The Integration of Semantic and Structural Knowledge in Knowledge Graph Entity Typing

Muzhi Li[1], Minda Hu[1], Irwin King[1], Ho-fung Leung[2]
[1]Dept. of Computer Science & Engineering, The Chinese University of Hong Kong
[2]Independent Researcher
{mzli,mindahu21,king}@cse.cuhk.edu.hk
ho-fung.leung@outlook.com

Abstract

The Knowledge Graph Entity Typing (KGET) task aims to predict missing type annotations for entities in knowledge graphs. Recent works only utilize the ***structural knowledge*** in the local neighborhood of entities, disregarding ***semantic knowledge*** in the textual representations of entities, relations, and types that are also crucial for type inference. Additionally, we observe that the interaction between semantic and structural knowledge can be utilized to address the false-negative problem. In this paper, we propose a novel **S**emantic and **S**tructure-aware KG **E**ntity **T**yping (SSET) framework, which is composed of three modules. First, the *Semantic Knowledge Encoding* module encodes factual knowledge in the KG with a Masked Entity Typing task. Then, the *Structural Knowledge Aggregation* module aggregates knowledge from the multi-hop neighborhood of entities to infer missing types. Finally, the *Unsupervised Type Re-ranking* module utilizes the inference results from the two models above to generate type predictions that are robust to false-negative samples. Extensive experiments show that SSET significantly outperforms existing state-of-the-art methods. [1]

1 Introduction

A knowledge graph (KG) is a graph-structured knowledge base that consists of triples in the form of (*head entity*, *relation*, *tail entity*). Each entity in the KG is annotated with one or multiple types. For example, entity "*Albert Einstein*" belongs to the "*/scientist/physicist*" type. Types (or *concepts*) in KGs provide a high-level summary of their instance entities, which is crucial in many natural language processing (NLP) tasks such as entity linking (Chen et al., 2020), relation extraction (Wang et al., 2019), and question answering (Hu et al., 2022b).

Currently, most KGs (e.g., Freebase (Bollacker et al., 2008) and YAGO (Suchanek et al., 2007)) are manually constructed by domain experts, which the type annotations are usually incomplete. The statistics on the FB15kET dataset show that 10% of the entities with the type "*/music/artist*" do not have type "*/people/person*" (Moon et al., 2017b). In light of this, we focus on the Knowledge Graph Entity Typing (KGET) task, which aims to infer missing type annotations for entities in a KG.

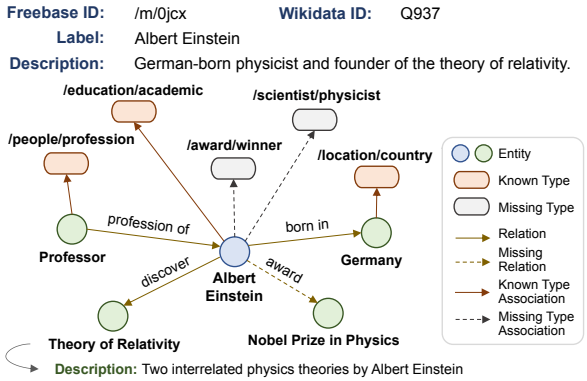

Figure 1: A knowledge graph fragment centered on entity "*Albert Einstein*". We aim to infer missing types of the target entity based on the structural and textual information provided in the local subgraph.

Recent studies in the KGET task can be categorized into three groups: (1) embedding-based methods (Moon et al., 2017b; Zhao et al., 2020), (2) GNN-based methods (Zhao et al., 2022; Zhuo et al., 2022; Pan et al., 2021; Jin et al., 2022), and (3) transformer-based methods (Hu et al., 2022a). However, each group of solutions has its drawbacks. Specifically, embedding-based methods ignore the rich neighbor information of entities. GNN-based methods fall short of capturing the interactions between non-directly connected neighbors of the target entity (Hu et al., 2022a). The only transformer-based method TET suffers from its high computational complexity. More importantly, all existing methods *ignore the textual information in the KG* and *suffer from serious false-negative problem* (Jin et al., 2022), which ultimately limits the type inference performance.

[1]https://github.com/RaynorLEE/KG-EntityTyping

Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), pages 6625–6638

We argue that *the textual representations of entities, relations, and types provide important semantic knowledge for type inference.* For example, in Figure 1, it is hard to infer that "*Albert Einstein*" belongs to the type "*/scientist/physicist*" based solely upon the structural knowledge since triple (*Albert Einstein*, *award*, *Nobel Prize in Physics*) is missing from the KG. Fortunately, the description of the neighbor entity "*Theory of Relativity*" reveals that the theory is in the domain of "*physics*", which can help us to conclude that "*Albert Einstein*" has "*/scientist/physicist*" type.

Moreover, we observe that *semantic and structural knowledge can complement each other to alleviate the false-negative problem.* Some valid type annotations are plausible but missing from the KG. Simply considering these missing type annotations as negative samples will result in significant performance degradation. We find that the semantic knowledge obtained from the textual information provides an informative prior for the alleviation of the false-negative problem. If both semantic and structural knowledge support that an entity should belong to a particular type, we can assign a higher relevance score to that type. Such an agreement allows us to mitigate the impact of false-negative samples with high confidence. In summary, the main objective of this paper is to capture and integrate semantic and structural knowledge from the local subgraph of each entity in the KG and apply both types of knowledge to the entity typing task.

Based on the aforementioned observations, we propose a novel **S**emantic and **S**tructure-aware KG **E**ntity **T**yping (SSET) framework, which consists of three modules. The *Semantic Knowledge Encoding module* encodes factual knowledge in the KG with a Masked Entity Typing task. The *Structural Knowledge Aggregation module* leverages knowledge from 1-hop neighbors, multi-hop neighbors, and known types of entities to predict missing types. The *Unsupervised Type Re-ranking module* utilizes the preliminary inference results obtained from the two modules mentioned above to generate type predictions that are robust to false-negative samples. Our contributions are summarized as follows:

- We propose a novel framework SSET which encodes and utilizes semantic and structural knowledge for the KGET task.
- We notice the complementary relationship between semantic and structural knowledge and integrate the two types of knowledge to alleviate the impact of false-negative samples.
- We conduct empirical and ablation experiments on two widely used datasets, showing that SSET significantly outperforms the existing state-of-the-art approaches.

2 Related Work

Embedding-based Methods. ETE (Moon et al., 2017b) is the first model specialized for the KGET task. It employs KG embedding methods (Bordes et al., 2013; Moon et al., 2017a) to embed entities and relations, and learns type embeddings by minimizing the distance between embeddings of entities and corresponding types. ConnectE (Zhao et al., 2020) jointly embeds entities and types into different spaces and proposes two type inference mechanisms, namely "E2T" and "TRT". Despite the simplicity and intuitiveness, these methods ignore the rich neighbor information of entities, which seriously limits their performance (Jin et al., 2022).

Graph Neural Network-based Methods. Graph neural networks (GNNs) (Kipf and Welling, 2017; Song et al., 2022) have been proven to be effective in capturing structural knowledge from graph-structured data, including KGs. Recently, several GNNs such as RGCN (Schlichtkrull et al., 2018), ConnectE-MRGAT (Zhao et al., 2022) have been proposed to better aggregate neighbor information in KGs and are applied in the KGET task. Most notably, CET (Pan et al., 2021) leverages knowledge from each neighbor of an entity in an independent and aggregated manner to mitigate the impact of noises from irrelevant neighbors. MiNer (Jin et al., 2022) further extends the scope of knowledge aggregation to multi-hop neighbors. Nevertheless, these methods primarily focus on modeling the graph structure and overlook the textual information provided in KGs.

