Structure-Augmented Text Representation Learning for Efficient Knowledge Graph Completion

Bo Wang[1], Tao Shen[2], Guodong Long[2], Tianyi Zhou[3], Ying Wang[4*], Yi Chang[1,5*]
[1]School of Artificial Intelligence, Jilin University
[2]Australian AI Institute, School of CS, FEIT, University of Technology Sydney
[3]Paul G. Allen School of Computer Science & Engineering, University of Washington, Seattle
[4]College of Computer Science and Technology, Jilin University
[5]International Center of Future Science, Jilin University
bowang19@mails.jlu.edu.cn,tao.shen@student.uts.edu.au,guodong.long@uts.edu.au,
tianyizh@uw.edu,wangying2010@jlu.edu.cn,yichang@jlu.edu.cn

ABSTRACT

Human-curated knowledge graphs provide critical supportive information to various natural language processing tasks, but these graphs are usually incomplete, urging auto-completion of them (a.k.a. knowledge graph completion). Prevalent graph embedding approaches, e.g., TransE, learn structured knowledge via representing graph elements (i.e., entities/relations) into dense embeddings and capturing their triple-level relationship with spatial distance. However, they are hardly generalizable to the elements never visited in training and are intrinsically vulnerable to graph incompleteness. In contrast, textual encoding approaches, e.g., KG-BERT, resort to graph triple's text and triple-level contextualized representations. They are generalizable enough and robust to the incompleteness, especially when coupled with pre-trained encoders. But two major drawbacks limit the performance: (1) high overheads due to the costly scoring of all possible triples in inference, and (2) a lack of structured knowledge in the textual encoder. In this paper, we follow the textual encoding paradigm and aim to alleviate its drawbacks by augmenting it with graph embedding techniques – a complementary hybrid of both paradigms. Specifically, we partition each triple into two asymmetric parts as in translation-based graph embedding approach, and encode both parts into contextualized representations by a Siamese-style textual encoder. Built upon the representations, our model employs both deterministic classifier and spatial measurement for representation and structure learning respectively. It thus reduces the overheads by reusing graph elements' embeddings to avoid combinatorial explosion, and enhances structured knowledge by exploring the spatial characteristics. Moreover, we develop a self-adaptive ensemble scheme to further improve the performance by incorporating triple scores from an existing graph embedding model. In experiments, we achieve state-of-the-art performance on three benchmarks and a zero-shot dataset for link prediction, with highlights of inference costs reduced by 1-2 orders of magnitude compared to a sophisticated textual encoding method.

*Joint Corresponding Author.

ACM ISBN 978-1-4503-8312-7/21/04.
https://doi.org/10.1145/3442381.3450043

ACM Reference Format:
Bo Wang, Tao Shen, Guodong Long, Tianyi Zhou, Ying Wang, Yi Chang. 2021. Structure-Augmented Text Representation Learning for Efficient Knowledge Graph Completion. In *Proceedings of the Web Conference 2021 (WWW '21), April 19–23, 2021, Ljubljana, Slovenia.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3442381.3450043

1 INTRODUCTION

Knowledge graph (KG) is a ubiquitous format of knowledge base (KB). It is structured as a directed graph whose vertices and edges respectively stand for entities and their relations. It is usually represented as a set of triples in the form of (*head entity, relation, tail entity*), or (*h, r, t*) for short. KGs as supporting knowledge play significant roles across a wide range of natural language processing (NLP) tasks, such as dialogue system [15], information retrieval [40], recommendation system [46], etc. However, human-curated knowledge graphs usually suffer from incompleteness [29], inevitably limiting their practical applications. To mitigate this issue, knowledge graph completion (KGC) aims to predict the missing triples in a knowledge graph. In this paper, we particularly target *link prediction* task for KGC, whose goal is to predict the missing *head* (*tail*) given the *relation* and *tail* (*head*) in a triple.

It is noteworthy that KG [2, 30, 36] is usually at a scale of billions and the number of involved entities is up to millions, so most graph neural networks (e.g., GCN [16]) operating on the whole graph are not scalable in computation. Thus, approaches for KGC often operate at triple level, which can be grouped into two paradigms, i.e., *graph embedding* and *textual encoding* approaches.

Graph embedding approaches attempt to learn the representations for graph elements (i.e., entities/relations) as low-dimension vectors by exploring their structured knowledge in a KG. Typically, they directly exploit the spatial relationship of the three embeddings in a triple to learn structured knowledge, which can be classified into two sub-categories. (1) *Translation-based* approaches, e.g., TransE [3] and RotatE [31], score the plausibility of a triple by applying a translation function to the embeddings of head and relation, and then measuring how close the resulting embedding to the tail embedding, i.e., $-||g(\boldsymbol{h}, \boldsymbol{r}) - \boldsymbol{t}||_p$; And (2) *semantic matching* approaches, e.g., DistMult [44] and QuatE [47], derive the plausibility of a graph triple via a matching function that directly operates on

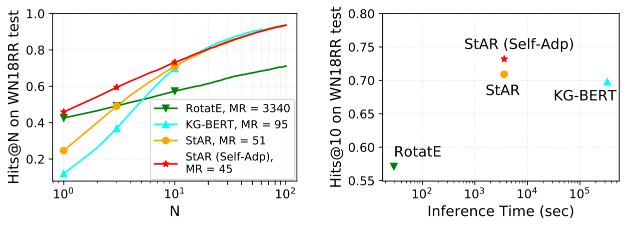

Figure 1: Summary comparisons on WN18RR test. RotatE and KG-BERT are state-of-the-art approaches of *graph embedding* and *textual encoding* paradigms respectively. StAR is our model, and "StAR (Self-Adp)" is our model plus our proposed self-adaptive ensemble.

the triple, i.e., $f(\boldsymbol{h}, \boldsymbol{r}, \boldsymbol{t})$. Despite their success in structure learning, they completely ignore contextualized information and thus have several drawbacks: (1) The trained models are not applicable to entities/relations unseen in training; And (2) they are intrinsically vulnerable to the graph incompleteness. These drawbacks severely weaken their generalization capability and prediction quality.

Textual encoding[1] approaches, e.g., KG-BERT [45], predict the missing parts for KGC using the contextualized representation of triples' natural language text. The text can refer to textual contents of entities and relations (e.g., their names or descriptions). Coupled with pre-trained word embedding [19, 24] or language model [12, 17], the textual encoder can easily generalize to unseen graph elements and is invulnerable to graph incompleteness issue. However, they are limited by two inherent constraints: (1) Applying textual encoder to link prediction requires costly inference on all possible triples, causing a combinatorial explosion; (2) The textual encoder is incompetent in structure learning, leading to a lack of structured knowledge and the entity ambiguity problem [10].

The experiments of these two paradigms also reflect their individual Pros and Cons. As shown in Figure 1 (left), KG-BERT achieves a high Hits@N (i.e., Top-N recall) when N is slightly large, but fails for small N due to entity ambiguity problem. In contrast, RotatE achieves a high Hits@1/@3 since it purely learns from structured knowledge without exposure to the ambiguity problem. But it still underperforms due to a lack of text contextualized information. And in Figure 1 (right), although KG-BERT outperforms RotatE, it requires much higher overheads due to combinatorial explosion.

Therefore, it is natural to integrate both the contextualized and structured knowledge into one model, while in previous works they are respectively achieved by textual encoding and graph embedding paradigms. To this end, we start with a textual encoding paradigm with better generalization and then aim at alleviating its intrinsic drawbacks, i.e., overwhelming overheads and insufficient structured knowledge. Specifically, taking the inspiration from translation-based graph embedding approach (e.g., TransE), we first partition each triple into two parts: one with a combination of *head* and *relation*, while the other with *tail*. Then, by applying a Siamese-style textual encoder to their text, we encode each part into separate contextualized representation. Lastly, we concatenate the two representations in an interactive manner [26] to form the final representation of the triple and train a binary neural classifier upon it. In the meantime, as we encode the triple by separated parts, we can measure their spatial relations like translation function [3, 31] and then conduct a structure learning using contrastive objective.

Consequently, on the one hand, our model can re-use the same graph elements' embeddings for different triples to avoid evaluating the combinatorial number of triples required in link prediction. On the other hand, it also augments the textual encoding paradigm by modeling structured knowledge, which is essential to graph-related tasks. In addition, our empirical studies on link prediction show that introducing such structured knowledge can effectively reduce false positive predictions and help entity disambiguation. As shown in Figure 1, our model improves the KG-BERT baseline in both performance and efficiency, but given a small N (e.g., $\leq$ 2), the performance is not that satisfactory. Motivated by this, we propose a self-adaptive ensemble scheme that incorporates our model's outputs with the triple scores produced by an existing graph embedding model (e.g., RotatE). Thereby, we can benefit from the advantages of both the graph embedding and textual encoding. Hence, as shown in Figure 1, our model plus the proposed self-adaptive ensemble with RotatE achieves more. Our main contributions are:

- We propose a hybrid model of *textual encoding* and *graph embedding* paradigms to learn both contextualized and structured knowledge for their mutual benefits: A Siamese-style textual encoder generalizes graph embeddings to unseen entities/relations, while augmenting it with structure learning contributes to entity disambiguation and high efficiency.
- We develop a self-adaptive ensemble scheme to merge scores from graph embedding approach and boost the performance.
- We achieve state-of-the-art results on three benchmarks and a zero-shot dataset; We show a remarkable speedup (6.5h vs. 30d on FB15k-237 [33]) over recent KG-BERT [45]; We provide a comparative analysis of the two paradigms.

