use pre-trained language models for knowledge graph completion. We treat triples in knowledge graphs as textual sequences and propose a novel framework named Knowledge Graph Bidirectional Encoder Representations from Transformer (KG-BERT) to model these triples. Our method takes entity and relation descriptions of a triple as input and computes scoring function of the triple with the KG-BERT language model. Experimental results on multiple benchmark knowledge graphs show that our method can achieve state-of-the-art performance in triple classification, link prediction and relation prediction tasks.

Introduction

Large-scale knowledge graphs (KG) such as FreeBase (Bollacker et al. 2008), YAGO (Suchanek, Kasneci, and Weikum 2007) and WordNet (Miller 1995) provide effective basis for many important AI tasks such as semantic search, recommendation (Zhang et al. 2016) and question answering (Cui et al. 2017). A KG is typically a multi-relational graph containing entities as nodes and relations as edges. Each edge is represented as a triplet (*head entity*, relation, *tail entity*) ((h, r, t) for short), indicating the relation between two entities, e.g., (*Steve Jobs*, founded, *Apple Inc.*). Despite their effectiveness, knowledge graphs are still far from being complete. This problem motivates the task of *knowledge graph completion*, which is targeted at assessing the plausibility of triples not present in a knowledge graph.

Much research work has been devoted to knowledge graph completion. A common approach is called knowledge graph embedding which represents entities and relations in triples as real-valued vectors and assess triples' plausibility with these vectors (Wang et al. 2017). However, most knowledge graph embedding models only use structure information in observed triple facts, which suffer from the sparseness of knowledge graphs. Some recent studies incorporate textual information to enrich knowledge representation (Socher et al. 2013; Xie et al. 2016; Xiao et al. 2017), but they learn unique text embedding for the same entity/relation in different triples, which ignore contextual information. For instance, different words in the description of *Steve Jobs* should have distinct importance weights connected to two relations "founded" and "isCitizenOf", the relation "wroteMusicFor" can have two different meanings "writes lyrics" and "composes musical compositions" given different entities. On the other hand, syntactic and semantic information in large-scale text data is not fully utilized, as they only employ entity descriptions, relation mentions or word co-occurrence with entities (Wang and Li 2016; Xu et al. 2017; An et al. 2018).

Recently, pre-trained language models such as ELMo (Peters et al. 2018), GPT (Radford et al. 2018), BERT (Devlin et al. 2019) and XLNet (Yang et al. 2019) have shown great success in natural language processing (NLP), these models can learn contextualized word embeddings with large amount of free text data and achieve state-of-the-art performance in many language understanding tasks. Among them, BERT is the most prominent one by pre-training the bidirectional Transformer encoder through masked language modeling and next sentence prediction. It can capture rich linguistic knowledge in pre-trained model weights.

In this study, we propose a novel method for knowledge graph completion using pre-trained language models. Specifically, we first treat entities, relations and triples as textual sequences and turn knowledge graph completion into a sequence classification problem. We then fine-tune BERT model on these sequences for predicting the plausibility of a triple or a relation. The method can achieve strong performance in several KG completion tasks. Our source code is available at https://github.com/yao8839836/kg-bert. Our contributions are summarized as follows:

- We propose a new language modeling method for knowledge graph completion. To the best of our knowledge, this is the first study to model triples' plausibility with a pre-trained contextual language model.
- Results on several benchmark datasets show that our method can achieve state-of-the-art results in triple classification, relation prediction and link prediction tasks.

*Corresponding Author

Related Work

Knowledge Graph Embedding

A literature survey of knowledge graph embedding methods has been conducted by (Wang et al. 2017). These methods can be classified into translational distance models and semantic matching models based on different scoring functions for a triple (h, r, t). Translational distance models use distance-based scoring functions. They assess the plausibility of a triple (h, r, t) by the distance between the two entity vectors $\mathbf{h}$ and $\mathbf{t}$, typically after a translation performed by the relation vector $\mathbf{r}$. The representative models are TransE (Bordes et al. 2013) and its extensions including TransH (Wang et al. 2014b). For TransE, the scoring function is defined as the negative translational distance $f(h, r, t) = -||\mathbf{h} + \mathbf{r} - \mathbf{t}||$. Semantic matching models employ similarity-based scoring functions. The representative models are RESCAL (Nickel, Tresp, and Kriegel 2011), DistMult (Yang et al. 2015) and their extensions. For DistMult, the scoring function is defined as a bilinear function $f(h, r, t) = \langle \mathbf{h}, \mathbf{r}, \mathbf{t} \rangle$. Recently, convolutional neural networks also show promising results for knowledge graph completion (Dettmers et al. 2018; Nguyen et al. 2018a; Schlichtkrull et al. 2018).

The above methods conduct knowledge graph completion using only structural information observed in triples, while different kinds of external information like entity types, logical rules and textual descriptions can be introduced to improve the performance (Wang et al. 2017). For textual descriptions, (Socher et al. 2013) firstly represented entities by averaging the word embeddings contained in their names, where the word embeddings are learned from an external corpus. (Wang et al. 2014a) proposed to jointly embed entities and words into the same vector space by aligning Wikipedia anchors and entity names. (Xie et al. 2016) use convolutional neural networks (CNN) to encode word sequences in entity descriptions. (Xiao et al. 2017) proposed semantic space projection (SSP) which jointly learns topics and KG embeddings by characterizing the strong correlations between fact triples and textual descriptions. Despite their success, these models learn the same textual representations of entities and relations while words in entity/relation descriptions can have different meanings or importance weights in different triples.