Transformer-based Methods. To the best of our knowledge, TET (Hu et al., 2022a) is the only transformer-based method specified for the KGET task. TET utilizes three transformers to encode the neighbors of each entity and model the knowledge interactions between different neighbors. However, these transformers have no relationship with pre-trained language models (PLMs) (Devlin et al., 2019; Su et al., 2022), which cannot model semantic knowledge in the textual representations of entities, relations, and types.

Semantic-aware Fine-grained Entity Typing. Recently, several methods have been proposed to incorporate semantic information into the fine-grained entity typing (FET) task (Jin et al., 2019). Notably, Cat2Type (Biswas et al., 2021) leverages Wikipedia category names to embed entity mentions for type prediction. Biswas et al. (2022) further employs graph walks to aggregate neighborhood information of entities from an external knowledge base. It should be noted that the FET task focuses on inferring the semantic type of an entity mention within plain text. Due to the differences in problem settings, it is infeasible to directly apply these FET solutions to the KGET task.

3 Problem Specification

We consider a KG $\mathcal{G}$ to consist of triples in the form of (e_s, r, e_o) and entity type assertions in the form of (e, t), where $e_s, e_o, e \in \mathcal{E}$, $r \in \mathcal{R}$, $t \in \mathcal{T}$. The notations e_s and e_o denote the subject and the object of a triple. $\mathcal{E}, \mathcal{R}, \mathcal{T}$ are the set of entities, relations, and types, respectively. In KGs, an entity can have multiple types, but some type annotations are missing from the KG. Here, we use $\mathcal{T}(e) = \{t|(e,t) \in \mathcal{G}\}$ to denote the set of known types of entity e. We denote the set of triples related to entity e by $\mathcal{N}(e) = \{(e, r, \tilde{e}) \cup (\tilde{e}, r, e) \in \mathcal{G}\}$. In most KG datasets such as FB15k (Bollacker et al., 2008) and YAGO43k (Suchanek et al., 2007), entities are provided with *labels* and *descriptions*, while relations and types are represented by their *textual identifiers*. We assume that these *textual information* are meaningful and contain valuable semantic knowledge for the KGET task. In this paper, we aim to reveal missing type annotations based on the local graph structure of entities and corresponding textual information.

4 Method

Figure 2 provides an overview of the architecture of our proposed SSET framework, which consists of three modules. First, the *1) Semantic Knowledge Encoding module* (SEM) infuses the knowledge from triples and entity-type assertions into the pre-trained language model (PLM). We expect the fine-tuned language model can be utilized to encode textual representations of entities, relations, and types, and to conduct preliminary type inference. Then the *2) Structural knowledge aggregation module* (SKA) aggregates knowledge from the local subgraph of each target entity and utilizes 1-hop neighbors, multi-hop neighbors, and known types of the entity to infer its missings types. Finally, the *3) Unsupervised Type Re-ranking module (UTR)* exploits the agreement between semantic and structural knowledge to alleviate the false-negative problem and generate the final type rankings.

4.1 Step 1: Semantic Knowledge Encoding

To incorporate the factual knowledge from the KG into the PLM, we propose a fine-tuning task known as Masked Entity Typing (MET). The MET task recovers missing types of the masked entity by utilizing *triples* and *entity-type assertions* that contain the entity. Since triples and entity-type assertions in KGs are differ from natural language sentences, we will first introduce the tokenization process before delving into the details of the MET task.

4.1.1 Tokenization

Before being inputted into the PLM encoder, triples and entity-type assertions need to be serialized into token sequences. Formally, we tokenize each triple (e_s, r, e_o) into the following sequence:

$$\text{[CLS] text}_{e_s} \text{ [SEP] text}_r \text{ [SEP] text}_{e_o} \text{ [SEP]}, \quad (1)$$

where $\text{text}_e = [\text{label}_e, \text{desc}_e]$ contains the label and the description of entity e, "text_r" is the textual identifier of relation r, e_s and e_o differentiates the subject and object of a triple. Similarly, we tokenize each entity-type assertion as follows:

$$\text{[CLS] text}_e \text{ [SEP] } \textit{has type} \text{ [SEP] text}_t \text{ [SEP]}, \quad (2)$$

where "*has type*" is a predicate connecting an entity and a type, "text_t" is the textual identifier of type t. We observed that types within some KGs such as FB15k (Bollacker et al., 2008) usually have multiple levels (e.g., "*/scientist/physicist*"). During the tokenization process, different levels of a type will be separated with a [SEP] token (e.g., "*scientist* [SEP] *physicist*").

4.1.2 Masked Entity Typing (MET) Task

In KGs, the triples and known types of an entity contribute differently to the inference of a specific missing type. For example, the triple (*Albert Einstein, award, Nobel Prize*) indicates that the entity "Albert Einstein" has the type "/award/winner". However, the triple (*Albert Einstein, born in, Germany*) does not provide relevant support for the same conclusion. To mitigate the impact of irrelevant information, we separately encode the token sequences of each triple and entity-type assertion that involves the target entity.

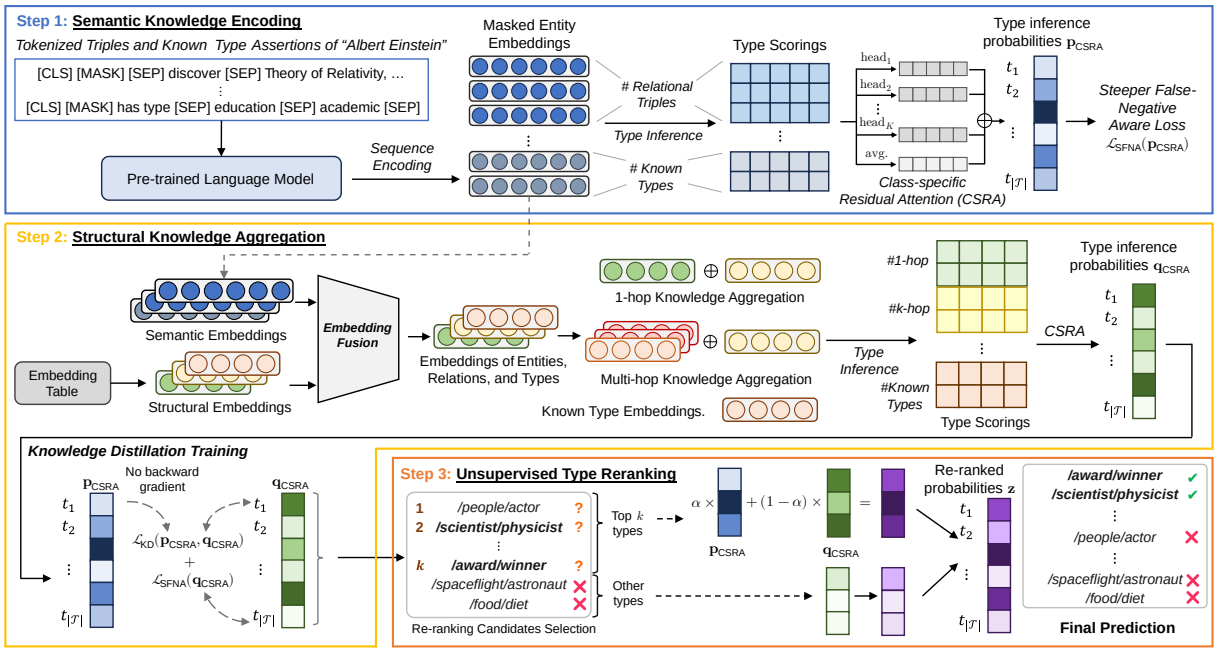

Figure 2: The end-to-end architecture of the SSET, consists of three modules: Semantic Knowledge Encoding module (top), Structural Knowledge Aggregation module (mid), and Unsupervised Type Re-Ranking module (bot).