2 BACKGROUND

We start this section with a formal definition of the link prediction task for KGC. Then, we summarize the pre-trained masked language model and its fine-tuning. And lastly we give a brief introduction to a state-of-the-art textual encoding approach, KG-BERT [45].

Link Prediction. Formally, a KG $\mathcal{G} = \{\mathcal{E}, \mathcal{R}\}$ consists of a set of triples (h, r, t), where $h, t \in \mathcal{E}$ are head and tail entity respectively while $r \in \mathcal{R}$ is the relation between them. Given a head h (tail t) and a relation r, the goal of link prediction is to find the most accurate tail t (head h) from $\mathcal{E}$ to make a new triple (h, r, t) plausible in $\mathcal{G}$. And during inference, given an incomplete triple $(h, r, ?)$ for example, a trained model is asked to score all candidature triples $\{(h, r, t')|t' \in \mathcal{E}\}$ and required to rank the only oracle triple (h, r, t^*) as high as possible. This is why the combinatorial explosion appears in a computation-intensive model defined at triple level.

Pre-Trained Masked Language Model. To obtain powerful textual encoders, masked language models (MLMs) pre-trained on large-scale raw corpora learn generic contextualized representations in a self-supervised fashion (e.g., BERT [12] and RoBERTa [17]). MLMs randomly mask some tokens and predict the masked tokens by considering their contexts on both sides. Specifically, given tokenized

[1]"*Textual encoding*" in this paper refers to capturing contextualized information across entities and relations in a triple [45], despite previous works (as detailed in §3.5) using the text of a stand-alone entity/relation to enhance corresponding graph embedding.

text $[w_1, \dots, w_n]$, a certain percentage (e.g., 15% in [12]) of the original tokens are then masked and replaced: of those, 80% with special token `[MASK]`, 10% with a token sampled from the vocabulary $\mathbb{V}$, and the remaining kept unchanged. The masked sequence of embedded tokens $[\boldsymbol{x}_1^{(m)}, \dots, \boldsymbol{x}_n^{(m)}]$ is passed into a Transformer encoder [35] to produce contextualized representations for the sequence:

$$\boldsymbol{C} = \text{Transformer-Enc}([\boldsymbol{x}_1^{(m)}, \dots, \boldsymbol{x}_n^{(m)}]) \in \mathbb{R}^{d_h \times n}. \tag{1}$$

The pre-training loss is defined as

$$\mathcal{L}^m = -\frac{1}{|\mathcal{M}|} \sum\nolimits_{i \in \mathcal{M}} \log P(w_i | \boldsymbol{C}_{:,i}), \tag{2}$$

where $\mathcal{M}$ is the set of masked token indices, and $P(w_i|\boldsymbol{C}_{:,i})$ is the probability of predicting the masked w_i. After pre-trained, they act as initializations of textual encoders, performing very well on various NLP tasks with task-specific modules and fine-tuning [12].

KG-BERT. As a recent textual encoding approach [45] for KGC, instead of using embeddings of entities/relations, it scores a triple upon triple-level contextualized representation. Specifically, a tokenizer with a word2vec [19, 24] first transforms the text x of each entity/relation to a sequence of word embeddings $\boldsymbol{X} = [\boldsymbol{x}_1, \dots, \boldsymbol{x}_n] \in \mathbb{R}^{d \times n}$. So, the text of a triple $(x^{(h)}, x^{(r)}, x^{(t)})$ can be denoted as $(\boldsymbol{X}^{(h)}, \boldsymbol{X}^{(r)}, \boldsymbol{X}^{(t)})$. Then, KG-BERT applies the Transformer encoder [35] to a concatenation of the *head*, *relation* and *tail*. The encoder is initialized by a pre-trained MLM, BERT, and the concatenation is $\tilde{\boldsymbol{X}} = [\boldsymbol{x}^{\texttt{[CLS]}}, \boldsymbol{X}^{(h)}, \boldsymbol{x}^{\texttt{[SEP]}}, \boldsymbol{X}^{(r)}, \boldsymbol{x}^{\texttt{[SEP]}}, \boldsymbol{X}^{(t)}, \boldsymbol{x}^{\texttt{[SEP]}}]$, where `[CLS]` and `[SEP]` are special tokens defined by Devlin et al. [12]. Based on this, KG-BERT produces a contextualized representation $\boldsymbol{c}$ for the entire triple, i.e.,

$$\boldsymbol{c} = \text{Pool}(\text{Transformer-Enc}(\tilde{\boldsymbol{X}})), \tag{3}$$

where $\text{Pool}(\cdot)$, defined in [12], collects the resulting of `[CLS]` to denote a contextualized representation for the sequence. Next, $\boldsymbol{c}$ is passed into a two-way classifier to determine if the triple is plausible or not. Lastly, the model is fine-tuned by minimizing a cross entropy loss. In inference, positive probability (i.e., confidence) of a triple is used as a ranking score. Such a simple approach shows its effectiveness for KGC, highlighting the significance of text representation learning. We thus follow this line and propose our model by avoiding combinatorial explosion and enhancing structure learning.

3 PROPOSED APPROACH

In this section, we first elaborate on a structure-aware triple encoder (§3.1) and a structure-augmented triple scoring module (§3.2), which compose our **St**ructure-**A**ugmented Text **R**epresentation (StAR) model to tackle link prediction for KBC (as illustrated in Figure 2). And we provide the details about training and inference, e.g., training objectives and efficiency, in §3.3. Then, we develop a self-adaptive ensemble scheme in §3.4 to make the best of an existing graph embedding approach and boost the performance. Lastly, in §3.5, we provide comparative analyses between our model and previous text-based approaches for graph-related tasks.

3.1 Structure-Aware Triple Encoding

In this subsection, we aim at encoding a graph triple into vector representation(s) in latent semantic space, with consideration of subsequent structure learning and inference speedup. The representation(s), similar to graph embeddings, can be fed into any downstream objective-specific module to fulfil triple scoring.

Recently, to accelerate the inference of a deep Transformer-based model [12, 35] in an information retrieval (IR) task, Reimers and Gurevych [26] adopted a two-branch Siamese architecture [9] to bypass pairwise input via encoding the query and candidate separately. This enables pre-computing the representations for all candidates and uses a light-weight matching net [26] to calculate relatedness. We take this inspiration to link prediction to avoid combinatorial explosion, but several open questions arise: (1) How to preserve contextualized knowledge across the entities and relation in a triple; (2) How to apply Siamese architecture to a triple with three components; and (3) How to facilitate structure learning in downstream modules.

These questions can be dispelled by digesting several techniques from translation-based graph embedding approaches, e.g., TransE [3] and RotatE [31]. The techniques include applying a *translation function* to the embeddings of *head* and *relation*, and structure learning via exploring spatial relationship (e.g., distance) between the function's output and *tail* embedding.

Specifically, TransE and RotatE explicitly define the translation function as real vector addition and complex vector product, respectively. In contrast, as a textual encoding approach, we implicitly formulate the function as applying a Transformer-based encoder to a text concatenation of *head* and *relation*. The concatenation is:

$$\tilde{\boldsymbol{X}}^{(h)} = [\boldsymbol{x}^{\texttt{[CLS]}}, \boldsymbol{X}^{(h)}, \boldsymbol{x}^{\texttt{[SEP]}}, \boldsymbol{X}^{(r)}, \boldsymbol{x}^{\texttt{[SEP]}}], \tag{4}$$

where $\boldsymbol{x}^{\texttt{[CLS]}}$ and $\boldsymbol{x}^{\texttt{[SEP]}}$ are embedded special tokens defined in [12]. Refer to KG-BERT in §2 for the details about pre-processing. Then, such "contextualizing" translation function is defined as

$$\boldsymbol{u} = \text{Pool}(\text{Transformer-Enc}(\tilde{\boldsymbol{X}}^{(h)})), \tag{5}$$

where $\text{Transformer-Enc}(\cdot)$ denotes the Transformer encoder consisting of multi-head self-attention layers[35]. We keep using the segment identifier given by Devlin et al. [12] to mark if a token is from an entity (i.e., 0) or a relation (i.e., 1). And again, $\text{Pool}(\cdot)$ collects the resulting of `[CLS]` to denote sequence-level contextualized representation. So, $\boldsymbol{u}$, a contextualized representation across *head* and *relation*, can be viewed as the translation function's output.

On the other side, we also encode *tail* by applying the Transformer encoder to its text, which is written as

$$\boldsymbol{v} = \text{Pool}(\text{Transformer-Enc}(\tilde{\boldsymbol{X}}^{(t)})), \tag{6}$$

$$\textit{where, } \tilde{\boldsymbol{X}}^{(t)} = [\boldsymbol{x}^{\texttt{[CLS]}}, \boldsymbol{X}^{(t)}, \boldsymbol{x}^{\texttt{[SEP]}}]. \tag{7}$$

Consequently, $\boldsymbol{v}$, a contextualized representation of *tail*, is viewed as *tail* embedding. In our experiment, we keep the two Transformer encoders (i.e., in Eq.(5) and (6)) parameter-tied for parameter efficiency [26]. And it is noteworthy that the Transformer encoder can be initialized by a pre-trained language model to further boost its capacity for representation learning, which alternates between BERT [12] and RoBERTa [17] in our experiments.