To address the above problems, (Wang and Li 2016) presented a text-enhanced KG embedding model TEKE which can assign different embeddings to a relation in different triples. TEKE utilizes co-occurrences of entities and words in an entity-annotated text corpus. (Xu et al. 2017) used an LSTM encoder with attention mechanism to construct contextual text representations given different relations. (An et al. 2018) proposed an accurate text-enhanced KG embedding method by exploiting triple specific relation mentions and a mutual attention mechanism between relation mention and entity description. Although these methods can handle the semantic variety of entities and relations in distinct triples, they could not make full use of syntactic and semantic information in large scale free text data, as only entity descriptions, relation mentions and word co-occurrence with entities are utilized. Compared with these methods, our method can learn context-aware text embeddings with rich language information via pre-trained language models.

Language Model Pre-training

Pre-trained language representation models can be divided into two categories: feature-based and fine tuning approaches. Traditional word embedding methods such as Word2Vec (Mikolov et al. 2013) and Glove (Pennington, Socher, and Manning 2014) aimed at adopting feature-based approaches to learn context-independent words vectors. ELMo (Peters et al. 2018) generalized traditional word embeddings to context-aware word embeddings, where word polysemy can be properly handled. Different from feature-based approaches, fine tuning approaches like GPT (Radford et al. 2018) and BERT (Devlin et al. 2019) used the pre-trained model architecture and parameters as a starting point for specific NLP tasks. The pre-trained models capture rich semantic patterns from free text. Recently, pre-trained language models have also been explored in the context of KG. (Wang, Kulkarni, and Wang 2018) learned contextual embeddings on entity-relation chains (sentences) generated from random walks in KG, then used the embeddings as initialization of KG embeddings models like TransE. (Zhang et al. 2019) incorporated informative entities in KG to enhance BERT language representation. (Bosselut et al. 2019) used GPT to generate tail phrase tokens given head phrases and relation types in a common sense knowledge base which does not cleanly fit into a schema comparing two entities with a known relation. The method focuses on generating new entities and relations. Unlike these studies, we use names or descriptions of entities and relations as input and fine-tune BERT to compute plausibility scores of triples.

Method

Bidirectional Encoder Representations from Transformers (BERT)

BERT (Devlin et al. 2019) is a state-of-the-art pre-trained contextual language representation model built on a multi-layer bidirectional Transformer encoder (Vaswani et al. 2017). The Transformer encoder is based on self-attention mechanism. There are two steps in BERT framework: *pre-training* and *fine-tuning*. During pre-training, BERT is trained on large-scale unlabeled general domain corpus (3,300M words from BooksCorpus and English Wikipedia) over two self-supervised tasks: masked language modeling and next sentence prediction. In masked language modeling, BERT predicts randomly masked input tokens. In next sentence prediction, BERT predicts whether two input sentences are consecutive. For fine-tuning, BERT is initialized with the pre-trained parameter weights, and all of the parameters are fine-tuned using labeled data from downstream tasks such as sentence pair classification, question answering and sequence labeling.

Knowledge Graph BERT (KG-BERT)

To take full advantage of contextual representation with rich language patterns, We fine tune pre-trained BERT for

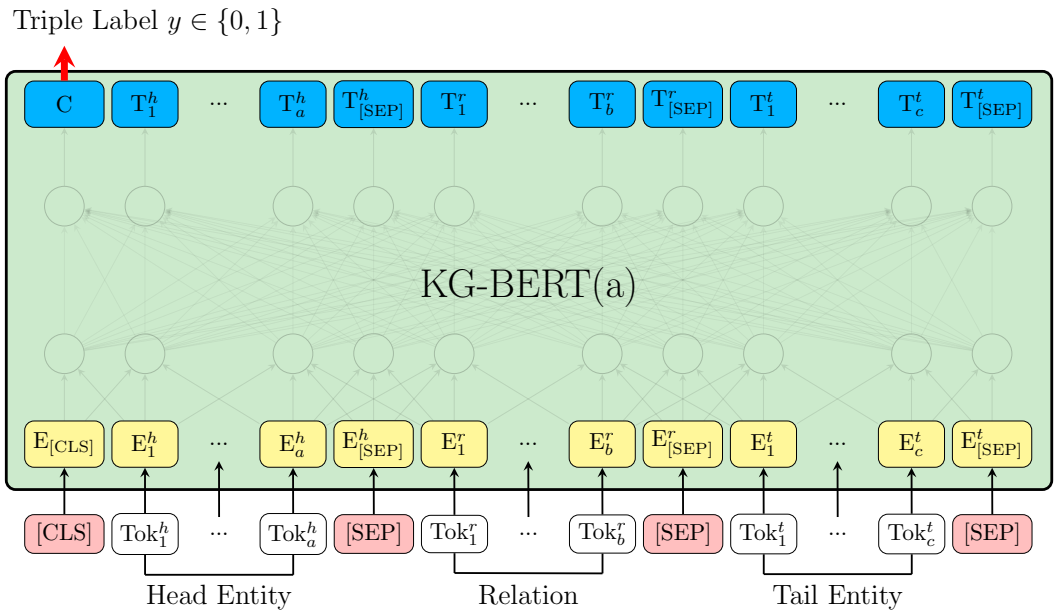


Figure 1: Illustrations of fine-tuning KG-BERT for predicting the plausibility of a triple.

