Joint Language Semantic and Structure Embedding for Knowledge Graph Completion

Jianhao Shen[1], Chenguang Wang[2]*, **Linyuan Gong[3], Dawn Song[3]**
[1]Peking University, [2]Washington University in St. Louis, [3]UC Berkeley
`jhshen@pku.edu.cn, chenguangwang@wustl.edu,`
`{gly,dawnsong}@berkeley.edu`

Abstract

The task of completing knowledge triplets has broad downstream applications. Both structural and semantic information plays an important role in knowledge graph completion. Unlike previous approaches that rely on either the structures or semantics of the knowledge graphs, we propose to jointly embed the semantics in the natural language description of the knowledge triplets with their structure information. Our method embeds knowledge graphs for the completion task via fine-tuning pre-trained language models with respect to a probabilistic structured loss, where the forward pass of the language models captures semantics and the loss reconstructs structures. Our extensive experiments on a variety of knowledge graph benchmarks have demonstrated the state-of-the-art performance of our method. We also show that our method can significantly improve the performance in a low-resource regime, thanks to the better use of semantics. The code and datasets are available at https://github.com/pkusjh/LASS.

1 Introduction

Knowledge graphs (KG), such as Wikidata and Freebase (Bollacker et al., 2008), consist of factual triplets. KGs have been useful resources for both humans and machines. A triplet in the form of *(head entity, relation, tail entity)*, where the relation involves both head and tail entities, has been used in a great variety of applications, such as question answering (Guu et al.; Hao et al., 2017) and web search (Xiong et al., 2017). Incompleteness has been a longstanding issue in KGs (Carlson et al., 2010), impeding their wider adoption in real-world applications.

KG completion aims to predict a missing entity or relation of a factual triplet. Structural patterns in the existing triplets are useful to predict the missing elements (Bordes et al., 2013; Sun et al., 2019). For

---
*Corresponding author

example, a composition pattern can be learned to predict the relation *grandmother_Of* based on two consecutive *mother_Of* relations. Besides the structure information, semantic relatedness between entities and relations is also critical to infer entities or relations with similar meanings (An et al., 2018; Yao et al., 2019; Wang et al., 2021). For example, if a relationship *CEO_Of* holds between two entities, the relation *employee_Of* also holds. There are two kinds of KG completion approaches that fall into different learning paradigms. First, the structure-based approaches treat entities and relations as nodes and edges, and use graph embedding methods to learn their representations. Second, the semantic-based approaches encode the text description of entities and relations via language models. While both structures and semantics are vital to KG completion, it is non-trivial for existing methods to process both structural and semantic information.

In this paper, we propose LASS, a joint language semantic and structure embedding for knowledge graph completion, which incorporates both semantics and structures in a KG triplet. LASS embeds a triplet into a vector space by fine-tuning pre-trained language models (LM) with respect to a structured loss. LASS involves both semantic embedding and structure embedding. The semantic embedding captures the semantics of the triplet, which corresponds to the forward pass of a pre-trained LM over the natural language description of the triplet. The structure embedding aims to reconstruct the structures in the semantic embedding, which corresponds to optimizing a probabilistic structured loss via the backpropagation of the LM. Intuitively, the structured loss treats the relationship between two entities as a translation between embeddings of the entities. LASS outperforms the existing approaches on a collection of KG completion benchmarks. We further evaluate LASS in low-resource settings and find that it is more data-efficient than other methods. The reason is that our method ex-

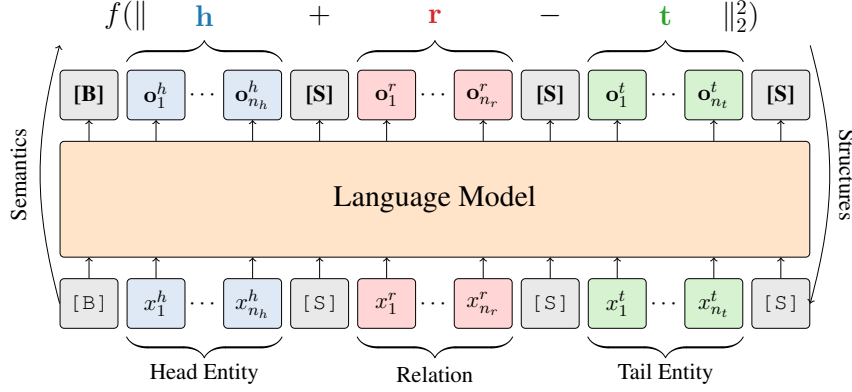$$f(\| \quad \mathbf{h} \quad + \quad \mathbf{r} \quad - \quad \mathbf{t} \quad \|_2^2)$$

Figure 1: Overview of LaSS. LaSS maps a knowledge triplet *(Head Entity, Relation, Tail Entity)*, in short *(h, r, t)*, to the corresponding embedding vectors, $\mathbf{h}, \mathbf{r}, \mathbf{t} \in \mathbb{R}^k$. LaSS embeds KGs for KG completion via fine-tuning pre-trained language models (LM) w.r.t. a probabilistic structured loss, where the forward pass of the LMs captures semantics and the loss reconstructs structures. In particular, LaSS consists of semantic embedding and structure embedding. The *semantic embedding* (leftmost arrow) is generated by a forward pass of the LMs followed by a pooling layer over the natural language description of a triplet. [B] (the beginning token) and [S] (the separator token) are special tokens of LMs attached to the description. For example, the textual description of head entity is $(x_1^h, \cdots, x_{n_h}^h)$. $\mathbf{h}$ is calculated as the mean pooling of the corresponding LM outputs $(\mathbf{o}_1^h, \cdots, \mathbf{o}_{n_h}^h)$. $\mathbf{r}$ and $\mathbf{t}$ are calculated similarly. The *structure embedding* (rightmost arrow) reconstructs KG structures in the semantic embeddings via optimizing a structured loss on top of the LMs through backpropagation. The structured loss is based on a score function $f(\|\mathbf{h} + \mathbf{r} - \mathbf{t}\|_2^2)$, which regards the relationship between two entities corresponds to a translation between the embeddings of the entities. The goal is to minimize the loss function so that $\mathbf{h} + \mathbf{r} \approx \mathbf{t}$ when *(h, r, t)* holds.

ploits both semantics and structures in the training data.

The contributions are the following:

- We design a natural language embedding approach, LaSS, that integrates both structural and semantic information of KGs, for KG completion. We train LaSS by fine-tuning pre-trained LMs w.r.t. a structured loss, where the forward pass of the LMs captures semantics and the loss reconstructs structures. The method consists of both the KG module and the LM module, which sheds light on the connections between the KGs and deep language representation, and advances the research at the intersection of the two areas.

- We evaluate LaSS on two KG completion tasks, link prediction and triplet classification, and obtain state-of-the-art performance. The results suggest that capturing both semantics and structures is critical to understand the KGs. The findings are beneficial to many downstream knowledge-driven applications.

- We show that we can significantly improve the performance in the low-resource settings over existing approaches, thanks to the improved transfer of knowledge about semantics.