To prevent the PLM from overfitting to the label and description of the target entity, we mask the tokens that correspond to the target entity in all relevant token sequences mentioned above. This operation forces the PLM to capture semantic knowledge from the textual representations of relations, neighboring entities, and known types. Subsequently, the masked sequences are inputted into the language model (LM) encoder (Devlin et al., 2019; Su et al., 2022).[2] Finally, we apply a non-linear classifier to the embeddings of the [MASK] tokens obtained from the output of the LM encoder:

$$\mathbf{S} = \mathbf{W}\sigma(\mathbf{H}^{\text{mask}}) + \mathbf{b}, \tag{3}$$

where $\mathbf{W} \in \mathbb{R}^{|\mathcal{T}| \times d}$ and $\mathbf{b} \in \mathbb{R}^{d}$ are the learnable parameters. $\mathbf{H}^{\text{mask}} = [\mathbf{h}_1^{\text{mask}}, \mathbf{h}_2^{\text{mask}}, \cdots, \mathbf{h}_{m+n}^{\text{mask}}] \in \mathbb{R}^{(m+n)\times d}$ are the [MASK] token embeddings from the m triples and n known types of the target entity. $\sigma(\cdot)$ denotes the ELU activation function. The output $\mathbf{S} = [\mathbf{s}_1, \mathbf{s}_2, \cdots, \mathbf{s}_{m+n}] \in \mathbb{R}^{(m+n)\times|\mathcal{T}|}$ consists of type scoring vectors for the m triples and n known types. In particular, s_i^j represents the relevance score between the i-th triple (or known type) of the entity and the j-th type in the KG.

Following the state-of-the-art KGET model MiNer (Jin et al., 2022), we adopt the class-specific residual attention mechanism (CSRA) (Zhu and Wu, 2021) to generate the preliminary type inference result. CSRA suggests utilizing H different temperatures $T_1, \cdots, T_H$ to pool the type relevance scores of the m triples or n known types. This process yields preliminary inference probabilities $\mathbf{p}_{\text{CSRA}} = [p^1, p^2, \cdots, p^{|\mathcal{T}|}]$ that the target entity belongs to each of the $|\mathcal{T}|$ types:

$$\mathbf{p}_{\text{CSRA}} = \text{sigmoid}\Big(\sum_{h=1}^{H}\big(\mathbf{s}_{T_h} + \frac{1}{m+n}\sum_{i=1}^{m+n}\mathbf{s}_i\big)\Big), \tag{4}$$

where $\mathbf{s}_T = [s_T^1, \cdots, s_T^{|\mathcal{T}|}] \in \mathbb{R}^{|\mathcal{T}|}$ is the pooled type scoring vector under temperature T. Here, we have

$$s_T^j = \sum_{i=1}^{m+n} \frac{\exp\left(T s_i^j\right)}{\sum_{k=1}^{m+n}\exp\left(T s_k^j\right)} \cdot s_i^j. \tag{5}$$

We utilize the steeper false-negative aware (SFNA) loss (Hu et al., 2022a) to fine-tune the PLM:

$$\mathcal{L}_{\text{SFNA}}(\mathbf{p}_{\text{CSRA}}) = \sum_{(e_i,t_j)\notin\mathcal{G}} f(p_i^j)\log(1-p_i^j) - \sum_{(e_i,t_j)\in\mathcal{G}} \log(p_i^j), \tag{6}$$

where p_i^j is the probability that entity i belongs to type j, $f(\cdot)$ is a re-weighting function (see Eq. (16)) that penalizes negative samples with too high or too low probabilities.

[2]We utilize pre-trained BERT and RoBERTa encoder as the backbone of the SEM module due to their simplicity. The proposed SSET framework can be generalized to any PLMs that can be fine-tuned such as GPT.

4.2 Step 2: Structural Knowledge Aggregation

In KGs, an entity's knowledge is manifested within its local subgraph. Simply considering the one-hop neighbors and known types of an entity is insufficient to reveal all of its missing types. The inference of certain challenging types necessitates a comprehensive understanding derived from the multi-hop neighbors of the target entity. Hence, in this module, we aim to utilize 1-*hop neighbors*, *multi-hop neighbors*, and the *known types* of the target entity to predict missing types. [3]

4.2.1 Embedding Fusion

Ideally, multi-hop structural knowledge improves the performance of type inference. However, without good initial embeddings, the inference results will be unreliable. In such cases, the SKA module cannot receive meaningful gradients to update the parameters and may learn to disregard the structural knowledge (Hu et al., 2022c).

To overcome the "cold-start" problem, we embed and cache textual representations of entities, relations, and types (denoted as $\mathbf{h}_e^{(t)}$, $\mathbf{h}_r^{(t)}$, and $\mathbf{h}_t^{(t)}$) from the fine-tuned LM. These embeddings are fixed during the training process, as the textual representations are independent of the KG's connectivity. In addition, to enhance the scalability of our model, we also incorporate a set of learnable structural embeddings for each entity, relation, and type, which are denoted as $\mathbf{h}_e^{(s)}$, $\mathbf{h}_r^{(s)}$, and $\mathbf{h}_t^{(s)}$.

Before knowledge aggregation, we fuse the textual and structural embeddings of each entity, relation, and type into a unified latent space. This is achieved by two multi-layer perceptions (MLPs) followed by a $\mathcal{L}_2$ normalization. Formally, we have

$$\mathbf{h}_{\{\cdot\}} = \frac{\text{MLP}_t(\mathbf{h}_{\{\cdot\}}^{(t)})}{||\text{MLP}_t(\mathbf{h}_{\{\cdot\}}^{(t)})||_2} + \frac{\text{MLP}_s(\mathbf{h}_{\{\cdot\}}^{(s)})}{||\text{MLP}_s(\mathbf{h}_{\{\cdot\}}^{(s)})||_2}, \quad (7)$$

where $\mathbf{h}_{\{\cdot\}}$ is the *unified knowledge embedding* of an entity, a relation, or a type. For simplicity, we denote $\mathbf{h}_e, \mathbf{h}_r,$ and $\mathbf{h}_t$ as $\mathbf{e}, \mathbf{r},$ and $\mathbf{t}$ hereinafter.