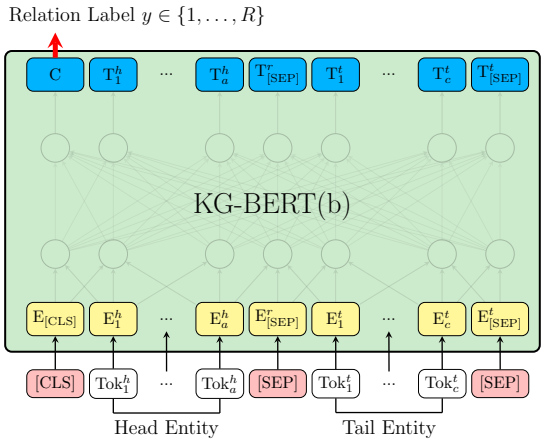


Figure 2: Illustrations of fine-tuning KG-BERT for predicting the relation between two entities.

knowledge graph completion. We represent entities and relations as their names or descriptions, then take the name/description word sequences as the input sentence of the BERT model for fine-tuning. As original BERT, a "sentence" can be an arbitrary span of contiguous text or word sequence, rather than an actual linguistic sentence. To model the plausibility of a triple, we packed the sentences of (h, r, t) as a single sequence. A sequence means the input token sequence to BERT, which may be two entity name/description sentences or three sentences of (h, r, t) packed together.

The architecture of the KG-BERT for modeling triples is shown in Figure 1. We name this KG-BERT version KG-BERT(a). The first token of every input sequence is always a special classification token [CLS]. The head entity is represented as a sentence containing tokens Tok_1^h, ..., Tok_a^h, e.g., "*Steven Paul Jobs was an American business magnate, entrepreneur and investor.*" or "*Steve Jobs*", the relation is represented as a sentence containing tokens Tok_1^r, ..., Tok_b^r, e.g., "founded", the tail entity is represented as a sentence containing tokens Tok_1^t, ..., Tok_c^t, e.g., "*Apple Inc. is an American multinational technology company headquartered in Cupertino, California.*" or "*Apple Inc.*". The sentences of entities and relations are separated by a special token [SEP]. For a given token, its input representation is constructed by summing the corresponding token, segment and position embeddings. Different elements separated by [SEP] have different segment embeddings, the tokens in sentences of head and tail entity share the same segment embedding e_A, while the tokens in relation sentence have a different segment embedding e_B. Different tokens in the same position $i \in \{1, 2, 3, \ldots, 512\}$ have a same position embedding. Each input token i has a input representation E_i. The token representations are fed into the BERT model architecture which is a multi-layer bidirectional Transformer encoder based on the original implementation described in (Vaswani et al. 2017). The final hidden vector of the special [CLS] token and i-th input token are denoted as $C \in \mathbb{R}^H$ and $T_i \in \mathbb{R}^H$, where H is the hidden state size in pre-trained BERT. The final hidden state C corresponding to [CLS] is used as the aggregate sequence representation for computing triple scores. The only new parameters introduced during triple classification fine-tuning are classification layer weights $W \in \mathbb{R}^{2 \times H}$. The scoring function for a triple $\tau = (h, r, t)$ is $\mathbf{s}_\tau = f(h, r, t) = \text{sigmoid}(CW^T)$, $\mathbf{s}_\tau \in \mathbb{R}^2$ is a 2-dimensional real vector with $s_{\tau 0}, s_{\tau 1} \in [0, 1]$ and $s_{\tau 0} + s_{\tau 1} = 1$. Given the positive triple set $\mathbb{D}^+$ and a negative triple set $\mathbb{D}^-$ constructed accordingly, we compute a

Research question }

instead of fine tunning BERT, can we few-shot LMMs?

cross-entropy loss with $\mathbf{s}_\tau$ and triple labels:

$$\mathcal{L} = -\sum_{\tau \in \mathbb{D}^+ \cup \mathbb{D}^-} (y_\tau \log(s_{\tau 0}) + (1 - y_\tau) \log(s_{\tau 1})) \quad (1)$$

where $y_\tau \in \{0, 1\}$ is the label (negative or positive) of that triple. The negative triple set $\mathbb{D}^-$ is simply generated by replacing head entity h or tail entity t in a positive triple $(h, r, t) \in \mathbb{D}^+$ with a random entity h' or t', i.e.,

$$\begin{aligned} \mathbb{D}^- = &\{(h', r, t) | h' \in \mathbb{E} \wedge h' \neq h \wedge (h', r, t) \notin \mathbb{D}^+\} \\ &\cup \{(h, r, t') | t' \in \mathbb{E} \wedge t' \neq t \wedge (h, r, t') \notin \mathbb{D}^+\} \end{aligned} \quad (2)$$

where $\mathbb{E}$ is the set of entities. Note that a triple will not be treated as a negative example if it is already in positive set $\mathbb{D}^+$. The pre-trained parameter weights and new weights W can be updated via gradient descent.