2 LaSS

We introduce LaSS to embed both semantics and structures of knowledge graphs (KG) with natural language. As shown in Figure 1, LaSS incorporates two embeddings: semantic embedding and structure embedding. The semantic embedding captures the semantics in the natural language description of the KG triplets. The structure embedding further reconstructs the structure information of the KGs from the semantic embedding. LaSS embeds KG in a vector space by fine-tuning a pre-trained language model (LM) w.r.t. a structured loss, where the forward pass performs semantic embedding and the optimization of structured loss conducts structure embedding.

2.1 Semantic Embedding

A KG of triplets is denoted as G. Each triplet of G is in the form of *(h, r, t)*, where $h, t \in E$ and $r \in R$. E is the set of entities, and R is the set of relations. The semantic similarities between the head entity h, relation r, and tail entity t are crucial to complete a factual triplet. For example, given h = "Bob Dylan" and r = "was born in", the task is to predict a missing t, where the candidates are "Duluth" and "Apple". The semantic similarity between "Bob Dylan" and "Duluth", as well as the similarity between "was born in" and "Duluth" should be larger than

their similarities with "Apple" as "Duluth" is the ground-truth answer. Pre-trained LMs capture the rich semantics in natural language via pre-training on large-scale textual corpora. This inspires us to use the semantics stored in the parameters of LMs to encode the semantics of triplets.

Formally, for a triplet *(h, r, t)*, both entities (*h* and *t*) and relation (*r*) are represented by their corresponding natural language descriptions. The head entity *h* is represented as a sequence of tokens, $T^h = (x_1^h, \cdots, x_{n_h}^h)$, describing the entity. Similarly, $T^t = (x_1^t, \cdots, x_{n_t}^t)$ represents the tail entity *t*. $T^r = (x_1^r, \cdots, x_{n_r}^r)$ denotes the relation *r*. We generate the semantic embedding via the forward pass of the LMs as shown in Figure 1. The knowledge graph completion tasks require explicit modeling of dependency of the head, relation and tail. For example, both the connections between head and tail, and relation and tail contribute to the prediction of the tail in the link prediction task. Therefore, we use the concatenation of T^h, T^r, and T^t as the input sequence to the LMs, and use the mean pooling over the output representation of every token in T^h, T^r, and T^t from the forward pass of LMs as $\mathbf{h}, \mathbf{r}, \mathbf{t} \in \mathbb{R}^k$, where k is the dimension of the embedding vectors.

More specifically, we construct the input sequence in the following format: [B] T^h [S] T^r [S] T^t [S], where [B] is a special symbol added in front of every input sequence, and [S] is a special separator token. The special tokens are different for various LMs. For example, [B] and [S] are implemented as [CLS] and [SEP] for BERT (Devlin et al., 2019) respectively. The input sequence is then converted to the corresponding input embeddings of the LMs. For example, the input embeddings of BERT are the sum of the token embeddings, the segment embeddings, and the position embeddings. The input embeddings are fed into the LM. We add a mean pooling layer on top of the output layer of the LM and perform mean pooling over the output representation of every token in T^h, i.e., $(\mathbf{o}_1^h, \cdots, \mathbf{o}_{n_h}^h)$, resulting in $\mathbf{h}$ as illustrated in Figure 1. We obtain $\mathbf{r}$ and $\mathbf{t}$ in the same way. The dimension k equals to the hidden size of the LM.

2.2 Structure Embedding

Structural information of KGs has been successfully used in the KG completion. Traditional approaches regard the relationship between two entities corresponds to a translation between the embeddings of the entities. This is different from the above semantic embedding and the forward pass cannot capture the structure information. We propose to incorporate the structure embedding by fine-tuning the pre-trained LM with a structure loss.

The goal is to reconstruct structure information in the semantic embedding. The updated embeddings of *h*, *r*, and *t* are still denoted as $\mathbf{h}$, $\mathbf{r}$, and $\mathbf{t}$, which incorporate structure information of KGs while preserve semantic information. We reconstruct structure information in the semantic embeddings via optimizing a probabilistic structured loss, in which the score function of a triplet *(h, r, t)* is defined by Eq. 1:

$$f(\mathbf{h}, \mathbf{r}, \mathbf{t}) = b - \frac{1}{2}\|\mathbf{h} + \mathbf{r} - \mathbf{t}\|_2^2 \qquad (1)$$

If *(h, r, t)* holds, we have $\mathbf{h} + \mathbf{r} \approx \mathbf{t}$. We also use $f(\|\mathbf{h} + \mathbf{r} - \mathbf{t}\|_2^2)$ to denote this in Figure 1 for simplicity. The score function is motivated by TransE (Bordes et al., 2013).

We define the following probabilistic model based on the score function (1):

$$\Pr(h|r, t) = \frac{\exp(f(\mathbf{h}, \mathbf{r}, \mathbf{t}))}{\sum_{\tilde{h} \in E} \exp(f(\tilde{\mathbf{h}}, \mathbf{r}, \mathbf{t}))} \qquad (2)$$

Here $\tilde{h}$ is the corrupted head sampled from the entity set E. $\Pr(r|h, t)$ and $\Pr(t|h, r)$ have a similar form except that the summation in the denominator is over corrupted relations and tails, respectively.

The probabilistic structured loss is defined in Eq. 3. The goal is to minimize the negative log likelihood over the KG:

$$L = - \sum_{(h,r,t) \in G} (\log \Pr(h|r, t) + \log \Pr(r|h, t) + \log \Pr(t|h, r)) \qquad (3)$$

Optimization Computing the probability in Eq. 2 is computationally inefficient since it requires a forward pass of all possible triplets $(\tilde{h}, r, t)$ to compute the denominator. We use negative sampling (Mikolov et al., 2013) to make training more efficient. Instead of minimizing $-\log \Pr(h|r, t)$ as in Eq. 3, we optimize the loss as is described in Eq. 4 for modeling *h*.

$$L_h = - \log \Pr(1|h, r, t)$$
$$- \sum_{i}^{n_{\text{ns}}} \mathbb{E}_{\tilde{h}_i \sim E \setminus \{h\}} \log \Pr(0|\tilde{h}_i, r, t) \qquad (4)$$

where $\Pr(1|h, r, t) = \sigma(f(\mathbf{h}, \mathbf{r}, \mathbf{t}))$.

The loss for modeling r and t is similarly defined. Here, hyperparameter n_{ns} is the number of negative samples. Each negatively sampled head $\tilde{h}_i$ is drawn uniformly without replacement from the entity set $E \backslash \{h\}$. A sample is not treated as a negative sample if it is already a positive example. We have the final structured loss $L = \sum_{(h,r,t) \in G} (L_h + L_r + L_t)$ by adopting the similar negative sampling procedures for relations and tail entities.

The training of LaSS is unified as fine-tuning an LM with respect to a structured loss. The semantic embedding is obtained by the forward pass of the LM. The structure embedding is conducted by optimizing the structured loss through backpropagation of the LM.