4.2.2 One-hop Knowledge Aggregation

Following the convention adopted in CET (Pan et al., 2021) and MiNer (Jin et al., 2022), we utilize TransE (Bordes et al., 2013) to aggregate the knowledge from each of the triples containing the target entity. Formally, for target entity e, its representation aggregated from the one-hop neighbor entity $\tilde{e}$ and the relation r can be computed as:

$$\mathbf{h}_{(r,\tilde{e})}^{(1)} = (\tilde{\mathbf{e}} - \mathbf{r}), \quad (8)$$

where $\tilde{\mathbf{e}}$ and $\mathbf{r}$ are the embeddings of entity $\tilde{e}$ and relation r, $(e, r, \tilde{e})$ is a triple in the KG.[4]

4.2.3 Multi-hop Knowledge Aggregation

Now, we introduce how we aggregate structural knowledge from the multi-hop neighborhood of the target entity. Inspired by MiNer (Jin et al., 2022), we iteratively represent each $(k-1)$-hop neighbor entity by its known types and related k-hop entities. We use 2-hop neighbors as an example to illustrate the knowledge aggregation process and will generalize to the case of more hops.

Considering the target entity e, we encode the 2-hop structural knowledge via each 1-hop neighbor connected to the entity:

$$\mathbf{h}_{(r,\tilde{e})}^{(2)} = f_{\text{agg}}^{(1)}(\tilde{e}) - \mathbf{r}, \quad (9)$$

where $\mathbf{h}_{(r,\tilde{e})}^{(2)}$ is the representation of entity e aggregated from the 2-hop neighborhood via the 1-hop neighbor $\tilde{e}$. $f_{\text{agg}}^{(1)}$ is an aggregation function that computes the representation of an entity with its related entities and known types:

$$f_{\text{agg}}^{(1)}(e) = \frac{\sum_{(e,r,\tilde{e})\in\mathcal{N}(e)} (\tilde{\mathbf{e}} - \mathbf{r}) + \sum_{t\in\mathcal{T}(e)} \mathbf{t}}{|\mathcal{N}(e)| + |\mathcal{T}(e)|}, \quad (10)$$

where $\mathcal{N}(e)$ is the set of triples containing entity e, $\mathcal{T}(e)$ is the set of entity e's known types, $\mathbf{t}$ is the embedding of type t. We can iteratively generalize the knowledge aggregation process to multi-hop neighbors $(k \geq 3)$ of the target entity with the following two equations:

$$\mathbf{h}_{(r,\tilde{e})}^{(k)} = f_{\text{agg}}^{(k-1)}(\tilde{e}) - \mathbf{r}, \quad (11)$$

$$f_{\text{agg}}^{(k)}(e) = \frac{\sum_{(e,r,\tilde{e})\in\mathcal{N}(e)} (f_{\text{agg}}^{(k-1)}(\tilde{e}) - \mathbf{r}) + \sum_{t\in\mathcal{T}(e)} \mathbf{t}}{|\mathcal{N}(e)| + |\mathcal{T}(e)|}. \quad (12)$$

4.2.4 Type Inference

We construct a matrix $\mathbf{H}^{\text{agg}} = [\mathbf{h}_{(r_1,\tilde{e}_1)}^{(1)}, \cdots, \mathbf{h}_{(r_m,\tilde{e}_m)}^{(K)}, \mathbf{t}_1, \cdots, \mathbf{t}_n]$ by concatenating the entity representations aggregated from the $1, 2, \cdots, K$-hop neighborhood and the embeddings of each

[3] We do not treat types of an entity as its neighbors. Distinguishing entity types from entities helps us to achieve better results.

[4] We also consider the neighbor entities $\tilde{e}$ connected by reversed relations r^{-1} (i.e. $(\tilde{e}, r^{-1}, e) \in \mathcal{G}$). We embed reversed relations by the negation of the embedding of the original relation s.t. $\mathbf{r}^{-1} = -\mathbf{r}$.

known type. Similar to the SEM module, we apply a non-linear classifier to the embedding matrix and utilize the CSRA (Zhu and Wu, 2021) mechanism to compute the probabilities $\mathbf{q}_{\text{CSRA}} = [q^1, q^2, \cdots, q^{|\mathcal{T}|}]$ that the entity belongs to each type.

4.2.5 Knowledge Distillation

Some plausible type annotations are missing from the KG. Simply considering missing types as negative samples will result in a serious false negative problem. We find that the preliminary type inference results obtained from the SEM module provide an informative prior that can be leveraged to alleviate the impact of those false-negative samples. Hence, we add the following knowledge distillation (KD) loss to align the inference results obtained from the SEM and the SKA modules:

$$\begin{aligned}\mathcal{L}_{\text{KD}}(p, q) = &\sum_{i=1}^{|\mathcal{E}|}\sum_{j=1}^{|\mathcal{T}|}(1-p_i^j)f(q_i^j)\log(1-q_i^j) \\ &-\sum_{i=1}^{|\mathcal{E}|}\sum_{j=1}^{|\mathcal{T}|}p_i^j\log(q_i^j),\end{aligned} \tag{13}$$

where p_i^j, q_i^j are the probabilities that entity i belongs to type j obtained from the two modules, respectively. $f(\cdot)$ is the re-weighting function defined in Eq. (16). The training objective for the SKA module is a weighted average of the KD loss and the SFNA loss:

$$\begin{aligned}\mathcal{L}_{\text{SKA}} = &\lambda \cdot \mathcal{L}_{\text{SFNA}}(\mathbf{q}_{\text{CSRA}}) \\ &+ (1-\lambda)\cdot \mathcal{L}_{\text{KD}}(\mathbf{p}_{\text{CSRA}}, \mathbf{q}_{\text{CSRA}}),\end{aligned} \tag{14}$$

where $\lambda \in [0, 1]$ is a hyperparameter balancing the two loss functions.

4.3 Step 3: Unsupervised Type Re-ranking

During the inference stage, the SEM and SKA modules generate two sets of probabilities to assess the likelihood of each entity belonging to each type in the KG. Motivated by the complementary nature of semantic and structural knowledge, we leverage the agreement between the probability outputs of both modules to generate our final type predictions.

Specifically, if both types of knowledge support that an entity belongs to a particular type, we will simultaneously obtain high probabilities from both modules. In such cases, we assign a lower (better) rank to the type with confidence. Conversely, if either module generates a low probability for a type, we assign it a higher rank. To achieve the aforementioned goals, we employ the re-ranking mechanism as in (Lovelace et al., 2021). Specifically, for each entity i, we retrieve the top k candidate types with the highest probabilities obtained from the SKA module. Then we re-evaluate the rankings of these types by taking a weighted average of the probabilities obtained from both modules. Formally, we have

$$\mathbf{z}_i^j = \begin{cases}\alpha \cdot p_i^j + (1-\alpha)\cdot q_i^j & \text{Rank}(q_i^j) \le k \\ q_i^j & \text{otherwise}\end{cases}, \tag{15}$$

where $0 \le \alpha \le 1$ is the hyperparameter that controls the balance between the two modules, z_i^j is the aggregated probability that entity i belongs to type j. These probabilities are ranked in descending order for model evaluation. Note that the aggregated probabilities will not be utilized to optimize the SEM or SKA modules. Such an *unsupervised setting* prevents the final type predictions from being affected by false-negative samples.