The architecture of the KG-BERT for predicting relations is shown in Figure 2. We name this KG-BERT version KG-BERT(b). We only use sentences of the two entities h and t to predict the relation r between them. In our preliminary experiment, we found predicting relations with two entities directly is better than using KG-BERT(a) with relation corruption, i.e., generating negative triples by replacing relation r with a random relation r'. As KG-BERT(a), the final hidden state C corresponding to [CLS] is used as the representation of the two entities. The only new parameters introduced in relation prediction fine-tuning are classification layer weights $W' \in \mathbb{R}^{R \times H}$, where R is the number of relations in a KG. The scoring function for a triple $\tau = (h, r, t)$ is $\mathbf{s}'_\tau = f(h, r, t) = \text{softmax}(CW'^T)$, $\mathbf{s}'_\tau \in \mathbb{R}^R$ is a R-dimensional real vector with $s'_{\tau i} \in [0, 1]$ and $\sum_i^R s'_{\tau i} = 1$. We compute the following cross-entropy loss with $\mathbf{s}'_\tau$ and relation labels:

$$\mathcal{L}' = -\sum_{\tau \in \mathbb{D}^+} \sum_{i=1}^{R} y'_{\tau i} \log(s'_{\tau i}) \quad (3)$$

where τ is an observed positive triple, $y'_{\tau i}$ is the relation indicator for the triple τ, $y'_{\tau i} = 1$ when $r = i$ and $y'_{\tau i} = 0$ when $r \neq i$.

Experiments

In this section we evaluate our KG-BERT on three experimental tasks. Specifically we want to determine:

- Can our model judge whether an unseen triple fact (h, r, t) is true or not?
- Can our model predict an entity given another entity and a specific relation?
- Can our model predict relations given two entities?

Datasets. We ran our experiments on six widely used benchmark KG datasets: WN11 (Socher et al. 2013), FB13 (Socher et al. 2013), FB15K (Bordes et al. 2013), WN18RR, FB15k-237 and UMLS (Dettmers et al. 2018). WN11 and WN18RR are two subsets of WordNet, FB15K and FB15k-237 are two subsets of Freebase. WordNet is a large lexical KG of English where each entity as a synset which is consisting of several words and corresponds to a distinct word sense. Freebase is a large knowledge graph of general world facts. UMLS is a medical semantic network containing semantic types (entities) and semantic relations. The test sets of WN11 and FB13 contain positive and negative triplets which can be used for triple classification. The test set of WN18RR, FB15K, FB15k-237 and UMLS only contain correct triples, we perform link (entity) prediction and relation prediction on these datasets. Table 1 provides statistics of all datasets we used.

Dataset	# Ent	# Rel	# Train	# Dev	# Test
WN11	38,696	11	112,581	2,609	10,544
FB13	75,043	13	316,232	5,908	23,733
WN18RR	40,943	11	86,835	3,034	3,134
FB15K	14,951	1,345	483,142	50,000	59,071
FB15k-237	14,541	237	272,115	17,535	20,466
UMLS	135	46	5,216	652	661

Table 1: Summary statistics of datasets.

For WN18RR, we use synsets definitions as entity sentences. For WN11, FB15K and UMLS, we use entity names as input sentences. For FB13, we use entity descriptions in Wikipedia as input sentences. For FB15k-237, we used entity descriptions made by (Xie et al. 2016). For all datasets, we use relation names as relation sentences.

Baselines. We compare our KG-BERT with multiple state-of-the-art KG embedding methods as follows: TransE and its extensions TransH (Wang et al. 2014b), TransD (Ji et al. 2015), TransR (Lin et al. 2015b), TransG (Xiao, Huang, and Zhu 2016), TranSparse (Ji et al. 2016) and PTransE (Lin et al. 2015a), DistMult and its extension DistMult-HRS (Zhang et al. 2018) which only used structural information in KG. The neural tensor network NTN (Socher et al. 2013) and its simplified version ProjE (Shi and Weninger 2017). CNN models: ConvKB (Nguyen et al. 2018a), ConvE (Dettmers et al. 2018) and R-GCN (Schlichtkrull et al. 2018). KG embeddings with textual information: TEKE (Wang and Li 2016), DKRL (Xie et al. 2016), SSP (Xiao et al. 2017), AATE (An et al. 2018). KG embeddings with entity hierarchical types: TKRL (Xie, Liu, and Sun 2016). Contextualized KG embeddings: DOLORES (Wang, Kulkarni, and Wang 2018). Complex-valued KG embeddings ComplEx (Trouillon et al. 2016) and RotatE (Sun et al. 2019). Adversarial learning framework: KBGAN (Cai and Wang 2018).

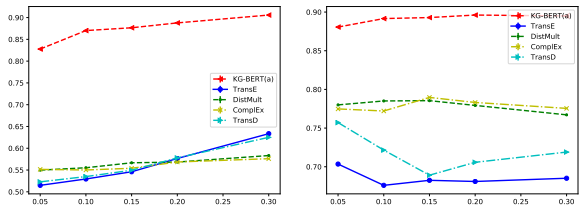
Settings. We choose pre-trained BERT-Base model with 12 layers, 12 self-attention heads and $H = 768$ as the initialization of KG-BERT, then fine tune KG-BERT with Adam implemented in BERT. In our preliminary experiment, we found BERT-Base model can achieve better results than BERT-Large in general, and BERT-Base is simpler and less sensitive to hyper-parameter choices. Following original BERT, we set the following hyper-parameters in KG-BERT

fine-tuning: batch size: 32, learning rate: 5e-5, dropout rate: 0.1. We also tried other values of these hyper-parameters in (Devlin et al. 2019) but didn't find much difference. We tuned number of epochs for different tasks: 3 for triple classification, 5 for link (entity) prediction and 20 for relation prediction. We found more epochs can lead to better results in relation prediction but not in other two tasks. For triple classification training, we sample 1 negative triple for a positive triple which can ensure class balance in binary classification. For link (entity) prediction training, we sample 5 negative triples for a positive triple, we tried 1, 3, 5 and 10 and found 5 is the best.