To sum up, also as answers to the questions above, (1) we trade off the contextualized knowledge with efficiency: keeping context across *head* and *relation*, while separating *tail* for reusable embeddings to avoid combinatorial explosion; (2) we partition each triple into two asymmetric parts as in TransE: a concatenation of *head*

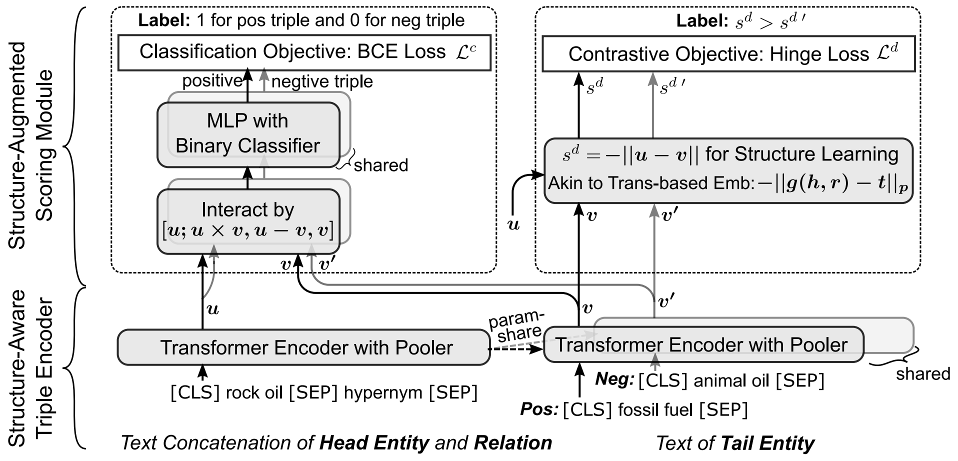
Structure-Augmented Scoring Module
Structure-Aware Triple Encoder
Label: 1 for pos triple and 0 for neg triple
Classification Objective: BCE Loss $\mathcal{L}^c$
positive
negtive triple
MLP with Binary Classifier
shared
Interact by $[u; u \times v, u - v, v]$
Label: $s^d > s^{d\,\prime}$
Contrastive Objective: Hinge Loss $\mathcal{L}^d$
$s^d = -||u - v||$ for Structure Learning
Akin to Trans-based Emb: $-||g(h,r) - t||_p$
param-share
Transformer Encoder with Pooler
Transformer Encoder with Pooler
shared
[CLS] rock oil [SEP] hypernym [SEP]
Neg: [CLS] animal oil [SEP]
Pos: [CLS] fossil fuel [SEP]
Text Concatenation of Head Entity and Relation
Text of Tail Entity

Figure 2: An overview of the proposed Structure-Augmented Text Representation (StAR) model for link prediction. This illustration is based on a corruption of tail entity, and in the same way for the corruption of a head entity or even relation. Note that a notation whose superscript includes " ′ " denotes it is derived from a negative example, otherwise from a positive one.

and *relation*, versus *tail*; and (3) we derive two contextualized embeddings for the two parts respectively, and aim to learn structured knowledge by exploring spatial characteristics between them.

3.2 Structure-Augmented Scoring Module

Given $\boldsymbol{u}$ and $\boldsymbol{v}$, we present two parallel scoring strategies as at the top of Figure 2 for both contextualized and structured knowledge.

3.2.1 Deterministic Representation Learning. Recently, some semantic matching graph embedding approaches for KGC use deterministic strategy [22, 23] to learn the representation of entities and relations. This strategy refers to using a binary classifier that determines if a triple is plausible or not. Such representation learning is especially significant to a text-based model, which has been adopted in KG-BERT for KGC and proven effective. But, this strategy cannot be applied to the pair of contextualized representations $\boldsymbol{u}$ and $\boldsymbol{v}$ produced by our Siamese-style encoder.

Fortunately, a common practice in NLP literature is to apply an interactive concatenation [4, 18, 26] to the pair of representations and then perform a neural binary classifier. Formally, we adopt the interactive concatenation written as

$$\boldsymbol{c} = [\boldsymbol{u}; \boldsymbol{u} \times \boldsymbol{v}; \boldsymbol{u} - \boldsymbol{v}; \boldsymbol{v}], \tag{8}$$

where $\boldsymbol{c}$ is used to represent the semantic relationship between the two parts of a triple. Then, similar to the top layer in KG-BERT, a two-way classifier is then applied to $\boldsymbol{c}$ and produces a two-dimensional categorical distribution corresponding to the negative and positive probabilities respectively, i.e.,

$$\boldsymbol{p} = P(z|\boldsymbol{c};\theta) \triangleq \text{softmax}(\text{MLP}(\boldsymbol{c};\theta)) \in \mathbb{R}^2, \tag{9}$$

where $\text{MLP}(\cdot)$ stands for a multi-layer perceptron, and θ is its learnable parameters. During the inference of link prediction, the 2*nd* dimension of $\boldsymbol{p}$, i.e., the positive probability,

$$s^c = p_2 \tag{10}$$

can serve as a score of the input triple to perform candidate ranking.

3.2.2 Spatial Structure Learning. In the meantime, it is possible to augment structured knowledge in the encoder by exploring the spatial characteristics between the two contextualized representations. Typically, translation-based graph embedding approaches conduct structure learning by measuring spatial distance. In particular, TransE [3] and RotatE [31] score a triple inversely proportional to the spatial distance between $g(\boldsymbol{h}, \boldsymbol{r})$ and $\boldsymbol{t}$, i.e., $-||g(\boldsymbol{h}, \boldsymbol{r}) - \boldsymbol{t}||$. And structured knowledge is acquired via maximizing the score margin between a positive triple and its corruptions (i.e., negative triples).

Here, as a triple is partitioned into two asymmetric parts by imitating the translation-based approaches, we can formulate $\boldsymbol{u} \leftarrow f(h, r)$ and $\boldsymbol{v} \leftarrow f(t)$, where $f(\cdot)$ denotes the textual encoder in §3.1. So, to acquire structured knowledge, we can score a triple by

$$s^d = -\text{Distance}(f(h,r), f(t)) \triangleq -||\boldsymbol{u} - \boldsymbol{v}||, \tag{11}$$

where $||\cdot||$ denotes $L2$-norm, and s^d is the plausible score based on the two contextualized representations, $\boldsymbol{u}$ and $\boldsymbol{v}$, of a triple.

3.3 Training and Inference

3.3.1 Training Objectives and Inference Details. Before presenting two training objectives, it is necessary to perform negative sampling and generate wrong triples. In detail, given a correct triple $tp = (h, r, t)$, we corrupt the triple and generate its corresponding wrong triple tp' by replacing either the head or tail entity with another entity randomly sampled from the entities $\mathcal{E}$ on $\mathcal{G}$ during training, which satisfies $tp' \in \{(h, r, t')|t' \in \mathcal{E} \wedge (h, r, t') \notin \mathcal{G}\}$ or $tp' \in \{(h', r, t)|h' \in \mathcal{E} \wedge (h', r, t) \notin \mathcal{G}\}$, where $\mathcal{E}$ denotes the ground set of all unique entities on $\mathcal{G}$. In the remainder, a variable with superscript " ′ " means that it is derived from a negative example.

Triple Classification Objective. Given the resulting confidence score s^c from Eq.(10) in deterministic representation learning (§3.2.1), we employ the following binary cross entropy loss to train the encoder w.r.t this objective, i.e.,

$$\mathcal{L}^c = -\frac{1}{|\mathcal{D}|}\sum_{tp\in\mathcal{D}}\frac{1}{1+|\mathcal{N}(tp)|}\left(\log s^c + \sum_{tp'\in\mathcal{N}(tp)}\log(1-s^{c\prime})\right), \quad (12)$$

where $\mathcal{D}$ denotes the training set containing only correct triples, $\mathcal{N}(tp)$ denotes a set of wrong triples generated from tp, s^c denotes positive probability of tp and $(1-s^{c\prime})$ denotes negative probability of the wrong triple tp'. We empirically find such representation learning using the deterministic strategy is critical to the success of textual encoding KGC, consistent with previous works [26, 45].

However, s^c might not contain sufficient information for ranking during inference since it is only the confidence for a single triple's correctness that does not take other triple candidates into account. This may cause inconsistency between the model's training and inference. To compromise, loss weights in Eq.(12) must be imbalanced between positive and negative examples, i.e., $|\mathcal{N}(tp)| \gg 1$, to distinguish the only positive triple among hundreds of thousands of corrupted ones during inference. Nonetheless, over-confident false positive predictions for a corruption (i.e., assigning a corrupted triple with $s^c \rightarrow 1.0$) still frequently appear to hurt the performance. These thus emphasize the importance of structure learning.

Triple Contrastive Objective. Given the distance-based score s^d from Eq.(11) in spatial structure learning (§3.2.1), we also train the encoder by using a contrastive objective. The contrastive objective considers a pairwise ranking between a correct triple and a wrong triple, where the latter is corrupted from the former by negative sampling. Formally, let s^d denote the score derived from a positive triple tp and $s^{d'}$ denote the score derived from a wrong triple tp', we define the loss by using a margin-based hinge loss function, i.e.,

$$\mathcal{L}^d = \frac{1}{|\mathcal{D}|}\sum_{tp\in\mathcal{D}}\frac{1}{|\mathcal{N}(tp)|}\sum_{tp'\in\mathcal{N}(tp)}\max(0, \lambda - s^d + s^{d'}). \quad (13)$$

In experiments, we qualitatively reveal that structure learning is significant to reducing false positive and disambiguating entities, and pushes our model to produce more reliable ranking scores.

Training and Inference Strategies. The loss $\mathcal{L}$ to train the StAR is a sum of the two losses defined in Eq.(12) and Eq.(13), i.e.,

$$\mathcal{L} = \mathcal{L}^c + \gamma\mathcal{L}^d, \quad (14)$$

where γ is the weight. After optimizing StAR w.r.t $\mathcal{L}$, s^c, s^d or their integration can be used as ranking basis during inference. We will present a thorough empirical study of the possible options of ranking score based on s^c and s^d in §4.4.