5 Experiments

5.1 Datasets

We evaluate our model on two real-world KGs: FB15k (Bordes et al., 2013) and YAGO43k (Moon et al., 2017b), which are derived from Google Freebase (Bollacker et al., 2008) and YAGO knowledge base (Suchanek et al., 2007), respectively. Two entity typing datasets FB15kET (Xie et al., 2016b) and YAGO43kET (Moon et al., 2017b) provide entity-type assertions by mapping entities from the two KGs to their entity types. Regarding the textual information, we use the labels and descriptions of FB15k entities released by Xie et al. (2016a). The YAGO43k dataset provides textual labels for each entity. We use Wikidata API to collect descriptions of entities in YAGO43k since the YAGO knowledge base is built upon Wikidata (Vrandečić and Krötzsch, 2014).[5] Relations and types in the two datasets are represented in textual identifiers. The statistics of datasets are shown in Appendix B.

5.2 Baselines

In this section, we compare our proposed SSET model with 14 baselines. Specifically, we consider traditional KG embedding methods TransE (Bordes et al., 2013), ComplEx (Trouillon et al., 2016), RotatE (Sun et al., 2019), and CompoundE (Ge et al., 2023); text-based KG completion method

[5]The YAGO knowledge base assigns a distinct Wikidata ID to each of its entities, enabling us to access relevant information such as descriptions from the Wikidata API.

Methods	FB15kET					YAGO43kET				
	Hit@1	Hit@3	Hit@10	MR	MRR	Hit@1	Hit@3	Hit@10	MR	MRR
Embedding-based methods										
TransE	0.504	0.686	0.835	18	0.618	0.304	0.497	0.663	393	0.427
ComplEx	0.463	0.680	0.841	20	0.595	0.316	0.504	0.658	631	0.435
RotatE	0.523	0.699	0.840	18	0.632	0.339	0.537	0.695	316	0.462
CompoundE	0.525	0.719	0.859	-	0.640	0.364	0.558	0.703	-	0.480
SimKGC	0.210	0.348	0.545	46	0.317	0.097	0.184	0.317	259	0.172
ETE	0.385	0.553	0.719	-	0.500	0.137	0.263	0.422	-	0.230
ConnectE	0.496	0.643	0.799	42	0.590	0.160	0.309	0.479	-	0.280
Graph Neural Network-based methods										
MRGAT	0.562	0.663	0.804	-	0.630	0.243	0.343	0.482	-	0.320
RACE2T	0.561	0.688	0.817	-	0.646	0.248	0.376	0.523	-	0.344
AttEt	0.517	0.677	0.821	-	0.620	0.244	0.413	0.565	-	0.350
RGCN	0.597	0.722	0.843	20	0.679	0.281	0.409	0.549	397	0.372
CET	0.613	0.745	0.856	19	0.697	0.398	0.567	0.696	250	0.503
MiNer	0.654	0.768	0.875	15	0.728	0.412	0.589	0.714	223	0.521
Transformer-based methods										
TET	0.638	0.762	0.872	-	0.717	0.408	0.571	0.695	-	0.510
SSET (w/ BERT)	0.693	0.800	0.895	12	0.761	0.473	0.644	**0.762**	244	0.576
SSET (w/ RoBERTa)	**0.697**	**0.804**	**0.898**	**11**	**0.765**	**0.476**	**0.647**	**0.762**	**212**	**0.579**

Table 1: Experiment Results of KGET on FB15kET and YAGO43kET datasets. The best results are in **bold** and the second-best ones are underlined. All results of baseline methods are referred from corresponding original papers except SimKGC. For SimKGC, we generate the results with the publicly available implementation.

Exp.	Model Settings	FB15kET					YAGO43kET				
		Hit@1	Hit@3	Hit@10	MR	MRR	Hit@1	Hit@3	Hit@10	MR	MRR
1	SEM (MET) only	0.688	0.797	0.894	12	0.758	0.468	0.636	0.756	**162**	0.571
2	SKA (w/o textual emb. & KD)	0.671	0.778	0.878	15	0.741	0.457	0.614	0.730	332	0.555
3	pre-trained RoBERTa + SKA (w/o KD)	0.665	0.780	0.881	15	0.738	0.451	0.618	0.732	344	0.553
4	SEM + SKA (w/o KD)	0.677	0.787	0.886	13	0.748	0.458	0.622	0.740	265	0.558
5	SEM + SKA (w/ KD)	0.678	0.792	0.892	12	0.750	0.461	0.633	0.751	213	0.566
6	SEM + SKA (w/ KD) + UTR	**0.697**	**0.804**	**0.898**	**11**	**0.765**	**0.476**	**0.647**	**0.762**	212	**0.579**

Table 2: Results for ablation studies with RoBERTa-base as backbone language model. The best results are in **bold**.

SimKGC (Wang et al., 2022), embedding-based KGET methods ETE (Moon et al., 2017b) and ConnectE (Zhao et al., 2020); GNN-based methods ConnectE-MRGAT (Zhao et al., 2022), RACE2T (Zou et al., 2022), RGCN (Schlichtkrull et al., 2018), AttEt (Zhuo et al., 2022), CET (Pan et al., 2021), and MiNer (Jin et al., 2022); as well as transformer-based method TET (Hu et al., 2022a).

5.3 Main Results

Table 1 summarizes the performance of SSET and all baselines on the two datasets. The experiment results show that the proposed model SSET significantly and consistently outperforms all baseline methods. Particularly, compared to the previous state-of-the-art method MiNer (Jin et al., 2022), SSET achieves an absolute Hit@1 improvement of 4.3% and 6.4% on the FB15kET and the YAGO43kET datasets with a RoBERT-base textual encoder. It demonstrates that SSET is highly effective for the KGET task.

Traditional KGE methods, such as TransE, ComplEx, RotatE, and CompoundE, cannot achieve desirable performance as they consider KGET as a link prediction task, which assumes that entities and their types are connected with a special relation "*has type*". Due to the significant differences in textual semantics between entities and types, the text-based KGC model SimKGC lags far behind in the KGET task. GNN-based methods outperform embedding-based KGET methods ETE and ConnectE as GNNs can better capture the structural knowledge of entities within the local subgraph. It should be noted that RGCN, CET and MiNer perform better than other GNN-based methods since they consider KGET as a multi-label node classification task (Song et al., 2021). This setting allows the model to mark all types that are not annotated to the target entity as negative samples, which typically improves inference performance.

We have evaluated two variants of SSET to assess the effects of selecting different backbone

Methods	k-hop	FB15kET					YAGO43kET				
		Hit@1	Hit@3	Hit@10	MR	MRR	Hit@1	Hit@3	Hit@10	MR	MRR
RGCN	1-hop	0.597	0.722	0.843	20	0.679	0.281	0.409	0.549	397	0.372
RGCN	2-hop	0.580	0.709	0.830	29	0.664	0.243	0.343	0.482	587	0.360
MiNer	1-hop	0.637	0.761	0.865	18	0.716	0.402	0.580	0.710	245	0.512
MiNer	2-hop	0.653	0.764	0.871	15	0.726	0.412	0.589	0.714	223	0.521
MiNer	3-hop	0.651	0.766	0.872	15	0.726	0.411	0.589	0.713	224	0.520
SSET (w/ RoBERTa)	1-hop	0.696	0.802	0.896	**11**	0.764	0.473	0.645	0.761	213	0.576
SSET (w/ RoBERTa)	2-hop	**0.697**	**0.804**	**0.898**	**11**	**0.765**	0.475	**0.647**	**0.762**	**206**	0.578
SSET (w/ RoBERTa)	3-hop	0.696	0.803	**0.898**	12	0.764	**0.476**	**0.647**	**0.762**	212	**0.579**

Table 3: Results for ablation studies with different scopes of the local neighborhood. The best results are in **bold**.

language models. The experiment results show that both BERT and RoBERTa can help SSET to achieve state-of-the-art performance. Furthermore, employing a larger language model (RoBERTa) leads to better performance. We also observe that the performance improvement on the YAGO43kET dataset is particularly remarkable compared to the FB15kET dataset. This can be attributed to the sparser graph structure of YAGO43kET, where entities have significantly fewer neighbors (7.8 on avg.) than FB15kET (32.3 on avg.). In such cases, leveraging textual semantics of entities, relations, and types becomes crucially important.