3 Experiments

3.1 Experimental Setup

Datasets We test the performance of our method on five KG benchmarks built with three KGs: Freebase (Bollacker et al., 2008), WordNet (Miller, 1994) and UMLS (Dettmers et al., 2018). Freebase is a large-scale KG containing general knowledge facts. We employ two subsets from Freebase, namely FB15K-237 (Toutanova and Chen, 2015), and FB13 (Socher et al., 2013). WordNet provides semantic knowledge of words. We use two subsets from WordNet, namely WN18RR (Dettmers et al., 2018), and WN11 (Socher et al., 2013). UMLS is a medical semantic network containing semantic entities and relations. The statistics are summarized in Table 1. We also provide a detailed description of the datasets in Appendix A.1.

Dataset	# Entity	# Relation	# Train	# Dev	# Test
FB15k-237	14,541	237	272,115	17,535	20,466
WN18RR	40,943	11	86,835	3,034	3,134
UMLS	135	46	5,216	652	661
FB13	75,043	13	316,232	5,908	23,733
WN11	38,696	11	112,581	2,609	10,544

Table 1: Statistics of knowledge graphs.

Implementation Details We use two families of LMs with LaSS. First, we adopt both BERT$_{BASE}$ and BERT$_{LARGE}$ from (Devlin et al., 2019) with LaSS, namely LaSS-BERT$_{BASE}$ and LaSS-BERT$_{LARGE}$. Second, RoBERTa family (Liu et al., 2019) is used, namely LaSS-RoBERTa$_{BASE}$ and LaSS-RoBERTa$_{LARGE}$.

We train LaSS with AdamW (Loshchilov and Hutter, 2019) on each KG dataset via fine-tuning

the corresponding LMs. The training hyperparameters are set as follows. For LaSS-BERT$_{BASE}$ and LaSS-RoBERTa$_{BASE}$, the batch size is set to 128, the learning rate is set to 3e-5 with linear warm-up and 0.01 weight decay. We set the batch size to 64 for LaSS-BERT$_{LARGE}$ and LaSS-RoBERTa$_{LARGE}$. The number of training epochs is set to 5. The margin b in Eq. 1 is empirically set to 7. We sample 5 negative entities or relations resulting in 15 negative triplets for each positive triplet for the negative sampling.

We represent entities and relations as their names or descriptions (Yao et al., 2019). For FB15k-237, we used entity descriptions from (Xie et al., 2016). For FB13, we use entity descriptions in Wikipedia. For WN18RR, we use definitions of synsets as entity descriptions. For WN11 and UMLS, the entity names are used as the entity descriptions. The relation descriptions are based on the relation names across all the datasets. The input sequence is constructed based on Sec. 2.1. For LaSS-BERT$_{BASE}$ and LaSS-BERT$_{LARGE}$, we use a character-level BPE vocabulary. `[B]` is replaced with `[CLS]`, and `[S]` is replaced with `[SEP]`. For LaSS-RoBERTa$_{BASE}$ and LaSS-RoBERTa$_{LARGE}$, we use a byte-level BPE vocabulary, and `[B]` and `[S]` are replaced with `BOS` and `EOS` respectively. We implement LaSS using the Transformers package (Wolf et al., 2020).

Comparison Methods We compare our method to state-of-the-art methods, including (i) shallow structure embedding: TransE (Bordes et al., 2013), TransH (Wang et al., 2014b), TransR (Lin et al., 2015), TransD (Ji et al., 2015), TransG (Xiao et al., 2016), TranSparse (Ji et al., 2016), DistMult (Yang et al., 2015), DistMult-HRS (Zhang et al., 2018), ConvE (Dettmers et al., 2018), ConvKB (Nguyen et al., 2018), ComplEx (Trouillon et al., 2016), RotatE (Sun et al., 2019), REFE (Chami et al., 2020), HAKE (Zhang et al., 2019a), and ComplEx-DURA (Zhang et al., 2020); (ii) deep structure embedding: NTN (Socher et al., 2013), DO-LORES (Wang et al., 2018), KBGAT (Nathani et al., 2019), GAATs (Wang et al., 2020), NePTuNe (Sonkar et al., 2021), and ComplEx-N3-RP (Chen et al., 2021); (iii) language semantic embedding: TEKE (Wang and Li, 2016), KG-BERT (Yao et al., 2019), and stAR (Wang et al., 2021). We present a detailed technical description of the above methods in Appendix A.2.

Method	WN11	FB13	Avg
NTN (Socher et al., 2013)	86.2	90.0	88.1
TransE (Bordes et al., 2013)	75.9	81.5	78.7
TransH (Wang et al., 2014b)	78.8	83.3	81.1
TransR (Lin et al., 2015)	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 et al., 2016)	87.4	87.3	87.4
TranSparse-S (Ji et al., 2016)	86.4	88.2	87.3
DistMult (Yang et al., 2015)	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., 2018)	87.6	88.8	88.2
DOLORES (Wang et al., 2018)	87.5	89.3	88.4
KG-BERT (Yao et al., 2019)	93.5	90.4	91.9
LaSS-BERT$_{BASE}$ (ours)	93.3	91.2	92.3
LaSS-BERT$_{LARGE}$ (ours)	**94.5**	**91.8**	**93.2**
LaSS-RoBERTa$_{BASE}$ (ours)	92.3	91.1	91.7
LaSS-RoBERTa$_{LARGE}$ (ours)	93.8	91.6	92.7

Table 2: Triplet classification accuracy on WN11 and FB13.

3.2 Triplet Classification

The task of triplet classification judges whether a given triplet *(h, r, t)* is correct or not. The task is a binary classification task. We use WN11 and FB13 for the task, since only the test sets of the two datasets contain both positive and negative triplets among all the datasets. For the task, we use the score function as defined in Eq. 1 , and set a score threshold. For a triplet, if the score is above the threshold, the triplet is classified as positive, otherwise negative. We set the threshold empirically based on the accuracy on the validation set. As shown in Table 2, we conclude with the following findings.

Head	Relation	Tail	Label
ron ziegler	gender	male	✓
john fortescue	profession	writer	✓
george j adams	cause of death	typhoid fever	✓
fleiss joseph	institution	columbia university	✓
edmund husserl	nationality	austria	✓
aleksandr bakulev	gender	female	✗
emile littre	profession	physicist	✗
joseph smith jr	cause of death	emphysema	✗
frank g slaughter	institution	university of toronto	✗
julius klinger	nationality	romania	✗

Table 3: Samples of LaSS's correct predictions on FB13, where KG-BERT (Yao et al., 2019) outputs wrong predictions. Label ✓ means a gold positive triplet. ✗ indicates a gold negative triplet.