Method	WN11	FB13	Avg.
NTN (Socher et al. 2013)	86.2	90.0	88.1
TransE (Wang et al. 2014b)	75.9	81.5	78.7
TransH (Wang et al. 2014b)	78.8	83.3	81.1
TransR (Lin et al. 2015b)	85.9	82.5	84.2
TransD (Ji et al. 2015)	86.4	89.1	87.8
TEKE (Wang and Li 2016)	86.1	84.2	85.2
TransG (Xiao, Huang, and Zhu 2016)	87.4	87.3	87.4
TranSparse-S (Ji et al. 2016)	86.4	88.2	87.3
DistMult (Zhang et al. 2018)	87.1	86.2	86.7
DistMult-HRS (Zhang et al. 2018)	88.9	89.0	89.0
AATE (An et al. 2018)	88.0	87.2	87.6
ConvKB (Nguyen et al. 2018a)	87.6	88.8	88.2
DOLORES (Wang, Kulkarni, and Wang 2018)	87.5	89.3	88.4
KG-BERT(a)	**93.5**	**90.4**	**91.9**

Table 2: Triple classification accuracy (in percentage) for different embedding methods. The baseline results are obtained from corresponding papers.

Triple Classification. Triple classification aims to judge whether a given triple (h, r, t) is correct or not. Table 2 presents triple classification accuracy of different methods on WN11 and FB13. We can see that KG-BERT(a) clearly outperforms all baselines by a large margin, which shows the effectiveness of our method. We ran our models 10 times and found the standard deviations are less than 0.2, and the improvements are significant ($p < 0.01$). To our knowledge, KG-BERT(a) achieves the best results so far. For more in-depth performance analysis, we note that TransE could not achieve high accuracy scores because it could not deal with 1-to-N, N-to-1, and N-to-N relations. TransH, TransR, TransD, TranSparse and TransG outperform TransE by introducing relation specific parameters. DistMult performs relatively well, and can also be improved by hierarchical relation structure information used in DistMult-HRS. ConvKB shows decent results, which suggests that CNN models can capture global interactions among the entity and relation embeddings. DOLORES further improves ConvKB by incorporating contextual information in entity-relation random walk chains. NTN also achieves competitive performances especially on FB13, which means it's an expressive model, and representing entities with word embeddings is helpful. Other text-enhanced KG embeddings TEKE and AATE outperform their base models like TransE and TransH, which



(a) WN11 (b) FB13

Figure 3: Test accuracy of triple classification by varying training data proportions.

demonstrates the benefit of external text data. However, their improvements are still limited due to less utilization of rich language patterns. The improvement of KG-BERT(a) over baselines on WN11 is larger than FB13, because WordNet is a linguistic knowledge graph which is closer to linguistic patterns contained in pre-trained language models.

Figure 3 reports triple classification accuracy with 5%, 10%, 15%, 20% and 30% of original WN11 and FB13 training triples. We note that KG-BERT(a) can achieve higher test accuracy with limited training triples. For instance, KG-BERT(a) achieves a test accuracy of 88.1% on FB13 with only 5% training triples and a test accuracy of 87.0% on WN11 with only 10% training triples which are higher than some baseline models (including text-enhanced models) with even the full training triples. These encouraging results suggest that KG-BERT(a) can fully utilize rich linguistic patterns in large external text data to overcome the sparseness of knowledge graphs.

The main reasons why KG-BERT(a) performs well are four fold: 1) The input sequence contains both entity and relation word sequences; 2) The triple classification task is very similar to next sentence prediction task in BERT pre-training which captures relationship between two sentences in large free text, thus the pre-trained BERT weights are well positioned for the inference of relationship among different elements in a triple; 3) The token hidden vectors are contextual embeddings. The same token can have different hidden vectors in different triples, thus contextual information is explicitly used. 4) The self-attention mechanism can discover the most important words connected to the triple fact.

Link Prediction. The link (entity) prediction task predicts the head entity h given $(?, r, t)$ or predicts the tail entity t given $(h, r, ?)$ where ? means the missing element. The results are evaluated using a ranking produced by the scoring function $f(h, r, t)$ ($s_{\tau 0}$ in our method) on test triples. Each correct test triple (h, r, t) is corrupted by replacing either its head or tail entity with every entity $e \in \mathbb{E}$, then these candidates are ranked in descending order of their plausibility score. We report two common metrics, Mean Rank (MR) of correct entities and Hits@10 which means the proportion of correct entities in top 10. A lower MR is better while a higher Hits@10 is better. Following (Nguyen et al. 2018b), we only report results under the *filtered* setting (Bordes et al. 2013) which removes all corrupted triples appeared in