In addition, the performance gain on Hit@1 is more pronounced compared to Hit@3 and Hit@10 on both datasets, highlighting the superior ability of our model to identify the most suitable type for a given entity. This can be explained by the incorporation of the unsupervised type re-ranking module, which ensures that the top-ranked types are simultaneously supported by both semantic and structural knowledge. We will provide relevant examples in the case study section.

5.4 Ablation Studies

We verify the effectiveness of each component in the SSET model by answering the following research questions (RQ). Table 2 shows the experiment results for ablation studies.

RQ1: Can the SEM and the SKA module generate plausible inference results? In **Exp. 1**, we directly utilize the MET fine-tuning task proposed in the SEM module to predict missing types. From the experiment results, we observe that the SEM module significantly and consistently outperforms all baseline methods across all metrics. With semantic knowledge from the textual representations in KGs, the SEM module only requires minimal structural knowledge (1-hop triples and known types of the target entity) to achieve significant performance improvement. In contrast, the GNN-based method MiNer (Jin et al., 2022) needs to consider 4-hop neighbors of each entity to achieve its best performance on FB15kET. This shows that the presence of textual semantics can compensate for the absence of multi-hop neighbors.

In **Exp. 2**, we utilize the SKA module to perform KGET. The embeddings of entities, relations, and types are randomly initialized and will be optimized as model parameters. Without additional knowledge from the textual representations, the proposed SKA module still significantly outperforms all baseline methods.

RQ2: Does language model fine-tuning improve the model performance? In **Exp. 3** and **Exp. 4**, we separately employ the pre-trained RoBERTa model and the fine-tuned RoBERTa model (obtained from the SEM module) to embed textual information of entities, relations, and types. The performance gap between **Exp. 2** and **Exp. 4** indicates that the textual embeddings from the fine-tuned RoBERTa model provide crucial semantic knowledge for type inference. However, the expected performance improvement is not observed in **Exp. 3** since the pre-trained RoBERTa model lacks factual knowledge from the KG, which confirms the effectiveness and necessity of the SEM module.

RQ3: Does knowledge distillation alleviate the false-negative problem? Among all metrics, the Mean Rank (MR) is more susceptible to the false negative problem since types with a high rank significantly increase the MR value. In **Exp. 5**, we incorporate the knowledge distillation (KD) loss into the training objective of the SKA module. Comparing the results between **Exp. 4** and **Exp. 5**, the MR exhibits a significant decrease (7.7% on FB15kET and 19.6% on YAGO43kET), empirically confirming that the KD loss can mitigate the impact of readily identifiable false-negative samples.

Entity	Missing Type	Module	Rank #1 Type	Rank #1 Score	Rank #2 Type	Rank #2 Score	Rank #3 Type	Rank #3 Score
/m/0k8z (Apple Inc.)	/popstra/company	SEM	/business/brand	0.947	/popstra/company	0.927	/business/sponsor	0.888
		SKA	/ontologies/ontology_instance	0.965	/popstra/company	0.958	/fblinux/topic	0.904
		UTR	/popstra/company	0.942	/ontologies/ontology_instance	0.900	/business/sponsor	0.876
/m/02x_h0 (Jermaine Dupri)	/music/composer	SEM	/celebrities/celebrity	0.942	/music/composer	0.919	/broadcast/artist	0.859
		SKA	/broadcast/artist	0.970	/music/composer	0.969	/internet/social_network_user	0.950
		UTR	/music/composer	0.944	/broadcast/artist	0.915	/internet/social_network_user	0.832
/m/0ptk_ (Canadian Dollar)	/finance/currency	SEM	/finance/currency	0.955	/military/military_post	0.490	/location/location	0.414
		SKA	/sports/sports_team_location	0.974	/finance/currency	0.971	/award/award_nominee	0.921
		UTR	/finance/currency	0.963	/location/location	0.536	/military/military_post	0.524

Table 4: Representative type re-ranking examples. We present top 3 type candidates and corresponding relevance scores from each of the three modules in SSET.

RQ4: Can unsupervised type re-ranking improve the entity typing performance? Comparing the results between **Exp. 5** and **Exp. 6**, we observe a remarkable performance improvement with the inclusion of the unsupervised type re-ranking module. In contrast, relying solely on either SEM or SKA module leads to suboptimal results, which suggests that both transformers and GNNs have certain limitations when it comes to modeling knowledge within a KG. Notably, despite not incurring any additional computational overhead during training, the incorporation of the re-ranking module results in a more significant performance gain compared to using textual embeddings or employing the knowledge distillation loss. This observation highlights that re-ranking is a direct, effective, and efficient knowledge integration approach.

RQ5: How do multi-hop neighbors affect the performance of SSET? From Table 3 we observe that leveraging multi-hop neighbors generally improves the performance of the proposed SSET model. Specifically, including 2-hop neighbors is sufficient for SSET to attain optimal results on the FB15kET dataset. In contrast, to achieve the best performance on the YAGO43kET dataset, SSET needs to aggregate knowledge from 3-hop neighbors. This observation can also be justified by the sparser graph structure of the YAGO43kET dataset. Furthermore, the proposed SSET model surpasses GNN-based methods RGCN (Schlichtkrull et al., 2018) and MiNer (Jin et al., 2022) with neighbor entities in different numbers of hops, demostrating that the proposed SSET model is robust to the scope of structural knowledge.

5.5 Case Study

In Table 4, we selected three representative inference results obtained from the three modules of the SSET model. These examples show how type re-ranking improves the accuracy of type inference. In the first example, the missing type */popstra/company* of entity */m/0k8z* (*Apple Inc.*) is initially considered as a negative sample, resulting in a limited relevance score during the training process. However, other candidate types, namely */business/brand* and */ontologies/ontology_instance*, fail to obtain sufficient support from both the SEM and the SKA modules. Consequently, after re-ranking, we can conclude that */popstra/company* is the most appropriate type for entity */m/0k8z*. The second example follows a similar pattern.

In the third example, the SEM module determines that entity */m/0ptk* (*Canadian Dollar*) belongs to the missing type */finance/currency* based on the textual semantics provided in the local neighborhood. [6] In contrast, the SKA module is unable to clearly differentiate between this type and other distractors. Nevertheless, type re-ranking can help the model to make desirable inferences. It further confirms the critical role of integrating textual semantics is in the KGET task.