We find that our methods consistently produce state-of-the-art results on triplet classification tasks. This indicates that our score function has captured semantics and structures that are crucial for the triplet classification. We also notice that LaSS-BERT generates slightly better results compared
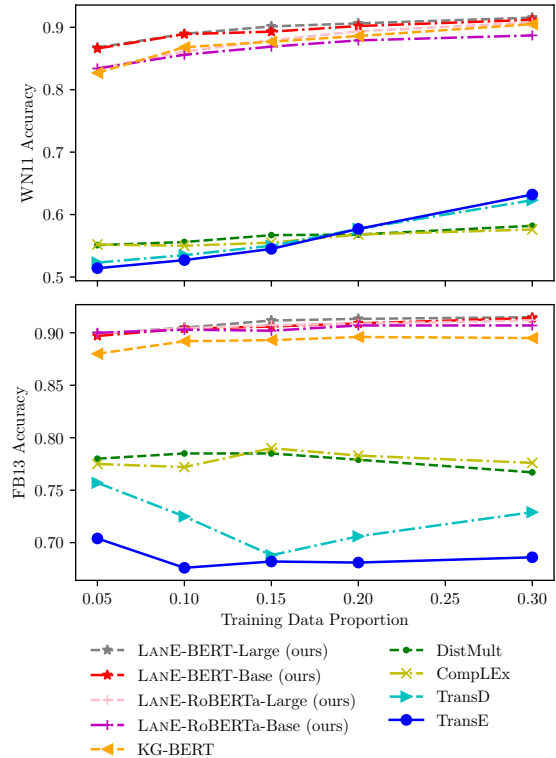

Figure 2: Triplet classification accuracy in a low-resource regime: training with different proportions of the corresponding training datasets on WN11 and FB13.

to LaSS-RoBERTa. This is due to RoBERTa removing the NSP objective, however the objective naturally fits in the triplet classification task. LaSS-RoBERTa still generates reasonable results. The reason is that the masked LM objective captures the necessary semantics needed for the triplet classification, and LaSS is able to preserve the important semantic information.

In Table 3, we also show some cases where LaSS-BERT$_{BASE}$ makes correct predictions while KG-BERT produces incorrect ones on FB13. Compared to KG-BERT, we find that LaSS is more capable in relations that require comprehensive structure information, such as "institution".

3.3 Low-Resource Settings

We additionally test the accuracy of triplet classification in a low-data regime, in particular, when using 5%, 10%, 15%, 20%, and 30% of the training data on WN11 and FB13. The results are shown in Figure 2. LaSS-BERT$_{LARGE}$ consistently outperforms the state-of-the-art KG-BERT. This indicates that LaSS is more data-efficient, as it leverages both semantics and structures in the training data. We also find that LaSS is able to produce competitive results with less training data

Method	FB15k-237		WN18RR		UMLS	
	Hits@10	MR	Hits@10	MR	Hits@10	MR
TransE (Bordes et al., 2013)	0.465	357	0.501	3384	0.989	1.84
DistMult (Yang et al., 2015)	0.419	254	0.49	5110	0.846	5.52
ComplEx (Trouillon et al., 2016)	0.428	339	0.51	5261	0.967	2.59
ConvE (Dettmers et al., 2018)	0.501	244	0.52	4187	0.990	1.51
RotatE (Sun et al., 2019)	0.533	177	0.571	3340	-	-
HAKE (Zhang et al., 2019a)	0.542	-	0.582	-	-	-
KBGAT (Nathani et al., 2019)	0.626	210	0.581	1940	-	-
KG-BERT (Yao et al., 2019)	0.420	153	0.524	97	0.990	1.47
REFE (Chami et al., 2020)	0.541	-	0.561	-	-	-
GAATs (Wang et al., 2020)	**0.650**	187	0.604	1270	-	-
ComplEx-DURA (Zhang et al., 2020)	0.560	-	0.571	-	-	-
StAR (Wang et al., 2021)	0.562	117	0.732	46	0.991	1.49
NePTuNe (Sonkar et al., 2021)	0.547	-	0.557	-	-	-
ComplEx-N3-RP (Chen et al., 2021)	0.568	-	0.580	-	**0.998**	-
LASS-BERT$_{BASE}$ (ours)	0.479	131	0.725	55	0.991	**1.39**
LASS-BERT$_{LARGE}$ (ours)	0.527	120	0.769	41	0.990	1.58
LASS-RoBERTa$_{BASE}$ (ours)	0.500	116	0.737	53	0.994	1.41
LASS-RoBERTa$_{LARGE}$ (ours)	0.533	**108**	**0.786**	**35**	0.989	1.56

Table 4: Link prediction results on FB15k-237, WN18RR and UMLS.

compared to existing methods even with full training data. LASS-BERT$_{LARGE}$ with 5% training data of WN11 outperforms most of the existing methods using full training data. When using 10% training data of FB13, LASS-BERT$_{LARGE}$ is able to perform comparably with KG-BERT with full training data, and outperforms the remaining methods. This is because LASS transfers the knowledge about semantics better to the tasks compared to existing approaches without fully leveraging the KG semantics. The results suggest that LASS is effective in low-resource scenarios.

3.4 Link Prediction

Link prediction aims to predict a missing entity given a relation and the other entity, which is evaluated as a ranking problem. We perform link prediction on FB15k-237, WN18RR and UMLS datasets. For each correct triplet (h, r, t), either h or t is corrupted by replacing it with every other entity in the entity set E. These triplets are ranked based on scores produced by Eq. 1 of LASS. The evaluation is under the filtered setting (Bordes et al., 2013), i.e., removing all the triplets that appear either in the train, dev, or test set. Two common metrics, Mean Rank (MR) and Hits@10 (the proportion of correct entities ranked in the top 10) are used to evaluate the results. A lower MR is better while a higher Hits@10 is better. From the results in Table 4, we summarize key observations as below.

We find all our methods significantly outperform

the compared methods in MR, and reach competitive or better Hits@10. LASS-RoBERTa$_{LARGE}$ performs the best on WN18RR, which outperforms the best compared method StAR by 11 units in MR and 5.4% in Hits@10. It also delivers the best MR on FB15k-237. On UMLS, the existing state-of-the-art performance sets a high standard. However, LASS-BERT$_{BASE}$ still outperforms others by at least 0.08 unit in MR. The reasons for the improvements are mainly two-fold. (i) LASS is able to capture the structural patterns in the existing triplets to predict the missing ones via the structured loss. Compared to KG-BERT, LASS is able to use the neighboring entities in the KGs for the prediction. (ii) LASS is able to maintain the semantics of the KGs through semantic embedding to avoid unreasonable triplets with high ranks. For example, if *CEO_Of* holds between two entities, the *employee_Of* also holds, but *birth_Place* does not hold. This is the main reason that LASS outperforms all structure embedding based methods by a large margin especially in MR. For instance, LASS significantly outperforms TransE, which shares the similar structured loss with LASS. Compare to the improvements made on FB15k-237, LASS-RoBERTa$_{LARGE}$ has significantly improved the state-of-the-art results on WN18RR. The main reason leading to such significant improvements is that the pre-trained LMs provide more semantics in the semantic embedding for WordNet as those LMs are trained on textual corpora to capture rela-

tionships between words. While WordNet provides the relationships between words, FB15k-237 contains real-world entities and relations, which are less captured by the LMs.