3.3.2 Model Efficiency. In the following, we analyze why our proposed model is significantly faster than its baseline, KG-BERT.

Training Efficiency. As overheads are dominated by the computations happening inside the Transformer encoder, we focus on analyzing the complexity of computing the contextualized embeddings by the encoder. In practice, the sequence lengths of the two asymmetric parts of a triple are similar because the length of an entity's text is usually much longer than a relation's text, especially when the entity description is included [38, 45]. Hence, Siamese-style StAR is 2× faster than KG-BERT in training as the complexity of Transformer encoder grows quadratically with sequence length.

Table 1: Inference efficiency comparison. L is the length of triple text. $|\mathcal{E}|$ and $|\mathcal{R}|$ are the numbers of all unique entities and relations in the graph respectively. Usually, $|\mathcal{E}|$ exceeds hundreds of thousands and is much greater than $|\mathcal{R}|$.

Inference on	Method	Complexity	Speed up
One Triple	KG-BERT	$O(L^2\|\mathcal{E}\|)$	$\sim 4\times$
	StAR	$O((L/2)^2(1+\|\mathcal{E}\|))$	
Entire Graph	KG-BERT	$O(L^2\|\mathcal{E}\|^2\|\mathcal{R}\|)$	$\sim 4\|\mathcal{E}\|\times$
	StAR	$O((L/2)^2\|\mathcal{E}\|(1+\|\mathcal{R}\|))$	

Inference Efficiency. Similarly, we also focus on analyzing the overheads used in the encoder during inference. As shown in Table 1, we list the complexities of both KG-BERT baseline and proposed StAR, and analyze the acceleration in two cases. In practice, on the test set of a benchmark, our approach, without combinatorial explosion, is faster than KG-BERT by two orders of magnitude.

3.4 Self-Adaptive Ensemble Scheme

StAR improves previous textual encoding approaches by introducing structure learning. It reduces those overconfident but false positive predictions and mitigates the entity ambiguity problem. However, compared to graph embedding operating at entity or relation level, our StAR based on the text inherently suffers from entity ambiguity. Fortunately, combining textual encoding with graph embedding paradigms can provide a remedy: Despite entity ambiguity existing, a textual encoding approach achieves a high recall in top-k with slightly large k (e.g., $k > 5$), whereas a graph embedding approach can then precisely allocate the correct one from the k candidates due to robustness to ambiguity. Note, $k \ll |\mathcal{E}|$. To the best of our knowledge, this is the first work to ensemble the two paradigms for mutual benefits. Surprisingly, simple averaging of the scores from the two paradigms significantly improves the performance. This motivates us to take a step further and develop a self-adaptive ensemble scheme.

Given an incomplete triple (i.e., $(h, r, ?)$ or $(?, r, t)$), we aim to learn a weight $\alpha \in [0, 1]$ to generate the final triple-specific score:

$$s^{(sa)} = \alpha \times s^{(tc)} + (1-\alpha) \times s^{(ge)}, \quad (15)$$

where $s^{(tc)}$ is derived from StAR and $s^{(ge)}$ is derived from RotatE [31]. Since $s^{(tc)} = s^c \in [0, 1]$ from Eq.(10) is normalized, we rescale all candidates' scores of RotatE into $[0, 1]$ to obtain $s^{(ge)}$. Specifically, for an incomplete triple, we first take the top-k candidates ranked by StAR and fetch their scores from the two models, which are denoted as $\boldsymbol{s}^{(tc)} \in [0, 1]^k$ and $\boldsymbol{s}^{(ge)} \in [0, 1]^k$ respectively. Then, we set an *unseen indicator* to force $\alpha = 1$ if an unseen entity/relation occurs in the incomplete triple. Next, to learn a triple-specific α, we build an MLP based upon two kinds of features: *ambiguity degree* $\boldsymbol{x}^{(ad)}$ and *score consistency* $\boldsymbol{x}^{(sc)}$. Particularly, the ambiguity degree $\boldsymbol{x}^{(ad)} \triangleq [\mathrm{Std}(\boldsymbol{V}); \mathrm{Mean}(\boldsymbol{M})]$ where "$\mathrm{Std}(\boldsymbol{V} \in \mathbb{R}^{d\times k}) \in \mathbb{R}^d$" is the standard deviation of the top-k entities' representations, and "$\mathrm{Mean}(\boldsymbol{M} \in \mathbb{R}^{k\times 100}) \in \mathbb{R}^k$" averages the largest 100 cosine similarities between each candidate and all entities in $\mathcal{E}$. Note each entity

Table 2: Summary statistics of benchmark datasets.

Dataset	# Ent	# Rel	# Train	# Dev	# Test
WN18RR	40,943	11	86,835	3,034	3,134
FB15k-237	14,541	237	272,115	17,535	20,466
UMLS	135	46	5,216	652	661
NELL-One	68,545	822	189,635	1,004	2,158

is denoted by its contextualized representation from Eq.(6). And, the score consistency $\boldsymbol{x}^{(sc)} \triangleq [|\boldsymbol{s}^{(tc)} - \boldsymbol{s}^{(ge)}|, \boldsymbol{s}^{(tc)} + \boldsymbol{s}^{(ge)}, \boldsymbol{s}^{(tc)}, \boldsymbol{s}^{(ge)}]$. Lastly, we pass the features into an MLP with activation σ, i.e.,

$$\alpha = \sigma(\mathrm{MLP}([\boldsymbol{x}^{(ad)}; \boldsymbol{x}^{(sc)}]; \theta^{(\alpha)})) \in [0, 1]. \tag{16}$$

In training, we fix the parameters of both our model and RotatE, and only optimize $\theta^{(\alpha)}$ by a margin-based hinge loss. In inference, we use the resulting $s^{(sa)}$ to re-rank the top-k candidates while keep the remaining unchanged. In experiments, we evaluated two variants of StAR: (1) **StAR (Ensemble)**: $k \leftarrow \infty$ and $\alpha \leftarrow 0.5$, equivalent to score average, as our ensemble baseline. (2) **StAR (Self-Adp)**: $k \leftarrow 1000$ and α is learnable.

3.5 Compared to Prior Text-Based Approach

Sharing a similar motivation, some previous approaches also use textual information to represent entities and relations. However, they are distinct from textual encoding approaches like KG-BERT or StAR and can be coarsely categorized into two groups:

Stand-alone Embedding. These approaches [28, 29] directly replace an entity/relation embedding in graph with its text representation. The representation is derived from applying a shallow encoder (e.g., CBoW and CNN) to text, regardless of contextual information across entities and relations. But, deep contextualized features are proven effective and critical to text representation for various NLP tasks [12, 25]. For KBC, the features are significant for entity disambiguation. Therefore, despite slightly improving generalization, they still deliver an inferior performance. In contrast, our model achieves a better trade-off between deep contextualized features and efficiency by the carefully designed triple encoder.

Joint Embedding. More similar to our work, some other approaches [13, 33, 37–39, 42, 43] also bring text representation learning into graph embedding paradigm. Standing opposite our model, they start with graph embeddings and aim at enriching the embeddings with text representations. Typically, they either use text embeddings to represent entities/relations and align heterogeneous representations into the same space [37, 39, 43], or employ large-scale raw corpora containing co-occurrence of entities to enrich the graph embeddings [42]. However, due to graph embeddings involved, they inevitably inherit the generalization problem and incompleteness issue. And same as the above, the representation learning here is also based on shallow networks without deep contextualized knowledge. In contrast, our model, based solely on text's contextualized representations and coupled with structure learning, is able to achieve mutual benefits of the two paradigms for KBC.

4 EXPERIMENT

In this section[2], we evaluate StAR on several popular benchmarks (§4.1), and verify the model's efficiency (§4.2) and generalization (§4.3). Then, we conduct an extensive ablation study in §4.4 to test various model selections and verify the significance of each proposed module. Lastly, in §4.5 we comprehensively analyze the difference between graph embedding approach and textual encoding approach, and assess the self-adaptive ensemble scheme.

Benchmark Datasets. We assessed the proposed approach on three popular and one zero-shot link prediction benchmarks, whose statistics are listed in Table 2. First, *WN18RR* [11] is a link prediction dataset from WordNet [20]. It consists of English phrases and their semantic relations. Second, *FB15k-237* [33] is a subset of Freebase [2], consisting of real-world named entities and their relations. Note, WN18RR and FB15k-237 are updated from WN18 and FB15k [3] respectively by removing inverse relations and data leakage, which are the most popular benchmarks.[3] And third, *UMLS* [11] is a small KG containing medical semantic entities and their relations. Finally, to verify model's generalization, *NELL-One* [41] is a few-shot link prediction dataset derived from NELL [6], where the relations in dev/test set never appear in train set. We adopted "In-Train" scheme by Chen et al. [8] and used zero-shot setting. And, in line with prior approaches [38, 45], we employed entity descriptions as their text for WN18RR and FB15k-237 from synonym definitions and Wikipedia paragraph [39] respectively. As for the text of relations and other datasets' entities, we directly used their text contents. Please refer to Appendix A for our training setups.

Evaluation Metrics. In the inference phase, given a test triple of a KG as the correct candidate, all other entities in the KG act as wrong candidates to corrupt its either head or tail entity. The trained model aims at ranking correct triple over corrupted ones with "*filtered*" setting [3]. For evaluation metrics, there are two aspects: (1) Mean rank (MR) and mean reciprocal rank (MRR) reflect the absolute ranking; and (2) Hits@N stands for the ratio of test examples whose correct candidate is ranked in top-N. And, although there are two ranking scores from Eq.(10) and Eq.(11), only s^c is used for ranking, and other options will be discussed in §4.4.