6 Conclusion

In this paper, we propose a novel model SSET for the KGET task. SSET focuses on modeling the integration and interaction between semantic and structural knowledge from various perspectives. The SEM module utilizes structural triples and entity-type assertions as the corpus to fine-tune the PLM. The SKA module fuses pre-trained textual embeddings into learnable structural embeddings to address the "cold start" issue and leverages the preliminary inference results from the SEM module to mitigate the false-negative problem. Finally, the UTR module enables us to combine inference scores from different modules with minimal computational overhead. The experiment results show that the proposed SSET model significantly and consistently outperforms all baseline methods.

[6] To mitigate overfitting and prevent potential information leakage, we refrain from utilizing the textual semantics of the target entity in the process of type inference.

Limitations

Although our proposed method achieved a significant breakthrough on the KGET task, it still has some limitations to be addressed in the future. First, the proposed SSET model is not capable of handling "Inductive Entity Typing". The KGET task discussed in this paper, along with related studies, operates under a "transductive setting". The transductive setting focuses on inferring missing entity-type assertions with the set of types present in the KG. We plan to tackle unseen types that are not mentioned in the KG in the future. In addition, the proposed FET model cannot deal with the fine-grained entity typing (FET) task. The FET task and the KGET task differ in the objective, the former predicts the types for entity mentions within natural language sentences, while the latter focuses on entities in KGs. We also leave the FET task as a future work.

Acknowledgements

Part of the research reported in this paper was done when Ho-fung Leung was with The Chinese University of Hong Kong. The research presented in this paper was partially supported by the Research Grants Council of the Hong Kong Special Administrative Region, China (CUHK 14222922, RGC GRF 2151185).

References

Russa Biswas, Jan Portisch, Heiko Paulheim, Harald Sack, and Mehwish Alam. 2022. Entity type prediction leveraging graph walks and entity descriptions. In *The Semantic Web – ISWC 2022*, pages 392–410, Cham. Springer International Publishing.

Russa Biswas, Radina Sofronova, Harald Sack, and Mehwish Alam. 2021. Cat2type: Wikipedia category embeddings for entity typing in knowledge graphs. In *Proceedings of the 11th Knowledge Capture Conference*, K-CAP '21, page 81–88, New York, NY, USA. Association for Computing Machinery.

Kurt 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 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, page 1247–1250, New York, NY, USA. Association for Computing Machinery.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems 26*, volume 26. Curran Associates, Inc.

Shuang Chen, Jinpeng Wang, Feng Jiang, and Chin-Yew Lin. 2020. Improving entity linking by modeling latent entity type information. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):7529–7537.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Xiou Ge, Yun Cheng Wang, Bin Wang, and C.-C. Jay Kuo. 2023. Compounding geometric operations for knowledge graph completion. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6947–6965, Toronto, Canada. Association for Computational Linguistics.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models.

Zhiwei Hu, Victor Gutierrez-Basulto, Zhiliang Xiang, Ru Li, and Jeff Pan. 2022a. Transformer-based entity typing in knowledge graphs. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5988–6001, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Zhiwei Hu, Victor Gutierrez Basulto, Zhiliang Xiang, Xiaoli Li, Ru Li, and Jeff Z. Pan. 2022b. Type-aware embeddings for multi-hop reasoning over knowledge graphs. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, pages 3078–3084. International Joint Conferences on Artificial Intelligence Organization. Main Track.

Ziniu Hu, Yichong Xu, Wenhao Yu, Shuohang Wang, Ziyi Yang, Chenguang Zhu, Kai-Wei Chang, and Yizhou Sun. 2022c. Empowering language models with knowledge graph reasoning for open-domain question answering. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9562–9581, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Hailong Jin, Lei Hou, Juanzi Li, and Tiansi Dong. 2019. Fine-grained entity typing via hierarchical multi graph convolutional networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*

(EMNLP-IJCNLP), pages 4969–4978, Hong Kong, China. Association for Computational Linguistics.

Zhuoran Jin, Pengfei Cao, Yubo Chen, Kang Liu, and Jun Zhao. 2022. A good neighbor, a found treasure: Mining treasured neighbors for knowledge graph entity typing. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 480–490, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

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*.

Weijie Liu, Peng Zhou, Zhiruo Wang, Zhe Zhao, Haotang Deng, and Qi Ju. 2020. FastBERT: a self-distilling BERT with adaptive inference time. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6035–6044, Online. Association for Computational Linguistics.

Justin Lovelace, Denis Newman-Griffis, Shikhar Vashishth, Jill Fain Lehman, and Carolyn Rosé. 2021. Robust knowledge graph completion with stacked convolutions and a student re-ranking network. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1016–1029, Online. Association for Computational Linguistics.

Changsung Moon, Steve Harenberg, John Slankas, and Nagiza Samatova. 2017a. Learning contextual embeddings for knowledge graph completion. Pacific Asia Conference on Information Systems.

Changsung Moon, Paul Jones, and Nagiza F. Samatova. 2017b. Learning entity type embeddings for knowledge graph completion. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, CIKM '17, page 2215–2218, New York, NY, USA. Association for Computing Machinery.

Weiran Pan, Wei Wei, and Xian-Ling Mao. 2021. Context-aware entity typing in knowledge graphs. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2240–2250, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Michael Schlichtkrull, Thomas Kipf, Peter Bloem, Rianne Berg, Ivan Titov, and Max Welling. 2018. *Modeling Relational Data with Graph Convolutional Networks*, pages 593–607.

Zixing Song, Ziqiao Meng, Yifei Zhang, and Irwin King. 2021. Semi-supervised multi-label learning for graph-structured data. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, CIKM '21, page 1723–1733, New York, NY, USA. Association for Computing Machinery.

Zixing Song, Yifei Zhang, and Irwin King. 2022. Towards an optimal asymmetric graph structure for robust semi-supervised node classification. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '22, page 1656–1665, New York, NY, USA. Association for Computing Machinery.

Ming-Hsiang Su, Chin-Wei Lee, Chi-Lun Hsu, and Ruei-Cyuan Su. 2022. RoBERTa-based traditional Chinese medicine named entity recognition model. In *Proceedings of the 34th Conference on Computational Linguistics and Speech Processing (ROCLING 2022)*, pages 61–66, Taipei, Taiwan. The Association for Computational Linguistics and Chinese Language Processing (ACLCLP).

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 '07, page 697–706, New York, NY, USA. Association for Computing Machinery.

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

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, page 2071–2080.

Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: A free collaborative knowledgebase. *Commun. ACM*, 57(10):78–85.

Liang Wang, Wei Zhao, Zhuoyu Wei, and Jingming Liu. 2022. SimKGC: Simple contrastive knowledge graph completion with pre-trained language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4281–4294, Dublin, Ireland. Association for Computational Linguistics.

Yuming Wang, Huiqiang Zhao, Lai Tu, Jingpei Dan, and Ling Liu. 2019. A relation extraction method based on entity type embedding and recurrent piecewise residual networks. In *Big Data – BigData 2019*, pages 33–48, Cham. Springer International Publishing.

Ruobing Xie, Zhiyuan Liu, Jia Jia, Huanbo Luan, and Maosong Sun. 2016a. Representation learning of

knowledge graphs with entity descriptions. *Proceedings of the AAAI Conference on Artificial Intelligence*, 30(1).