Method	WN18RR		FB15k-237		UMLS	
	MR	Hits@10	MR	Hits@10	MR	Hits@10
TransE (our results)	2365	50.5	223	47.4	1.84	98.9
TransH (our results)	2524	50.3	255	48.6	1.80	**99.5**
TransR (our results)	3166	50.7	237	51.1	1.81	99.4
TransD (our results)	2768	50.7	246	48.4	1.71	99.3
DistMult (our results)	3704	47.7	411	41.9	5.52	84.6
ComplEx (our results)	3921	48.3	508	43.4	2.59	96.7
ConvE (Dettmers et al. 2018)	5277	48	246	49.1	–	–
ConvKB (Nguyen et al. 2018a)	2554	52.5	257	51.7	–	–
R-GCN (Schlichtkrull et al. 2018)	–	–	–	41.7	–	–
KBGAN (Cai and Wang 2018)	–	48.1	–	45.8	–	–
RotatE (Sun et al. 2019)	3340	**57.1**	177	**53.3**	–	–
KG-BERT(a)	**97**	52.4	**153**	42.0	**1.47**	99.0

Table 3: Link prediction results on WN18RR, FB15k-237 and UMLS datasets. The baseline models denoted (our results) are implemented using OpenKE toolkit (Han et al. 2018), other baseline results are taken from the original papers.

Method	Mean Rank	Hits@1
TransE (Lin et al. 2015a)	2.5	84.3
TransR (Xie, Liu, and Sun 2016)	2.1	91.6
DKRL (CNN) (Xie et al. 2016)	2.5	89.0
DKRL (CNN) + TransE (Xie et al. 2016)	2.0	90.8
DKRL (CBOW) (Xie et al. 2016)	2.5	82.7
TKRL (RHE) (Xie, Liu, and Sun 2016)	1.7	92.8
TKRL (RHE) (Xie, Liu, and Sun 2016)	1.8	92.5
PTransE (ADD, len-2 path) (Lin et al. 2015a)	**1.2**	93.6
PTransE (RNN, len-2 path) (Lin et al. 2015a)	1.4	93.2
PTransE (ADD, len-3 path) (Lin et al. 2015a)	1.4	94.0
SSP (Xiao et al. 2017)	**1.2**	–
ProjE (pointwise) (Shi and Weninger 2017)	1.3	95.6
ProjE (listwise) (Shi and Weninger 2017)	**1.2**	95.7
ProjE (wlistwise) (Shi and Weninger 2017)	**1.2**	95.6
KG-BERT (b)	**1.2**	**96.0**

Table 4: Relation prediction results on FB15K dataset. The baseline results are obtained from corresponding papers.

training, development, and test set before getting the ranking lists.

Table 3 shows link prediction performance of various models. We test some classical baseline models with OpenKE toolkit (Han et al. 2018)[1], other results are taken from the original papers. We can observe that: 1) KG-BERT(a) can achieve lower MR than baseline models, and it achieves the lowest mean ranks on WN18RR and FB15k-237 to our knowledge. 2) The Hits@10 scores of KG-BERT(a) is lower than some state-of-the-art methods. KG-BERT(a) can avoid very high ranks with semantic relatedness of entity and relation sentences, but the KG structure information is not explicitly modeled, thus it could not rank some neighbor entities of a given entity in top 10. CNN models ConvE and ConvKB perform better compared to the graph convolutional network R-GCN. ComplEx could not perform well on WN18RR and FB15k-237, but can be improved using adversarial negative sampling in KBGAN and RotatE.

Relation Prediction. This task predicts relations between two given entities, i.e., $(h, ?, t)$. The procedure is similar to link prediction while we rank the candidates with the relation scores $\mathbf{s}'_\tau$. We evaluate the relation ranking using Mean Rank (MR) and Hits@1 with *filtered* setting.

Table 4 reports relation prediction results on FB15K. We note that KG-BERT(b) also shows promising results and achieves the highest Hits@1 so far. The KG-BERT(b) is analogous to sentence pair classification in BERT fine-tuning and can also benefit from BERT pre-training. Text-enhanced models DKRL and SSP can also outperform structure only methods TransE and TransH. TKRL and PTransE work well with hierarchical entity categories and extended path information. ProjE achieves very competitive results by treating KG completion as a ranking problem and optimizing ranking score vectors.

Attention Visualization. We show attention patterns of KG-BERT in Figure 4 and Figure 5. We use the visualization tool released by (Vig 2019)[2]. Figure 4 depicts the attention patterns of KG-BERT(a). A positive training triple (*__twenty_dollar_bill_NN_1*, _hypernym, *__note_NN_6*) from WN18RR is taken as the example. The entity descriptions "a United States bill worth 20 dollars" and "a piece of paper money" as well as the relation name "hypernym" are used as the input sequence. We observe that some important words such as "paper" and "money" have higher attention scores connected to the label token [CLS], while some less related words like "united" and "states" obtain less attentions. On the other hand, we can see that different attention