Evaluation Protocol. We must emphasize that, as stated by Sun et al. [32], previous methods (e.g., ConvKB, KBAT and CapsE) use an inappropriate evaluation protocol and thus mistakenly report very high results. The mistake frequently appears in a method whose score is normalized, says $[0, 1]$, due to float precision problem. So, we strictly follow the "RANDOM" protocol proposed by Sun et al. [32] to evaluate our models, and avoid comparisons with vulnerable methods that have not been re-evaluated.

4.1 Evaluations on Link Prediction

The link prediction results of competitive approaches and ours on the three benchmarks are shown in Table 3. It is observed our

[2]The source code is available at https://github.com/wangbo9719/StAR_KGC.

[3]WN18 and FB15k suffer from informative value [11, 33], which causes $> 80\%$ of the test triples (e^1, r^1, e^2) can be found in the training set with another relation: (e^1, r^2, e^2) or (e^2, r^2, e^1). Dettmers et al. [11] used a rule-based model that learned the inverse relation and achieved state-of-the-art results on the dataset. Thereby it is suggested they should not be used for link prediction evaluation anymore.

Table 3: Link prediction results on WN18RR, FB15k-237 and UMLS. †Resulting numbers are reported by Nathani et al. [21], ◇Resulting numbers are re-evaluated by [32], and others are taken from the original papers; UMLS results are reported by Yao et al. [45], except ConvE from our re-implementation. The bold numbers denote the best results in each genre while the underlined ones are state-of-the-art performance.

	WN18RR					**FB15k-237**					**ULMS**	
	Hits@1	Hits@3	Hits@10	MR	MRR	Hits@1	Hits@3	Hits@10	MR	MRR	Hits@10	MR
Graph embedding approach												
TransE [3]†	.043	.441	.532	2300	.243	.198	.376	.441	323	.279	.989	1.84
DistMult [44]†	.412	.470	.504	7000	.444	.199	.301	.446	512	.281	.846	5.52
ComplEx [34]†	.409	.469	.530	7882	.449	.194	.297	.450	546	.278	.967	2.59
KBGAN [5]	-	-	.469	-	.215	-	-	.458	-	.277	-	-
R-GCN [27]†	.080	.137	.207	6700	.123	.100	.181	.300	600	.164	-	-
ConvE [11]†	.419	.470	.531	4464	.456	.225	.341	.497	245	.312	**.990**	**1.51**
ConvKB [22]◇	-	-	.524	3433	.249	-	-	.421	309	.243	-	-
KBAT [21]◇	-	-	.554	1921	.412	-	-	.331	270	.157	-	-
CapsE [23]◇	-	-	.559	**718**	.415	-	-	.356	403	.150	-	-
QuatE [47]	.436	**.500**	.564	3472	.481	.221	.342	.495	**176**	.311	-	-
RotatE [31]	.428	.492	.571	3340	.476	.241	.375	.533	177	.338	-	-
TuckER [1]	**.443**	.482	.526	-	.470	**.266**	**.394**	**.544**	-	**.358**	-	-
AttH[7]	**.443**	.499	**.573**	-	**.486**	.252	.384	.540	-	.348		
Textual encoding approach												
KG-BERT [45]	.041	.302	.524	97	.216	-	-	.420	153	-	.990	**1.47**
StAR	**.243**	**.491**	**.709**	**51**	**.401**	**.205**	**.322**	**.482**	**117**	**.296**	**.991**	1.49
Our ensemble model												
StAR (Ensemble)	.449	.551	.675	540	.524	.264	.399	.559	**109**	.362	-	-
StAR (Self-Adp)	**.459**	**.594**	**.732**	**46**	**.551**	**.266**	**.404**	**.562**	117	**.365**	-	-

Table 4: Comparisons with KG-BERT on WN18RR. "T/Ep" stands for time per training epoch and "Infer" denotes inference time on test set. The time was collected on RTX6000 with mixed precision.

	Hits@1	@3	@10	MR	MRR	T/Ep	Infer
KG-BERT$_{\text{BERT-base}}$	.041	.302	.524	97	.216	40m	32h
StAR$_{\text{BERT-base}}$	.222	.436	.647	99	.364	20m	0.9h
KG-BERT$_{\text{RoBERTa-base}}$	.130	.320	.636	84	.278	40m	32h
StAR$_{\text{RoBERTa-base}}$	.202	.410	.621	71	.343	20m	0.9h
KG-BERT$_{\text{RoBERTa-large}}$	.119	.387	.698	95	.297	79m	92h
StAR$_{\text{RoBERTa-large}}$	.243	.491	.709	51	.401	55m	1.0h

Table 5: Link prediction results on NELL-One. StAR with zero-shot setting is competitive with few-shot GMatching [41] and MetaR [8].

Methods	*N*-Shot	Hits@1	Hits@5	Hits@10	MRR
GMatching$_{\text{ComplEx}}$	Five-Shot	.14	.26	.31	.20
MetaR		.17	.35	.44	.26
GMatching$_{\text{TransE}}$	One-Shot	.12	.21	.26	.17
GMatching$_{\text{DistMult}}$		.11	.22	.30	.17
GMatching$_{\text{ComplEx}}$		.12	.26	.31	.19
MetaR		.17	.34	.40	.25
StAR$_{\text{BERT-base}}$	Zero-Shot	**.17**	**.35**	**.45**	**.26**

proposed StAR is able to achieve state-of-the-art or competitive performance on all these datasets. The improvement is especially significant in terms of MR due to the great generalization performance of textual encoding approach, which will be further analyzed in the section below. And on WN18RR, StAR surpasses all other methods by a large margin in terms of Hits@10. Although it only achieves inferior performance on Hits@1 compared to graph embedding approaches, it still remarkably outperforms KG-BERT from the same genre by introducing structured knowledge.

Further, coupled with the proposed self-adaptive scheme, the proposed model delivers new state-of-the-art performance on all metrics. Specifically, our self-adaptive model "StAR (Self-Adp)" significantly surpasses its ensemble baseline "StAR (Ensemble)" on most metrics. And, even if Hits@1 is the main weakness for a textual encoding paradigm, our self-adaptive model is still superior than the best semantic matching graph embedding approach TuckER.

4.2 Comparison with KG-BERT Baseline

Since our approach is an update from the non-Siamese-style baseline, says KG-BERT, we compared StAR with KG-BERT on WN18RR in detail, including different initializations. As shown in Table 4, our proposed StAR consistently achieves superior performance over most metrics. As for empirical efficiency, it is observed our model is faster than KG-BERT despite training or inference, which is roughly consistent with the theoretical analysis in §3.3.2.

4.3 Generalization to Unseen Graph Elements

Textual encoding approaches are more generalizable to unseen entities/relations than graph embedding ones. This can be more significant when the set of entities or relations is not closed, i.e., unseen graph elements (i.e., entities/relations) appear during inference. For example, 209 out of 3134 and 29 out of 20466 test triples involve

Table 6: Probing tasks based on WN18RR for analyzing models' generalization performance.

		Hits@1	Hits@3	Hits@10	MR	MRR
Original Task	**StAR**	.243	.491	.709	51	.401
	RotatE	.428	.492	.571	3340	.476
	TransE	.042	.441	.532	2300	.243
First Probing Task	**StAR**	.240	.452	.673	71	.384
	RotatE	.005	.007	.012	17955	.007
	TransE	.000	.007	.016	20721	.007
Second Probing Task	**StAR**	.301	.497	.676	99	.427
	TransE	.005	.121	.210	13102	.078
Third Probing Task	**StAR**	.244	.493	.712	49	.402
	RotatE	.455	.523	.612	1657	.507

unseen entities on WN18RR and FB15k-237 respectively. This inevitably hurts the performance of graph embedding approaches, especially for the unnormalized metric MR.

First, we employed a few-shot dataset, NELL-One, to perform a zero-shot evaluation where relations in test never appear in training set. As shown in Table 5, StAR with zero-shot setting is competitive with graph embedding approaches with one/five-shot setting.

Then, to verify the generalization to unseen entities, we built two probing settings on WN18RR. The *first probing task* keeps training set unchanged and makes the test set only consist of the triples with unseen entities. And in the *second probing task*, we randomly removed 1900 entities from training set to support inductive entity representations [14] during test for TransE. The setting is detailed in Appendix B. As shown in Table 6, StAR is competitive across the settings but advanced graph embedding approaches (e.g., RotatE) show a substantial drop in the first task. Even if we used translation formula to inductively complete unseen entities' embeddings in the second probing task, the degeneration of TransE is significant. These verify StAR's promising generalization to unseen elements.

Lastly, to verify the proposed model is still competitive even if applied to close sets of entities/relations, we built the *third probing task* as in Table 6. We only kept the WN18RR test triples with entities/relations visited during training while removed the others.

4.4 Ablation Study

To explore each module's contribution, we conducted an extensive ablation study about StAR and the self-adaptive ensemble scheme as shown in Table 7. For single StAR, (1) *Ablating Objective*: First, each of the components in Eq.(14) were removed to estimate the significance of structure and representation learning. (2) *Contexts' concatenation*: Then, how to concatenate and encode the text from a triple is also non-trivial for learning structured knowledge. Two other options only achieved sub-optimal results. (3) *Distance measurement*: Two other methods, i.e., Bilinear and Cosine, similarity were also applied to Eq.(11) to measure the distance for structure learning. (4) *Ranking Basis*: Since two scores can be derived from the two objectives respectively, it is straightforward to integrate them in either additive or multiplicative way. As these ranking bases achieve similar performance, we further calculated the Pearson correlation between s^c and s^d and found the coefficient is 0.939 (p-value=7×10^{-4}), which means the two scores are linearly related.