We also notice that LASS only produces moderate Hits@10 on FB15k-237. The main reason is that FB15k-237 presents more complex relations between entities compared to other link prediction datasets shown in Table 1. Therefore, a more complex structured loss is expected for LASS to gain further improvements. We leave it as one of the future explorations. Besides, on FB15k-237 and WN18RR, LASS-BERT$_{LARGE}$ outperforms LASS-BERT$_{BASE}$, and LASS-RoBERTa$_{LARGE}$ also outperforms LASS-RoBERTa$_{BASE}$. This confirms the recent findings (Petroni et al., 2019) that larger LMs store more semantic knowledge in the parameters. We expect further improvements when larger LMs are used with LASS. On UMLS, we observe slightly different trends. This is mainly because UMLS is a relatively small dataset, thus large models can suffer from overfitting. Overall, RoBERTa improves the BERT pre-training procedure from several perspectives. The improved pre-training procedure enables RoBERTa to generate better performance in many downstream tasks. This suggests that an improved pre-training procedure can enrich the semantics learned in the corresponding LMs.

Both link prediction and triplet classification are core KG completion tasks. The results show that the proposed LASS generalizes well in KG completion tasks. Different from KG-BERT that designs different models for the tasks, our method does not introduce task-specific parameters or losses for different tasks.

3.5 Case Study

We show uncurated examples to illustrate why LASS can yield the above results, especially how the parameters of the LMs capture the semantics and structures. As attention layers are basic building blocks of the LMs, we focus on visualizing the attention weights with different input sequences.

We use BertViz (Vig, 2019) to illustrate the attention weights of the LMs. Given an example of a positive triplet, h = "symbololatry, the worship of symbols", r = "hypernym", and t = "veneration, religious zeal", Figure 3a shows the attention weights of the last layer of LASS-BERT$_{BASE}$ on WN11. We find that semantically related tokens
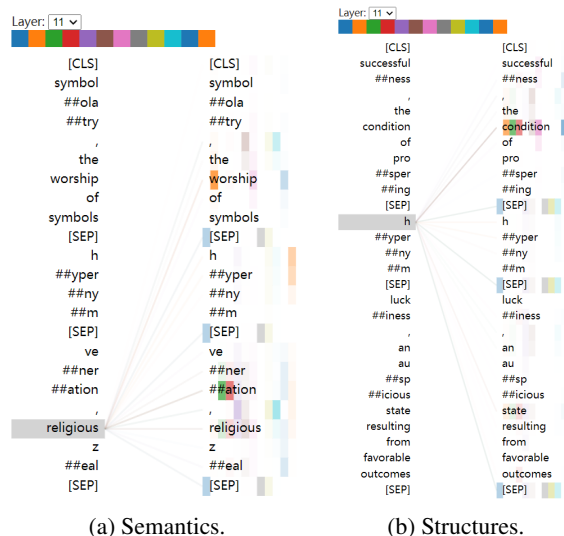

(a) Semantics. (b) Structures.

Figure 3: Illustration of attention weights of the last layer of LASS-BERT$_{BASE}$.

attend to each other with relatively high scores. For example, "religious" attends intensively to "worship" and "veneration". As in multi-head self attention (Vaswani et al., 2017), different attention heads in different colors attend to different aspects of the input, the heads are then concatenated to compute the final attention weights. The darker the color, the larger the attention score. This demonstrates that the semantic embedding of LASS is effective in capturing the semantics in the natural language description of the triplets.

We show another positive example with h = "successfulness, the condition of prospering", r = "hypernym", and t = "luckiness, an auspicious state resulting from favorable outcomes". Figure 3b illustrates the attention weights of the last layer of LASS-BERT$_{BASE}$ on WN11. We observe that tokens are highly attended to each other with similar structure roles in the triplet, even though they share fewer semantic similarities. For instance, the attention score between "hypernym" and "condition" is large. There is also a large attention score between "hypernym" and "state". This is because both "condition" and "state" capture the critical structure information of the triplet. The results indicate that the structure embedding of LASS is able to reconstruct the structure information in the semantic embeddings.

3.6 Error Analysis

To better understand the limitations of LASS, we perform a detailed analysis of the errors. We use triplet classification as an example. We investigate

the errors made by LASS-BERT$_\text{BASE}$ on WN11 and summarize the errors based on the relations in Table 5. We find most errors are caused by relations that are hard to be distinguished from each other due to their semantic similarities. For example, "domain topic" and "domain region" are such relations with an unclear semantic boundary.

Relation	Percentage (%)
domain topic	19.8
domain region	10.8
member meronym	9.1
has instance	8.4
has part	8.1
similar to	7.1
part of	6.3
synset domain topic	5.5
type of	4.6
member holonym	3.9
subordinate instance of	3.2

Table 5: Analysis of most common errors of LASS-BERT$_\text{BASE}$ categorized by relations on WN11.

4 Discussion

Structure Losses There are several directions to further improve LASS. LASS uses the probabilistic structured loss based on the score function of TransE, which learns a single representation for every entity and relation in the same embedding space. However, different relationships expect different entity embeddings. We propose to enable an entity to have distinct distributed representations when involved in different relations. For example, a new score function $\|\mathbf{h}_r + \mathbf{r} - \mathbf{t}_r\|_2^2$ models entities and relations in distinct spaces, and performs the translation between entity embeddings in relation space. The idea is in the same spirit as TransH (Wang et al., 2014b) and TransR (Lin et al., 2015). However, a downside of leveraging those losses is that they will bring additional computation overhead. Our method aims to trade off the computation costs and effectiveness. Exploring computation-light methods that involve alternative losses is one of the future investigations.

Pre-trained LMs We have explored two pre-trained LM families: BERT and RoBERTa. There are three possible directions along this line. First, as indicated in the experimental findings, larger LMs often store more semantics, which can improve the semantic embedding module of LASS.

We propose to examine larger pre-trained LMs, such as GPT-2 (Radford et al., 2019), GPT-3 (Brown et al., 2020), and Megatron-LM (Shoeybi et al., 2019). Incorporating longer language descriptions (e.g., Wikipedia page) of the entities in the knowledge graphs will provide richer knowledge for improved natural language understanding. Second, the fine-tuning procedure of the deep LMs for KG completion tasks, especially link prediction, is still computationally inefficient. Investigating light LM architectures, such as AL-BERT (Lan et al., 2020), to speed up the training process, is one of the promising directions. Finally, our proposed method is generally useful for many knowledge-driven downstream NLP tasks (e.g., question answering, factual probing) as well as low-resource NLP tasks. Ensembling our method with autoregressive models (e.g., GPT-2) will enable the method to perform text generation tasks.

5 Related Work

Pre-trained LMs Pre-trained LMs, such as BERT, have recently been used to obtain state-of-the-art results in many NLP benchmarks (Devlin et al., 2019; Liu et al., 2019). These models are usually based on Transformers (Vaswani et al., 2017) and trained on unlabeled text corpora. They are used to improve downstream tasks via embedding (Peters et al., 2018), fine-tuning (Radford et al., 2018), or few-shot learning (Radford et al., 2019). Fine-tuning bidirectional Transformers is the most widely used scheme in recent NLP applications, and the approach described in this paper is also based on this scheme. The main difference is that we design a structured loss on top of the LMs aiming to capture structures in natural language.