[1]https://github.com/thunlp/OpenKE

[2]https://github.com/jessevig/bertviz

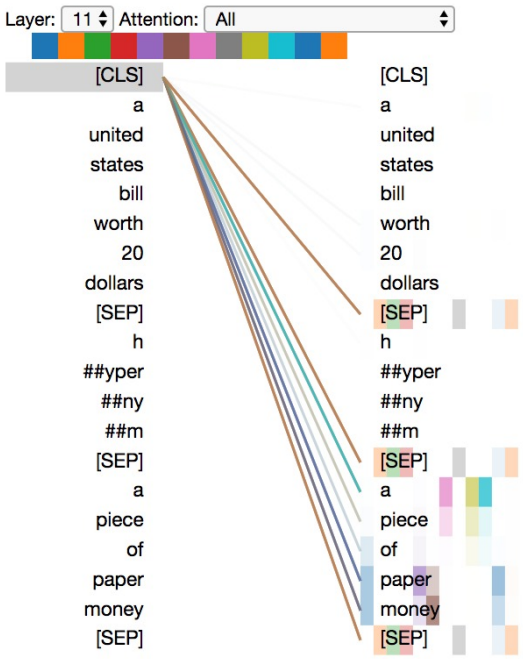


Figure 4: Illustrations of attention patterns of KG-BERT(a). A positive training triple (*__twenty_dollar_bill_NN_1*, _hypernym, *__note_NN_6*) from WN18RR is used as the example. Different colors mean different attention heads. Transparencies of colors reflect the attention scores. We show the attention weights between [CLS] and other tokens in layer 11 of the Transformer model.

heads focus on different tokens. [SEP] is highlighted by the same six attention heads, "a" and "piece" are highlighted by the three same attention heads, while "paper" and "money" are highlighted by other four attention heads. As mentioned in (Vaswani et al. 2017), multi-head attention allows KG-BERT to jointly attend to information from different representation subspaces at different positions, different attention heads are concatenated to compute the final attention values. Figure 5 illustrates attention patterns of KG-BERT(b). The triple (*20th century*, /time/event/includes_event, *World War II*) from FB15K is taken as input. We can see similar attention patterns as in KG-BERT(a), six attention heads attend to "century" in head entity, while other three attention heads focus on "war" and "ii" in tail entity. Multi-head attention can attend to different aspects of two entities in a triple.

Discussions. From experimental results, we note that KG-BERT can achieve strong performance in three KG completion tasks. However, a major limitation is that BERT model is expensive, which makes the link prediction evaluation very time consuming, link prediction evaluation needs to replace head or tail entity with almost all entities, and all corrupted triple sequences are fed into the 12 layer Transformer model. Possible solutions are introducing 1-N scoring models like ConvE or using lightweight language models.

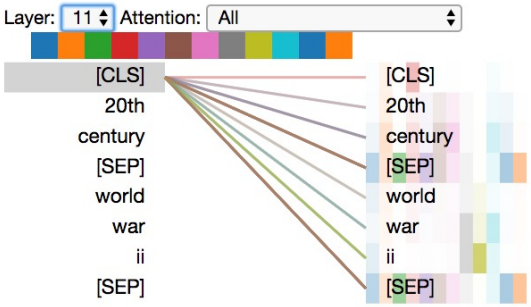


Figure 5: Illustrations of attention patterns of KG-BERT(b). The example is taken from FB15K. Two entities *20th century* and *World War II* are used as input, the relation label is /time/event/includes_event.

Conclusion and Future Work

In this work, we propose a novel knowledge graph completion method termed Knowledge Graph BERT (KG-BERT). We represent entities and relations as their name/description textual sequences, and turn knowledge graph completion problem into a sequence classification problem. KG-BERT can make use of rich language information in large amount free text and highlight most important words connected to a triple. The proposed method demonstrates promising results by outperforming state-of-the-art results on multiple benchmark KG datasets.

Some future directions include improving the results by jointly modeling textual information with KG structures, or utilizing pre-trained models with more text data like XLNet. And applying our KG-BERT as a knowledge-enhanced language model to language understanding tasks is an interesting future work we are going to explore.

References

An, B.; Chen, B.; Han, X.; and Sun, L. 2018. Accurate text-enhanced knowledge graph representation learning. In *NAACL*, 745–755.

Bollacker, K.; Evans, C.; Paritosh, P.; Sturge, T.; and Taylor, J. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *SIGMOD*, 1247–1250.

Bordes, A.; Usunier, N.; Garcia-Duran, A.; Weston, J.; and Yakhnenko, O. 2013. Translating embeddings for modeling multi-relational data. In *NIPS*, 2787–2795.

Bosselut, A.; Rashkin, H.; Sap, M.; Malaviya, C.; Celikyilmaz, A.; and Choi, Y. 2019. COMET: Commonsense transformers for automatic knowledge graph construction. In *ACL*, 4762–4779.

Cai, L., and Wang, W. Y. 2018. KBGAN: Adversarial learning for knowledge graph embeddings. In *NAACL*, 1470–1480.

Cui, W.; Xiao, Y.; Wang, H.; Song, Y.; Hwang, S.-w.; and Wang, W. 2017. KBQA: learning question answering over qa corpora and knowledge bases. *Proceedings of the VLDB Endowment* 10(5):565–576.

Dettmers, T.; Minervini, P.; Stenetorp, P.; and Riedel, S. 2018. Convolutional 2d knowledge graph embeddings. In *AAAI*, 1811–1818.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*, 4171–4186.

Han, X.; Cao, S.; Lv, X.; Lin, Y.; Liu, Z.; Sun, M.; and Li, J. 2018. OpenKE: An open toolkit for knowledge embedding. In *EMNLP*, 139–144.

Ji, G.; He, S.; Xu, L.; Liu, K.; and Zhao, J. 2015. Knowledge graph embedding via dynamic mapping matrix. In *ACL*, 687–696.

Ji, G.; Liu, K.; He, S.; and Zhao, J. 2016. Knowledge graph completion with adaptive sparse transfer matrix. In *AAAI*.