Table 7: Ablation study on WN18RR. Note that *Full model denotes using two objectives for training, " $[h, r]$ **vs.** $[t]$**" as concatenation scheme,** $L2$ **norm as measurement, and** s^c **as ranking basis during inference. And** $\text{Rescale}(\cdot)$ **denotes scaling all scores to** $[0, 1]$**.**

Perspective	Detail	Hits@10	MR	MRR
***Single Model**: Module Ablation and Selection in "StAR"*				
*Full model**	$\text{StAR}_{\text{RoBERTa-large}}$	.709	51	.401
Objective	· w/o contrastive obj	.685	68	.399
	· w/o classification obj	.653	67	.337
Concatenation, e.g., Eq.(4, 7)	· $[h, r]$ vs. $[r, t]$	.520	106	.204
	· $[h]$ vs. $[r, t]$	.668	51	.402
Distance in Eq.(11)	· Bilinear	.605	79	.354
	· Cosine Similarity	.691	76	.439
Ranking Basis	· s^d	.701	62	.406
	· $\text{Rescale}(s^d) + s^c$	.706	48	.408
	· $\text{Rescale}(s^d) \times s^c$	.704	51	.408
***Ensemble Model**: Feature Ablation in "StAR (Self-Adp)"*				
Full model	StAR (Self-Adp)	.732	46	.551
Feature	· w/o hard indicator	.712	50	.540
	· w/o ambiguity degree	.734	45	.537
	· w/o score consistency	.720	45	.540
	· w/o self-Adp α in Eq.(16)	.675	540	.524

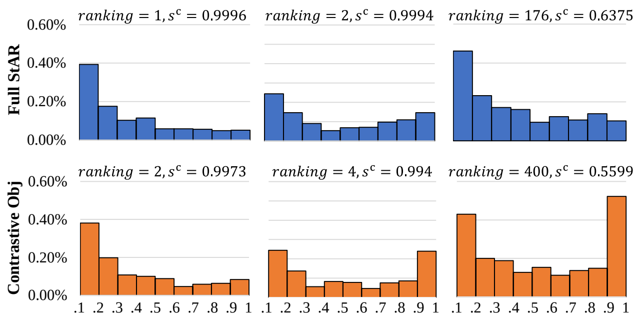

Figure 3: Three random comparative cases of frequency histogram for $s^{c\,\prime}$ **assigned to a triple's all tail corruptions.** x**-axis denotes** $s^{c\,\prime}$ **and** y**-axis denotes frequency over the number of all corruptions. The text above each histogram shows the ranking and** s^c **for the corresponding un-corrupted (i.e., oracle) triple. Note, interval of [0.0, 0.1] is removed since most negative triples'** $s^{c\,\prime}$ **will fall into it.**

For "StAR (Self-Adp)" (in § 3.4), we ablated its features: (1) unseen indicator, (2) ambiguity degree, and (3) score consistency.

4.5 Further Analyses

Here, we analyze the effect of structure learning on our textual encoding approach, and compare the proposed model with a graph embedding approach. And we also attempt to qualitatively measure the effectiveness of the proposed self-adaptive ensemble scheme.

What is the effect of introducing structure learning into a textual encoding approach. As shown in Figure 3, we compared the frequency histograms of ranking score $s^{c\,\prime}$ derived from classification objective for negative triples from either the "full StAR" or "StAR w/o contrastive objective". It is observed, the textual encoding model augmented with structure learning reduces the number

Table 8: Top-5 ranking results of candidate entities for different approaches. The first column includes incomplete triples for inference, and their labels. And the others include the ranking position and Top-5 ranked candidates where an underline denotes it is the gold entity.

Incomplete Triple	Positive entity ranking position & Top-5 ranked candidate entities		
	StAR (Self-Adp) [Ensemble]	**StAR** [Textual Encoding]	**RotatE** [Graph Embedding]
(*world war ii, has part, ?*) ← *tarawa-makin*	5, (*world war ii, jutland, meuse river, soissons, tarawa-makin*)	12, (*world war ii, world war i, world war, seven years' war, meuse river*)	10, (*jutland, world war ii, somme river, verdun, soissons*)
(*clarify, hypernym, ?*) ← *modify*	2, (*clarify, modify, change integrity, convert*^{a}*, convert*^{b})	3, (*clarify, straighten out, modify, alter, transubstantiate*)	66, (*cook*^{a}*, season, ready, cook*^{b}*, preserve*)
(*mechanical system, hypernym, ?*) ← *system*^{a}	2, (*mechanical system, system*^{a}*, mechanism, system*^{b}*, machine*)	3, (*system*^{b}*, mechanical system, system*^{a}*,mechanism, system*^{c})	24, (*mechanical system, production line, suspension system, . . .*)

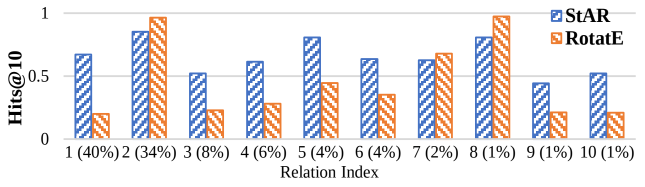

Figure 4: A comparison between StAR and RotatE regarding different relations on WN18RR test set. Relations corresponding to the indices are 1) *hypernym*, 2) *derivationally related form*, 3) *member meronym*, 4) *has part*, 5) *instance hypernym*, 6) *synset domain topic of*, 7) *also see*, 8) *verb group*, 9) *member of domain region*, 10) *member of domain usage*. The number in a parenthesis denotes its proportion of test triples with the corresponding relation. Note relation "*similar to*" is ignored since its proportion is less than 0.1%.

Table 9: An example of polysemy in WordNet: three meanings of "*sensitive*" are viewed as three separate nodes.

- sensitive^a: able to feel or perceive.
- sensitive^b: responsive to physical stimuli.
- sensitive^c: being susceptible to the attitudes, feelings, or circumstances of others.

Table 10: Applying self-loop filter to WN18RR.

	Hits@1	@3	@10	MR	MRR
StAR	.243	.491	.709	51	.401
+ Self-loop Filter	.328	.533	.719	50	.460

of false positive predictions and produces more accurate ranking scores. This also verifies the structured knowledge can alleviate the over-confidence problem (§3.3.1).

How does StAR bring improvements. As shown in Figure 4, a detailed comparison regarding different relations is conducted between StAR and RotatE. It is observed StAR achieves more consistent results than RotatE. However, StAR performs worse on several certain relations, e.g., the 8*th* relation in Figure 4, *verb group*. After checking the test triples falling into *verb group*, we found "polysemy" occurs in half of the triples, e.g., (*strike*^{a}*, verb group, strike*^{b}), which hinders StAR from correctly ranking. These imply that even coupled with structured knowledge, a textual encoding approach is still vulnerable to entity ambiguity or word polysemy, and emphasize the importance of our self-adaptive ensemble scheme.

Why does StAR achieve better Hits@10 but worse Hits@1 than RotatE. As shown in Table 3, it is observed that the textual encoding approach (e.g., KG-BERT, StAR) can outperform graph embedding approach (e.g., TransE, RotatE) by a large margin on Hits@10 but underperform on Hits@1. To dig this out, we conducted a case study based on the inference on WN18RR. In particular, given an oracle test triple, (*sensitive*^{a}*, derivationally related form, sense*), after corrupting its tail and ranked by our StAR, the top-12 tail candidates are (*sensitive*^{a}*, sensitivity, sensibility, sensing, sense impression, sentiency, sensitive*^{b}*, sense, feel, sensory, sensitive*^{c}*, perceptive*), where gold tail is only ranked 8*th*. It is observed there are many semantically-similar tail entities that can fit the oracle triple, which seem to be false negative labels for a context encoder. But this is not a matter for graph embedding approaches since they only consider graph's structure despite text. It is worth mentioning "polysemy" or "ambiguity" issue usually appears in WN18RR (an example in Table 9). The issue is more severe in FB15K-237, which partially explains why StAR only achieves competitive results. Fortunately, this issue can be significantly alleviated by the self-adaptive ensemble scheme. And, it is interesting the oracle head is ranked 1*st* for tail in this case but self-loop will never appear in WN18RR's test set. Hence, as shown in Table 10, after filtering self-loop candidates during inference, the performance is improved.

How does the self-adaptive ensemble scheme bring improvements. As shown in Table 3, "StAR (Self-Adp)" improves the performance than "StAR" or RotatE used alone. Intuitively, the improvement is brought from the mutual benefits of representation and structure learning. For further confirmation, we randomly listed some triples in WN18RR test, where the triples experience a certain improvement when applying self-adaptive ensemble scheme. As shown in Table 8, as demonstrated in the 1*st* and 3*rd* examples, it is observed that graph structure helps distinguish semantically-similar candidate entities and alleviate the "polysemy" problem. In addition, since the rich contextualized information empowers model with a high top-k recall, the self-adaptive ensemble model still achieves a satisfactory ranking result as shown in the 2*nd* example, even if the graph embedding model underperforms. As a result, due to the complementary benefits, the self-adaptive ensemble scheme offers significant improvements over previous approaches.