Knowledge Graph Embedding KG embedding aims to map entities and their relations to a continuous vector space. Traditional KG embedding methods represent each entity or each relation with a fixed vector. For any triplet (h,r,t), they use a scoring function $f(\mathbf{h}, \mathbf{r}, \mathbf{t})$ to model its likelihood. The scoring function of TransE (Bordes et al., 2013) is a negative translational distance. It can be augmented with different geometric transformations such as linear projections (Wang et al., 2014b; Lin et al., 2015) or rotations (Sun et al., 2019). Other models based on bilinear transformations (Yang et al., 2015), and convolutions (Dettmers et al., 2018), also show promising results on KG completion benchmarks. Our structured loss is motivated

by TransE. The main differences are the following. TransE (Bordes et al., 2013) treats the relation as a translation of the embeddings from the head to the tail. Therefore $\mathbf{h} + \mathbf{r} \approx \mathbf{t}$ when *(h, r, t)* holds. TransE designs a margin-based ranking loss based on the l_2 norm $\|\mathbf{h} + \mathbf{r} - \mathbf{t}\|_2^2$. The key differences between LASS and TransE are: (i) LASS leverages the natural language semantics in LMs, while TransE does not; (ii) LASS is a probabilistic structured loss, and is more computationally efficient and data-efficient compared to TransE. The main advantage of the probabilistic loss is that we eliminate the norm calculation that TransE requires to prevent the training process from trivially minimizing its loss by increasing the embeddings of entities or relations. The ranking loss of TransE calculates the loss of some training examples as zeros, which will not contribute to the optimization procedure. Our probabilistic loss makes use of all the training examples. Besides, we introduce corrupted relations in the loss, which provides more flexibility in incorporating the KG structure.

Traditional KG embedding approaches aforementioned regard entities and relations as basic units, without using any extra information. However, studies (Socher et al., 2013; Wang et al., 2014a; Xie et al., 2016) show that a KG model that models the natural language descriptions of entities and relations usually outperforms those methods that only model the structure of knowledge triplets. Petroni et al. (2019) use LMs as virtual KGs to answer factual questions. ERNIE (Zhang et al., 2019b) integrates structural KGs into pre-trained models to improve knowledge-driven NLP tasks. By contrast, we aim to combine both the structures and semantics of the KGs via a unified optimization procedure for the task of KG completion. KG-BERT (Yao et al., 2019) models KG completion tasks as sentence classification tasks and solves them by fine-tuning pre-trained LMs. There are several key differences between our LASS and KG-BERT (Yao et al., 2019): (i) LASS reconstructs the structures of KGs via structure embedding, while KG-BERT does not; (ii) LASS unifies the link prediction and triplet classification under the same architecture, while KG-BERT designs different architectures for different tasks; (iii) LASS works with two families of LMs, while KG-BERT only works with BERT$_{\text{BASE}}$. LASS is not particularly designed for BERT, shedding light on understanding the role of semantics in LMs for KG comple-

tion.

6 Conclusion

We propose a new embedding method that leverages both semantics and structures of the knowledge graphs for the task of knowledge graph completion, and offers additional benefits in low-resource settings. The method maps a knowledge graph triplet to an embedding space via fine-tuning language models, where the forward pass captures semantics and the loss reconstructs structures. Our method has shown significant improvements on knowledge graph completion benchmarks. The implementation has made no modifications to the language model architectures. The results suggest that the learned embeddings are generally useful in downstream knowledge-driven applications, and potentially useful for more natural language understanding tasks. We hope our results will foster further research in this direction.

7 Ethical Considerations

We hereby acknowledge that all of the co-authors of this work are aware of the provided *ACM Code of Ethics* and honor the code of conduct. The followings give the aspects of both our ethical considerations and our potential impacts to the community. This work uses pre-trained LMs for knowledge graph completion. The risks and potential misuse of LMs are discussed in Brown et al. (2020). There are potential undesirable biases in the datasets, such as unfaithful descriptions from Wikipedia. We do not anticipate the production of harmful outputs after using our model, especially towards vulnerable populations.

8 Environmental Considerations

We use BERT and RoBERTa as our pre-trained LMs. According to the estimation in Strubell et al. (2019), pre-training a base model costs 1,507 kWh·PUE and emits 1,438 lb CO_2, while pre-training a large model requires 4 times the resources of a base model. In addition, our fine-tuning takes less than 1% gradient-steps of the number of steps of pre-training. Therefore, our energy cost and CO_2 emissions are relatively small. Besides, the results in the low-resource settings show that our method has better sampling efficiency. This indicates that we can further reduce energy consumption when training with fewer data.

Acknowledgement

References

Bo An, Bo Chen, Xianpei Han, and Le Sun. 2018. Accurate text-enhanced knowledge graph representation learning. In *NAACL-HLT*, pages 745–755.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *SIGMOD*, pages 1247–1250.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *NeurIPS*, pages 2787–2795.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *NeurIPS*, pages 1877–1901.

Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R. Hruschka, and Tom M. Mitchell. 2010. Toward an architecture for never-ending language learning. In *AAAI*, pages 1306–1313.

Ines Chami, Adva Wolf, Da-Cheng Juan, Frederic Sala, Sujith Ravi, and Christopher Ré. 2020. Low-dimensional hyperbolic knowledge graph embeddings. In *ACL*, pages 6901–6914.

Yihong Chen, Pasquale Minervini, Sebastian Riedel, and Pontus Stenetorp. 2021. Relation prediction as an auxiliary training objective for improving multi-relational graph representations. In *AKBC*.

Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2d knowledge graph embeddings. In *AAAI*, pages 1811–1818.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, pages 4171–4186.

Kelvin Guu, John Miller, and Percy Liang. Traversing knowledge graphs in vector space. In *EMNLP*, pages 318–327.

Yanchao Hao, Yuanzhe Zhang, Kang Liu, Shizhu He, Zhanyi Liu, Hua Wu, and Jun Zhao. 2017. An end-to-end model for question answering over knowledge base with cross-attention combining global knowledge. In *ACL*, pages 221–231.

Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Knowledge graph embedding via dynamic mapping matrix. In *ACL-IJCNLP*, pages 687–696.

Guoliang Ji, Kang Liu, Shizhu He, and Jun Zhao. 2016. Knowledge graph completion with adaptive sparse transfer matrix. In *AAAI*, pages 985–991.

Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and S Yu Philip. 2021. A survey on knowledge graphs: Representation, acquisition, and applications. *TNNLS*, 33:494–514.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. In *ICLR*.

Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion. In *AAAI*, pages 2181–2187.

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

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *ICLR*.

Alexa T. McCray. 2003. An upper-level ontology for the biomedical domain. *Comparative and Functional Genomics*, pages 80–84.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *NeurIPS*, pages 3111–3119.

George A. Miller. 1994. Wordnet: A lexical database for english. In *HLT*.

Deepak Nathani, Jatin Chauhan, Charu Sharma, and Manohar Kaul. 2019. Learning attention-based embeddings for relation prediction in knowledge graphs. In *ACL*, pages 4710–4723.

Dai Quoc Nguyen, Tu Dinh Nguyen, Dat Quoc Nguyen, and Dinh Phung. 2018. A novel embedding model for knowledge base completion based on convolutional neural network. In *NAACL-HLT*, pages 327–333.