Lin, Y.; Liu, Z.; Luan, H.; Sun, M.; Rao, S.; and Liu, S. 2015a. Modeling relation paths for representation learning of knowledge bases. In *EMNLP*, 705–714.

Lin, Y.; Liu, Z.; Sun, M.; Liu, Y.; and Zhu, X. 2015b. Learning entity and relation embeddings for knowledge graph completion. In *AAAI*.

Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G. S.; and Dean, J. 2013. Distributed representations of words and phrases and their compositionality. In *NIPS*, 3111–3119.

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

Nguyen, D. Q.; Nguyen, D. Q.; Nguyen, T. D.; and Phung, D. 2018a. A convolutional neural network-based model for knowledge base completion and its application to search personalization. *Semantic Web*.

Nguyen, D. Q.; Nguyen, T. D.; Nguyen, D. Q.; and Phung, D. 2018b. A novel embedding model for knowledge base completion based on convolutional neural network. In *NAACL*, 327–333.

Nickel, M.; Tresp, V.; and Kriegel, H.-P. 2011. A three-way model for collective learning on multi-relational data. In *ICML*, 809–816.

Pennington, J.; Socher, R.; and Manning, C. 2014. Glove: Global vectors for word representation. In *EMNLP*.

Peters, M. E.; Neumann, M.; Iyyer, M.; Gardner, M.; Clark, C.; Lee, K.; and Zettlemoyer, L. 2018. Deep contextualized word representations. In *NAACL*, 2227–2237.

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

Schlichtkrull, M.; Kipf, T. N.; Bloem, P.; Van Den Berg, R.; Titov, I.; and Welling, M. 2018. Modeling relational data with graph convolutional networks. In *ESWC*, 593–607.

Shi, B., and Weninger, T. 2017. ProjE: Embedding projection for knowledge graph completion. In *AAAI*.

Socher, R.; Chen, D.; Manning, C. D.; and Ng, A. 2013. Reasoning with neural tensor networks for knowledge base completion. In *NIPS*, 926–934.

Suchanek, F. M.; Kasneci, G.; and Weikum, G. 2007. Yago: a core of semantic knowledge. In *WWW*, 697–706. ACM.

Sun, Z.; Deng, Z.-H.; Nie, J.-Y.; and Tang, J. 2019. Rotate: Knowledge graph embedding by relational rotation in complex space. In *ICLR*.

Trouillon, T.; Welbl, J.; Riedel, S.; Gaussier, É.; and Bouchard, G. 2016. Complex embeddings for simple link prediction. In *ICML*, 2071–2080.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. In *NIPS*, 5998–6008.

Vig, J. 2019. A multiscale visualization of attention in the transformer model. *arXiv preprint arXiv:1906.05714*.

Wang, Z., and Li, J.-Z. 2016. Text-enhanced representation learning for knowledge graph. In *IJCAI*, 1293–1299.

Wang, Z.; Zhang, J.; Feng, J.; and Chen, Z. 2014a. Knowledge graph and text jointly embedding. In *EMNLP*.

Wang, Z.; Zhang, J.; Feng, J.; and Chen, Z. 2014b. Knowledge graph embedding by translating on hyperplanes. In *AAAI*.

Wang, Q.; Mao, Z.; Wang, B.; and Guo, L. 2017. Knowledge graph embedding: A survey of approaches and applications. *IEEE TKDE* 29(12):2724–2743.

Wang, H.; Kulkarni, V.; and Wang, W. Y. 2018. Dolores: Deep contextualized knowledge graph embeddings. *arXiv preprint arXiv:1811.00147*.

Xiao, H.; Huang, M.; Meng, L.; and Zhu, X. 2017. SSP: semantic space projection for knowledge graph embedding with text descriptions. In *AAAI*.

Xiao, H.; Huang, M.; and Zhu, X. 2016. TransG: A generative model for knowledge graph embedding. In *ACL*, volume 1, 2316–2325.

Xie, R.; Liu, Z.; Jia, J.; Luan, H.; and Sun, M. 2016. Representation learning of knowledge graphs with entity descriptions. In *AAAI*.

Xie, R.; Liu, Z.; and Sun, M. 2016. Representation learning of knowledge graphs with hierarchical types. In *IJCAI*, 2965–2971.

Xu, J.; Qiu, X.; Chen, K.; and Huang, X. 2017. Knowledge graph representation with jointly structural and textual encoding. In *IJCAI*, 1318–1324.

Yang, B.; Yih, W.-t.; He, X.; Gao, J.; and Deng, L. 2015. Embedding entities and relations for learning and inference in knowledge bases. In *ICLR*.

Yang, Z.; Dai, Z.; Yang, Y.; Carbonell, J.; Salakhutdinov, R.; and Le, Q. V. 2019. XLNet: Generalized autoregressive pretraining for language understanding. *arXiv preprint arXiv:1906.08237*.

Zhang, F.; Yuan, N. J.; Lian, D.; Xie, X.; and Ma, W.-Y. 2016. Collaborative knowledge base embedding for recommender systems. In *KDD*, 353–362. ACM.

Zhang, Z.; Zhuang, F.; Qu, M.; Lin, F.; and He, Q. 2018. Knowledge graph embedding with hierarchical relation structure. In *EMNLP*, 3198–3207.

Zhang, Z.; Han, X.; Liu, Z.; Jiang, X.; Sun, M.; and Liu, Q. 2019. ERNIE: Enhanced language representation with informative entities. In *ACL*, 1441–1451.