5 RELATED WORK

Structure Learning for Link Prediction. Previous graph embedding approaches explore structured knowledge through spatial measurement or latent matching in low-dimension vector space. Specifically, on the one hand, translation-based graph embedding approach [3, 31] applies a translation function to *head* and *relation*, and compares the resulting with *tail* via spatial measurement. The most well-known one, TransE [3], implements the function and the measurement with real vector addition and $L2$ norm respectively – scoring a triple by $-||(\boldsymbol{h}+\boldsymbol{r})-\boldsymbol{t}||$. However, the graph embeddings defined in real vector space hardly deal with the *symmetry* relation pattern, and thereby underperform. To remedy this, RotatE [31] defines the graph embeddings in complex vector space, and implements the translation function with the production of two complex numbers in each dimension. On the other hand, semantic matching graph embedding approach [1, 44, 47] uses a matching function $f(\boldsymbol{h},\boldsymbol{r},\boldsymbol{t})$ operating on whole triple to directly derive its plausibility score. For example, DistMult [44] applies a bilinear function to each triple's components and uses the latent similarity in vector space as the plausibility score. In spite of their success, the rich text contextualized knowledge is entirely ignored, leading to less generalization.

Text Representation Learning. In an NLP literature, text representation learning is fundamental to any NLP task, which aims to produce expressively powerful text embedding with contextualized knowledge [12, 25]. When applied to KGC, some approaches [28, 29] directly replace the graph embeddings with their text embedding. For example, Socher et al. [29] simply use continuous CBoW as the representation of triple's component, and then proposed a neural tensor network for relation classification. ConMask [28] learns relationship-dependent entity embeddings of the entity's name and parts of description based on fully CNN. These approaches are not competitive since the deep contextualized representation of a triple is not leveraged. In contrast, KG-BERT [45], as a textual encoding approach, applies pre-trained encoder to a concatenation of triples' text for deep contextualized representations. Such a simple method is very effective, but unfortunately suffers from high overheads.

Jointly Learning Methods. Unlike the approaches above learning either knowledge solely, several works explore jointly learning both text and structured knowledge. Please refer to the end of §3.5 for more detail. For example, taking into account the sharing of substructure in the textual relations in a large-scale corpus, Toutanova et al. [33] applied a CNN to the lexicalized dependency paths of the textual relation, for augmented relation representations. Xie et al. [39] propose a representation learning method for KGs via embedding entity descriptions, and explored CNN encoder in addition to CBoW. They used the objective across this representation and graph embeddings that a vector integration of head and relation was close to the vector of tail to learn the model, as in translation-based graph embedding approaches [3]. In contrast, our work only operates on homogeneous textual data and employs the contexts for entities/relations themselves (i.e., only their own text contents or description), rather than acquiring textual knowledge (e.g., textual relations by Toutanova et al. [33]) from large-scale corpora to enrich traditional graph embeddings via joint embedding.

6 CONCLUSION

In this work, we propose a structure-augmented text representation (StAR) model to tackle link prediction task for knowledge graph completion. Inspired by translation-based graph embedding designed for structure learning, we first apply a Siamese-style textual encoder to a triple for two contextualized representations. Then, based on the two representations, we present a scoring module where two parallel scoring strategies are used to learn both contextualized and structured knowledge. Moreover, we propose a self-adaptive ensemble scheme with graph embedding approach, to further boost the performance. The empirical evaluations and thorough analyses on several mainstream benchmarks show our approach achieves state-of-the-art performance with high efficiency.

ACKNOWLEDGEMENT

The authors would like to thank the anonymous referees for their valuable comments. This work is supported by the National Natural Science Foundation of China (No.61976102, No.U19A2065, No.61872161) and the Foundation of Development and Reform of Jilin Province (2019C053-8).

REFERENCES

[1] Ivana Balazevic, Carl Allen, and Timothy M. Hospedales. 2019. TuckER: Tensor Factorization for Knowledge Graph Completion. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan (Eds.). Association for Computational Linguistics, 5184–5193. https://doi.org/10.18653/v1/D19-1522

[2] Kurt D. Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2008, Vancouver, BC, Canada, June 10-12, 2008*, Jason Tsong-Li Wang (Ed.). ACM, 1247–1250. https://doi.org/10.1145/1376616.1376746

[3] Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating Embeddings for Modeling Multi-relational Data. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, Christopher J. C. Burges, Léon Bottou, Zoubin Ghahramani, and Kilian Q. Weinberger (Eds.). 2787–2795. http://papers.nips.cc/paper/5071-translating-embeddings-for-modeling-multi-relational-data

[4] Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015.* The Association for Computational Linguistics, 632–642. https://doi.org/10.18653/v1/d15-1075

[5] Liwei Cai and William Yang Wang. 2018. KBGAN: Adversarial Learning for Knowledge Graph Embeddings. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, Marilyn A. Walker, Heng Ji, and Amanda Stent (Eds.). Association for Computational Linguistics, 1470–1480. https://doi.org/10.18653/v1/n18-1133

[6] Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R. Hruschka Jr., and Tom M. Mitchell. 2010. Toward an Architecture for Never-Ending Language Learning. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2010, Atlanta, Georgia, USA, July 11-15, 2010*, Maria Fox and David Poole (Eds.). AAAI Press. http://www.aaai.org/ocs/index.php/AAAI/AAAI10/paper/view/1879

[7] Ines Chami, Adva Wolf, Da-Cheng Juan, Frederic Sala, Sujith Ravi, and Christopher Ré. 2020. Low-Dimensional Hyperbolic Knowledge Graph Embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel R. Tetreault (Eds.). Association for Computational Linguistics, 6901–6914. https://www.aclweb.org/anthology/2020.acl-main.617/

[8] Mingyang Chen, Wen Zhang, Wei Zhang, Qiang Chen, and Huajun Chen. 2019. Meta Relational Learning for Few-Shot Link Prediction in Knowledge Graphs.

In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan (Eds.). Association for Computational Linguistics, 4216–4225. https://doi.org/10.18653/v1/D19-1431

[9] Sumit Chopra, Raia Hadsell, and Yann LeCun. 2005. Learning a Similarity Metric Discriminatively, with Application to Face Verification. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2005), 20-26 June 2005, San Diego, CA, USA*. IEEE Computer Society, 539–546. https://doi.org/10.1109/CVPR.2005.202

[10] Silviu Cucerzan. 2007. Large-Scale Named Entity Disambiguation Based on Wikipedia Data. In *EMNLP-CoNLL 2007, Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, June 28-30, 2007, Prague, Czech Republic*, Jason Eisner (Ed.). ACL, 708–716. https://www.aclweb.org/anthology/D07-1074/

[11] Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2D Knowledge Graph Embeddings. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, Sheila A. McIlraith and Kilian Q. Weinberger (Eds.). AAAI Press, 1811–1818. https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/17366

[12] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, Jill Burstein, Christy Doran, and Thamar Solorio (Eds.). Association for Computational Linguistics, 4171–4186. https://doi.org/10.18653/v1/n19-1423

[13] Lea Dieudonat, Kelvin Han, Phyllicia Leavitt, and Esteban Marquer. 2020. Exploring the Combination of Contextual Word Embeddings and Knowledge Graph Embeddings. *arXiv preprint arXiv:2004.08371* (2020).

[14] William L. Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (Eds.). 1024–1034. http://papers.nips.cc/paper/6703-inductive-representation-learning-on-large-graphs

[15] He He, Anusha Balakrishnan, Mihail Eric, and Percy Liang. 2017. Learning Symmetric Collaborative Dialogue Agents with Dynamic Knowledge Graph Embeddings. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, Regina Barzilay and Min-Yen Kan (Eds.). Association for Computational Linguistics, 1766–1776. https://doi.org/10.18653/v1/P17-1162

[16] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net. https://openreview.net/forum?id=SJU4ayYgl

[17] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *CoRR* abs/1907.11692 (2019). arXiv:1907.11692 http://arxiv.org/abs/1907.11692

[18] Yang Liu, Chengjie Sun, Lei Lin, and Xiaolong Wang. 2016. Learning Natural Language Inference using Bidirectional LSTM model and Inner-Attention. *CoRR* abs/1605.09090 (2016). arXiv:1605.09090 http://arxiv.org/abs/1605.09090

[19] Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*. 3111–3119. http://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality

[20] George A Miller. 1998. *WordNet: An electronic lexical database*. MIT press.