Thomas Pellissier Tanon, Denny Vrandečić, Sebastian Schaffert, Thomas Steiner, and Lydia Pintscher. 2016. From freebase to wikidata: The great migration. In *WWW*, pages 1419–1428.

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

Fabio Petroni, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H. Miller, and Sebastian Riedel. 2019. Language models as knowledge bases? In *EMNLP-IJCNLP*, pages 2463–2473.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. *OpenAI blog*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI blog*.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. 2019. Megatron-lm: Training multibillion parameter language models using model parallelism. *CoRR*.

Richard Socher, Danqi Chen, Christopher D. Manning, and Andrew Y. Ng. 2013. Reasoning with neural tensor networks for knowledge base completion. In *NeurIPS*, pages 926–934.

Shashank Sonkar, Arzoo Katiyar, and Richard G. Baraniuk. 2021. Neptune: Neural powered tucker network for knowledge graph completion. *CoRR*.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and policy considerations for deep learning in NLP. In *ACL*, pages 3645–3650.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019. Rotate: Knowledge graph embedding by relational rotation in complex space. In *ICLR*.

Kristina Toutanova and Danqi Chen. 2015. Observed versus latent features for knowledge base and text inference. In *CVSC-WS*, pages 57–66.

Kristina Toutanova, Danqi Chen, Patrick Pantel, Hoifung Poon, Pallavi Choudhury, and Michael Gamon. 2015. Representing text for joint embedding of text and knowledge bases. In *EMNLP*, pages 1499–1509.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Eric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *ICML*, pages 2071–2080.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NeurIPS*, pages 5998–6008.

Jesse Vig. 2019. A multiscale visualization of attention in the transformer model. In *ACL*, pages 37–42.

Bo Wang, Tao Shen, Guodong Long, Tianyi Zhou, and Yi Chang. 2021. Structure-Augmented Text Representation Learning for Efficient Knowledge Graph Completion. In *WWW*, pages 1737–1748.

Haoyu Wang, Vivek Kulkarni, and William Yang Wang. 2018. Dolores: Deep contextualized knowledge graph embeddings. In *AKBC*.

Rui Wang, Bicheng Li, Shengwei Hu, Wenqian Du, and Min Zhang. 2020. Knowledge Graph Embedding via Graph Attenuated Attention Networks. *IEEE Access*, 8:5212–5224.

Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014a. Knowledge graph and text jointly embedding. In *EMNLP*, pages 1591–1601.

Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014b. Knowledge graph embedding by translating on hyperplanes. In *AAAI*, pages 1112–1119.

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

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and et al. 2020. Huggingface's transformers: State-of-the-art natural language processing. *CoRR*.

Han Xiao, Minlie Huang, and Xiaoyan Zhu. 2016. Transg: A generative model for knowledge graph embedding. In *ACL*, pages 2316–2325.

Ruobing Xie, Zhiyuan Liu, Jia Jia, Huanbo Luan, and Maosong Sun. 2016. Representation learning of knowledge graphs with entity descriptions. In *AAAI*, pages 2659–2665.

Chenyan Xiong, Russell Power, and Jamie Callan. 2017. Explicit semantic ranking for academic search via knowledge graph embedding. In *WWW*, pages 1271–1279.

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding entities and relations for learning and inference in knowledge bases. In *ICLR*.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. Kg-bert: Bert for knowledge graph completion. *CoRR*.

Zhanqiu Zhang, Jianyu Cai, and Jie Wang. 2020. Duality-induced regularizer for tensor factorization based knowledge graph completion. In *NeurIPS*, pages 21604–21615.

Zhanqiu Zhang, Jianyu Cai, Yongdong Zhang, and Jie Wang. 2019a. Learning Hierarchy-Aware Knowledge Graph Embeddings for Link Prediction. In *AAAI*, pages 3065–3072.

Zhao Zhang, Fuzhen Zhuang, Meng Qu, Fen Lin, and Qing He. 2018. Knowledge graph embedding with hierarchical relation structure. In *EMNLP*, pages 3198–3207.

Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019b. Ernie: Enhanced language representation with informative entities. In *ACL*, pages 1441–1451.

A Experimental Setup Details

We describe additional details of our experimental setup including datasets and comparison methods in this section.

A.1 Datasets

We introduce the link prediction and triplet classification datasets as below.

A.1.1 Link Prediction

- **FB15k-237**. Freebase is a large collaborative knowledge graph consisting of data composed mainly by its community members. It is an online collection of structured data harvested from many sources, including individual and user-submitted wiki contributions (Pellissier Tanon et al., 2016). FB15k is a selected subset of Freebase that consists of 14,951 entities and 1,345 relationships (Bordes et al., 2013). FB15K-237 is a variant of FB15K where inverse relations and redundant relations are removed, resulting in 237 relations (Toutanova et al., 2015).

- **WN18RR**. WordNet is a lexical database of semantic relations between words in English. WN18 (Bordes et al., 2013) is a subset of WordNet which consists of 18 relations and 40,943 entities. WN18RR is created to ensure that the evaluation dataset does not have inverse relations to prevent test leakage (Dettmers et al., 2018).

- **UMLS**. UMLS semantic network (McCray, 2003) is an upper-level ontology of Unified Medical Language System. The semantic network, through its 135 semantic types, provides a consistent categorization of all concepts represented in the UMLS. The 46 links between the semantic