[21] Deepak Nathani, Jatin Chauhan, Charu Sharma, and Manohar Kaul. 2019. Learning Attention-based Embeddings for Relation Prediction in Knowledge Graphs. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, Anna Korhonen, David R. Traum, and Lluís Màrquez (Eds.). Association for Computational Linguistics, 4710–4723. https://doi.org/10.18653/v1/p19-1466

[22] Dai Quoc Nguyen, Tu Dinh Nguyen, Dat Quoc Nguyen, and Dinh Q. Phung. 2018. A Novel Embedding Model for Knowledge Base Completion Based on Convolutional Neural Network. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 2 (Short Papers)*, Marilyn A. Walker, Heng Ji, and Amanda Stent (Eds.). Association for Computational Linguistics, 327–333. https://doi.org/10.18653/v1/n18-2053

[23] Dai Quoc Nguyen, Thanh Vu, Tu Dinh Nguyen, Dat Quoc Nguyen, and Dinh Q. Phung. 2019. A Capsule Network-based Embedding Model for Knowledge Graph Completion and Search Personalization. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, Jill Burstein, Christy Doran, and Thamar Solorio (Eds.). Association for Computational Linguistics, 2180–2189. https://doi.org/10.18653/v1/n19-1226

[24] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*. ACL, 1532–1543. https://doi.org/10.3115/v1/d14-1162

[25] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep Contextualized Word Representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, Marilyn A. Walker, Heng Ji, and Amanda Stent (Eds.). Association for Computational Linguistics, 2227–2237. https://doi.org/10.18653/v1/n18-1202

[26] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan (Eds.). Association for Computational Linguistics, 3980–3990. https://doi.org/10.18653/v1/D19-1410

[27] Michael Sejr Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling Relational Data with Graph Convolutional Networks. In *The Semantic Web - 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3-7, 2018, Proceedings (Lecture Notes in Computer Science)*, Aldo Gangemi, Roberto Navigli, Maria-Esther Vidal, Pascal Hitzler, Raphaël Troncy, Laura Hollink, Anna Tordai, and Mehwish Alam (Eds.), Vol. 10843. Springer, 593–607. https://doi.org/10.1007/978-3-319-93417-4_38

[28] Baoxu Shi and Tim Weninger. 2018. Open-World Knowledge Graph Completion. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, Sheila A. McIlraith and Kilian Q. Weinberger (Eds.). AAAI Press, 1957–1964. https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16055

[29] Richard Socher, Danqi Chen, Christopher D. Manning, and Andrew Y. Ng. 2013. Reasoning With Neural Tensor Networks for Knowledge Base Completion. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, Christopher J. C. Burges, Léon Bottou, Zoubin Ghahramani, and Kilian Q. Weinberger (Eds.). 926–934. http://papers.nips.cc/paper/5028-reasoning-with-neural-tensor-networks-for-knowledge-base-completion

[30] Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: a core of semantic knowledge. In *Proceedings of the 16th International Conference on World Wide Web, WWW 2007, Banff, Alberta, Canada, May 8-12, 2007*, Carey L. Williamson, Mary Ellen Zurko, Peter F. Patel-Schneider, and Prashant J. Shenoy (Eds.). ACM, 697–706. https://doi.org/10.1145/1242572.1242667

[31] Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019. RotatE: Knowledge Graph Embedding by Relational Rotation in Complex Space. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net. https://openreview.net/forum?id=HkgEQnRqYQ

[32] Zhiqing Sun, Shikhar Vashishth, Soumya Sanyal, Partha P. Talukdar, and Yiming Yang. 2020. A Re-evaluation of Knowledge Graph Completion Methods. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel R. Tetreault (Eds.). Association for Computational Linguistics, 5516–5522. https://www.aclweb.org/anthology/2020.acl-main.489/

[33] 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 *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, Lluís Màrquez, Chris Callison-Burch, Jian Su, Daniele Pighin, and Yuval Marton (Eds.). The Association for Computational Linguistics, 1499–1509. https://doi.org/10.18653/v1/d15-1174

[34] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex Embeddings for Simple Link Prediction. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016 (JMLR Workshop and Conference Proceedings)*,

Maria-Florina Balcan and Kilian Q. Weinberger (Eds.), Vol. 48. JMLR.org, 2071–2080. http://proceedings.mlr.press/v48/trouillon16.html

[35] 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 *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (Eds.). 5998–6008. http://papers.nips.cc/paper/7181-attention-is-all-you-need

[36] Denny Vrandecic and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Commun. ACM* 57, 10 (2014), 78–85. https://doi.org/10.1145/2629489

[37] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge Graph and Text Jointly Embedding. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, Alessandro Moschitti, Bo Pang, and Walter Daelemans (Eds.). ACL, 1591–1601. https://doi.org/10.3115/v1/d14-1167

[38] Han Xiao, Minlie Huang, Lian Meng, and Xiaoyan Zhu. 2017. SSP: Semantic Space Projection for Knowledge Graph Embedding with Text Descriptions. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, Satinder P. Singh and Shaul Markovitch (Eds.). AAAI Press, 3104–3110. http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14306

[39] Ruobing Xie, Zhiyuan Liu, Jia Jia, Huanbo Luan, and Maosong Sun. 2016. Representation Learning of Knowledge Graphs with Entity Descriptions. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*, Dale Schuurmans and Michael P. Wellman (Eds.). AAAI Press, 2659–2665. http://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/view/12216

[40] Chenyan Xiong, Russell Power, and Jamie Callan. 2017. Explicit Semantic Ranking for Academic Search via Knowledge Graph Embedding. In *Proceedings of the 26th International Conference on World Wide Web, WWW 2017, Perth, Australia, April 3-7, 2017*, Rick Barrett, Rick Cummings, Eugene Agichtein, and Evgeniy Gabrilovich (Eds.). ACM, 1271–1279. https://doi.org/10.1145/3038912.3052558

[41] Wenhan Xiong, Mo Yu, Shiyu Chang, Xiaoxiao Guo, and William Yang Wang. 2018. One-Shot Relational Learning for Knowledge Graphs. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii (Eds.). Association for Computational Linguistics, 1980–1990. https://doi.org/10.18653/v1/d18-1223

[42] Jiacheng Xu, Xipeng Qiu, Kan Chen, and Xuanjing Huang. 2017. Knowledge Graph Representation with Jointly Structural and Textual Encoding. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, Carles Sierra (Ed.). ijcai.org, 1318–1324. https://doi.org/10.24963/ijcai.2017/183

[43] Ikuya Yamada, Hiroyuki Shindo, Hideaki Takeda, and Yoshiyasu Takefuji. 2016. Joint Learning of the Embedding of Words and Entities for Named Entity Disambiguation. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning, CoNLL 2016, Berlin, Germany, August 11-12, 2016*, Yoav Goldberg and Stefan Riezler (Eds.). ACL, 250–259. https://doi.org/10.18653/v1/k16-1025

[44] 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 *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Yoshua Bengio and Yann LeCun (Eds.). http://arxiv.org/abs/1412.6575

[45] Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. KG-BERT: BERT for Knowledge Graph Completion. *CoRR* abs/1909.03193 (2019). arXiv:1909.03193 http://arxiv.org/abs/1909.03193

[46] Fuzheng Zhang, Nicholas Jing Yuan, Defu Lian, Xing Xie, and Wei-Ying Ma. 2016. Collaborative Knowledge Base Embedding for Recommender Systems. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, Balaji Krishnapuram, Mohak Shah, Alexander J. Smola, Charu C. Aggarwal, Dou Shen, and Rajeev Rastogi (Eds.). ACM, 353–362. https://doi.org/10.1145/2939672.2939673

[47] Shuai Zhang, Yi Tay, Lina Yao, and Qi Liu. 2019. Quaternion Knowledge Graph Embeddings. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett (Eds.). 2731–2741. http://papers.nips.cc/paper/8541-quaternion-knowledge-graph-embeddings

Table 11: The grid searching of hyperparameters. Note, the hyperparameters in the first part, i.e., Batch Size and γ, were tuned based on WN18RR benchmark, RoBERTa initialization, learning rate = 10^{-5}, number of training epochs = 6. After the first was part tuned, the remaining was tuned subsequently.

Hyperparm	Note	Value	Search scope
Batch Size	-	16	{16, 32, 64}
$\|\mathcal{N}(tp)\|$	-	5	{5}
λ	-	1.0	{1.0}
γ	-	1.0	{0.5, 1.0, 2.0}
Learning Rate	RoBERTa	10^{-5}	$\{10^{-5}\}$
	BERT	5×10^{-5}	$\{5 \times 10^{-5}\}$
Number of Training Epochs	WN18RR	7	{6, 7, 8, 9}
	FB15k-237	7	
	NELL-One	8	
	UMLS	20	{5-25}

A TRAINING SETUPS

In training phase, the initialization of Transformer encoder is alternated between BERT and RoBERTa. The model is fine-tuned by Adam optimizer. For the hyperparameters in StAR, based on the best Hits@10 on dev set, we set batch size = 16, learning rate = $10^{-5}/5 \times 10^{-5}$ for the models initialized with RoBERTa and BERT respectively, number of training epochs = 7 on WN18RR and FB15k-237, 8 on NELL-One, 20 on UMLS, $|\mathcal{N}(tp)| = 5$, $\lambda = 1$ in Eq.(13), and $\gamma = 1$ in Eq.(14). As for grid searching of hyperparameters, we list the searching scopes and the tuned hyperparameters for best in Table 11. Note, we sampled 5 negative triples for each positive triple by following Yao et al. [45] without any tuning, and we also did not tune the random seed while kept the same among the experiments.

For the hyperparameters in self-adaptive ensemble scheme, based on the best Hits@10 on WN18RR/FB15k-237 dev set, we set batch size = 32/64, learning rate = $10^{-3}/10^{-5}$, number of training epochs = 1, number of negative samples = 5/10, and margin = 0.60/0.44 in hinge loss function.

B PROBING TASKS

The *first probing task* keeps training set unchanged but makes the test set only consist of the test triples involving unseen entities. And, in *second probing task*, we conducted a more reasonable comparison by supporting inductive representations [14] for unseen entities in a translation-based approach, and thus made following changes : (1) 1900 entities were sampled from test set, and only a test triple containing at least one of the sampled entities can be kept, resulting in 1758 test triples in this probing task; (2) Those training triples that do not contain the sampled entities are used as new training set; and (3) Those training triples containing exact one of the sampled entities are used as support set to inductively generate the embedding for the unseen entities via translation formula, such as "$\boldsymbol{h} + \boldsymbol{r} = \boldsymbol{t}$" in TransE [3]. Using the second probing setting can assign the unseen entities with competent embeddings, thus leading to a fairer comparison than the first one. Note, if an unseen entity is involved in multiple triple on the support set, an average over the multiple inductive representations is used as its single vector representation. ,