Method	Score Function	
TransE	$-\|\mathbf{h}+\mathbf{r}-\mathbf{t}\|$	$\mathbf{h},\mathbf{r},\mathbf{t}\in\mathbb{R}^k$
TransH	$-\left\|(\mathbf{h}-\mathbf{w}_r^\top\mathbf{h}\mathbf{w}_r)+\mathbf{r}-(\mathbf{t}-\mathbf{w}_r^\top\mathbf{t}\mathbf{w}_r)\right\|$	$\mathbf{h},\mathbf{t},\mathbf{r},\mathbf{w}_r\in\mathbb{R}^k$
TransR	$-\|\mathbf{M}_r\mathbf{h}+\mathbf{r}-\mathbf{M}_r\mathbf{t}\|$	$\mathbf{h},\mathbf{t}\in\mathbb{R}^k,\mathbf{M}_r\in\mathbb{R}^{k\times d}$
TransD	$-\left\|(\mathbf{w}_r\mathbf{w}_h^\top+\mathbf{I})\mathbf{h}+\mathbf{r}-(\mathbf{w}_r\mathbf{w}_t^\top+\mathbf{I})\mathbf{t}\right\|$	$\mathbf{h},\mathbf{t},\mathbf{w}_h\mathbf{w}_t\in\mathbb{R}^k,\mathbf{r},\mathbf{w}_r\in\mathbb{R}^d$
TransG	$\sum_i\pi_r^i\exp\left(-\frac{\|\boldsymbol{\mu}_h+\boldsymbol{\mu}_r^i-\boldsymbol{\mu}_t\|}{\sigma_h^2+\sigma_t^2}\right)$	$\mathbf{h}\sim\mathcal{N}\left(\boldsymbol{\mu}_h,\sigma_h^2\mathbf{I}\right),\mathbf{t}\sim\mathcal{N}\left(\boldsymbol{\mu}_t,\Sigma_t\right),\boldsymbol{\mu}_h,\boldsymbol{\mu}_t\in\mathbb{R}^k$
TranSparse-S	$-\|\mathbf{M}_r\left(\theta_r\right)\mathbf{h}+\mathbf{r}-\mathbf{M}_r\left(\theta_r\right)\mathbf{t}\|_{1/2}^2-\|\mathbf{M}_r^1\left(\theta_r^1\right)\mathbf{h}+\mathbf{r}-\mathbf{M}_r^2\left(\theta_r^2\right)\mathbf{t}\|_{1/2}^2$	$\mathbf{h},\mathbf{t}\in\mathbb{R}^k,\mathbf{r}\in\mathbb{R}^d,\mathbf{M}_r\left(\theta_r\right)\in\mathbb{R}^{k\times d},\mathbf{M}_r^1\left(\theta_r^1\right),\mathbf{M}_r^2\left(\theta_r^2\right)\in\mathbb{R}^{k\times d}$
DistMult	$\langle\mathbf{r},\mathbf{h},\mathbf{t}\rangle$	$\mathbf{h},\mathbf{r},\mathbf{t}\in\mathbb{R}^k$
ConvKB	$\text{concat}(g([\boldsymbol{h},\boldsymbol{r},\boldsymbol{t}]*\omega))\mathbf{w}$	$\mathbf{h},\mathbf{r},\mathbf{t}\in\mathbb{R}^k$
ComplEx	$\Re(\langle\mathbf{r},\mathbf{h},\bar{\mathbf{t}}\rangle)$	$\mathbf{h},\mathbf{r},\mathbf{t}\in\mathbb{C}^k$
ConvE	$\langle\sigma(\text{vec}(\sigma([\bar{\mathbf{r}},\bar{\mathbf{h}}]*\boldsymbol{\Omega}))\mathbf{W}),\mathbf{t}\rangle$	$\mathbf{h},\mathbf{r},\mathbf{t}\in\mathbb{R}^k$
RotatE	$-\|\mathbf{h}\circ\mathbf{r}-\mathbf{t}\|^2$	$\mathbf{h},\mathbf{r},\mathbf{t}\in\mathbb{C}^c,|r_i|=1$
REFE	$-\text{arctanh}(\|-\langle\mathbf{h},\text{Ref}(\mathbf{r})\rangle\oplus^c\mathbf{t}\|)$	$\mathbf{h},\mathbf{r},\mathbf{t}\in\mathbb{R}^k$
HAKE	$\text{RotatE}-\|\sin((\mathbf{h}+\mathbf{r}-\mathbf{t})/2)\|_1$	$\mathbf{h},\mathbf{r},\mathbf{t}\in\mathbb{R}^k$
ComplEx-DURA	$\text{ComplEx}-\langle\mathbf{h},\mathbf{r}\rangle^2-\|\mathbf{t}\|^2$	$\mathbf{h},\mathbf{r},\mathbf{t}\in\mathbb{C}^k$

Table 6: The score functions $f_r(\mathbf{h},\mathbf{t})$ of shallow structure embedding models for knowledge graph embedding, where $\langle\cdot\rangle$ denotes the generalized dot product, $\circ$ denotes the Hadamard product, σ denotes activation function and $*$ denotes 2D convolution. $\bar{\cdot}$ denotes conjugate for complex vectors, and 2D reshaping for real vectors in the ConvE model. $\text{Ref}(\theta)$ denotes the reflection matrix induced by rotation parameters θ. $\oplus^c$ is Möbius addition that provides an analogue to Euclidean addition for hyperbolic space.

types provide the structure for the network and represent important relationships in the biomedical domain.

A.1.2 Triplet Classification

• **WN11 and FB13** are subsets of WordNet and FreeBase respectively for triplet classification, where Socher et al. (2013) randomly switch entities from correct testing triplets resulting in a total of doubling the number of test triplets with an equal number of positive and negative examples.

A.2 Comparison Methods

We compare LASS to three types of knowledge graph completion methods: shallow structure embedding, deep structure embedding, and language semantic embedding.[1]

A.2.1 Shallow Structure Embedding

TransE (Bordes et al., 2013), TransH (Wang et al., 2014b), TransR (Lin et al., 2015), TransD (Ji et al., 2015), TransG (Xiao et al., 2016), TranSparse-S (Ji et al., 2016), DistMult (Yang et al., 2015), ConvKB (Nguyen et al., 2018), ComplEx (Trouillon et al., 2016), ConvE (Dettmers et al., 2018), RotatE (Sun et al., 2019), REFE (Chami et al., 2020), HAKE (Zhang et al., 2019a), and ComplEx-DURA (Zhang et al., 2020) are methods based only on the structure of the knowledge graphs. DistMult-HRS (Zhang et al., 2018) is an extension of DistMult which is combined with a three-layer hierarchical relation structure (HRS) loss. Each of these

methods proposes a scoring function regarding a knowledge triplet, without using the natural language descriptions or names of entities or relations. The scoring functions are shown in Table 6.

A.2.2 Deep Structure Embedding

• **NTN** (Neural Tensor Network) (Socher et al., 2013) models entities across multiple dimensions by a bilinear tensor neural layer.

• **DOLORES** (Wang et al., 2018) is based on bidirectional LSTMs and learns deep representations of entities and relations from constructed entity-relation chains.

• **KBGAT** proposes an attention-based feature embedding that captures both entity and relation features in any given entity's neighborhood, and additionally encapsulates relation clusters and multi-hop relations (Nathani et al., 2019).

• **GAATs** integrates an attenuated attention mechanism in a graph neural network to assign different weights in different relation paths and acquire the information from the neighborhoods (Wang et al., 2020).

• **NePTuNe** takes advantage of both TuckER and NTN by carefully crafted nonlinearities and a shared core tensor intrinsic to the Tucker decomposition (Sonkar et al., 2021).

• **ComplEx-N3-RP** introduces an auxiliary training task to predict relation types as a self-supervised objective. (Chen et al., 2021).

---

[1]We refer the readers to (Ji et al., 2021) for a more comprehensive review of the knowledge graph completion methods.

A.2.3 Language Semantic Embedding

- **TEKE** (Wang and Li, 2016) takes advantage of the context information in a text corpus. The textual context information is incorporated to expand the semantic structure of the knowledge graph and each relation is enabled to own different representations for different head and tail entities.

- **AATE** (An et al., 2018) is a text-enhanced knowledge graph representation learning method, which can represent a relation/entity with different representations in different triples by exploiting additional textual information.

- **KG-BERT** (Yao et al., 2019) considers triples in knowledge graphs as textual sequences, where each textual sequence is a concatenation of text descriptions of the head entity, the relation, and the tail entity. Then KG-BERT treats the knowledge graph completion task as a text binary classification task, and then solves it by fine-tuning a pre-trained BERT.

- **StAR** (Wang et al., 2021) partitions each triplet into two asymmetric parts as in translation-based graph embedding approach, and encodes both parts into contextualized representations by a Siamese-style textual encoder (BERT or RoBERTa).