Ruobing Xie, Zhiyuan Liu, and Maosong Sun. 2016b. Representation learning of knowledge graphs with hierarchical types. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, IJCAI'16, page 2965–2971. AAAI Press.

Yu Zhao, Anxiang Zhang, Huali Feng, Qing Li, Patrick Gallinari, and Fuji Ren. 2020. Knowledge graph entity typing via learning connecting embeddings. *Knowledge-Based Systems*, 196:105808.

Yu Zhao, Han Zhou, Anxiang Zhang, Ruobing Xie, Qing Li, and Fuzhen Zhuang. 2022. Connecting embeddings based on multiplex relational graph attention networks for knowledge graph entity typing. *IEEE Transactions on Knowledge and Data Engineering*.

K. Zhu and J. Wu. 2021. Residual attention: A simple but effective method for multi-label recognition. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 184–193, Los Alamitos, CA, USA. IEEE Computer Society.

Jianhuan Zhuo, Qiannan Zhu, Yinliang Yue, Yuhong Zhao, and Weisi Han. 2022. A neighborhood-attention fine-grained entity typing for knowledge graph completion. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, WSDM '22, page 1525–1533, New York, NY, USA. Association for Computing Machinery.

Changlong Zou, Jingmin An, and Guanyu Li. 2022. Knowledge graph entity type prediction with relational aggregation graph attention network. In *The Semantic Web*, pages 39–55, Cham. Springer International Publishing.

A Re-weighting Function in the SFNA Loss

In the SFNA Loss (Hu et al., 2022a), the reweighting function $f(\cdot)$ will assign a lower weight to the negative samples with too high or too low probabilities. The negative samples with a high type prediction probability are very likely to be false-negative samples. The negative samples with a low type prediction probability are usually easy samples. The application of the following re-weighting function (Hu et al., 2022a) allows our model to pay less attention to these negative samples.

$$f(x) = \begin{cases} 3x - 2x^2, & x \leq 0.5 \\ x - 2x^2 + 1, & x > 0.5 \end{cases}. \quad (16)$$

B The Statistics of Datasets

The statistics of the FB15kET and YAGO43kET datasets are shown in Table 5.

Dataset	FB15kET	YAGO43kET
#Entities	14,951	42,335
#Relations	1,345	37
#Types	3,851	45,182
#Triples	483,142	331,687
#Train	136,618	375,853
#Valid	15,749	42,739
#Test	15,780	42,750

Table 5: Statistics of Datasets

C Evaluation Metrics

For each test sample, we calculate the probabilities that the entity belongs to each of the types within the KG and then sort them in descending order. For a fair comparison, we adopt the filtered setting (Bordes et al., 2013) to remove all type annotations existed in training, validation, and test sets before ranking. We compute Hits@1, Hits@3, Hits@10, Mean Rank (MR), and Mean Reciprocal Rank (MRR) for performance evaluation. Higher results stand for better performance for all evaluation metrics except MR.

D Experiment Settings

Our implementation of the semantic knowledge encoder is based on the "BERT-base-uncased" and the "RoBERTa-base" PLM from the Huggingface transformer library. We use Adam optimizer (Kingma and Ba, 2015) to optimize the module parameters. The initial learning rate $1e$-4 remains unchanged within the initial decay interval of 50 epochs. After that, we half the learning rate and double the decay interval. We fine-tune the BERT (Devlin et al., 2019) and the RoBERTa (Su et al., 2022) model with a batch size of 32 for a maximum of 400 epochs. During evaluation, we set the batch size to 1 due to the GPU memory limitation. The tokenized sequences within each batch will be padded to the length of the longest sequence. We adopt similar experiment settings for the SKA module except that the learning rate $1e$-3 and $2e$-3 on the two datasets remains unchanged. For each entity, we sample 7 triples and sample 3 known types during training. All triples and known types of the target entity will be used in model inference. We select the best hyper-parameters based on the model performance on the validation set. Detailed parameter settings are listed in Table 3.

Dataset	FB15kET	YAGO43kET
SEM learning rate	$5e$-4	$5e$-4
SKA learning rate	$1e$-3	$2e$-3
Training batch size	32	32
Textual Emb. dim.	768	768
Structural Emb. dim.	100	100
# triples sampled: m	7	3
# known types sampled: n	8	3
# Heads in CSRA: H	5	5
λ	0.75	0.8
α	0.5	0.5

Table 6: The hyperparameters of SSET model in the FB15kET and YAGO43kET datasets

E Complexity Analysis

We evaluate the computational cost of the SEM and SKA modules based on the number of floating-point operations (FLOPs) required during the training or inference stage. It should be noted that the computational cost of the SKA module depends on the scope of the knowledge aggregation. The results corresponding to the two datasets are listed in Table 7.

Module	FB15kET	YAGO43kET
SEM (w/ BERT)	21.82G	23.25G
SEM (w/ RoBERTa)	25.28G	25.63G
SKA (1-hop)	327.6M	2.15G
SKA (2-hop)	1.74G	5.27G
SKA (3-hop)	10.99G	20.09G

Table 7: The number of FLOPs required by the SEM and SKA modules on different datasets with different language models.

All experiments with the BERT language model are conducted on a Lenovo workstation with an

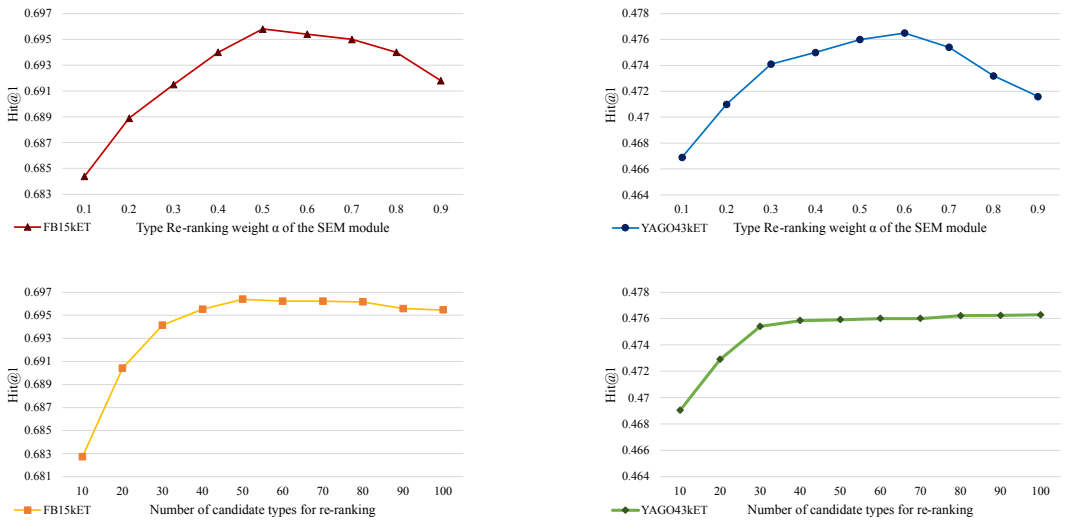

Figure 3: Experimental results of ablation studies with different experimental conditions.

Intel Xeon W-2123 processor and an NVIDIA GeForce RTX3090 GPU with 24GB memory. We evaluated SSET with RoBERTa language model on a server with an Intel Xeon Platinum 8358 processor and 8 NVIDIA A100 40G GPUs. [7] We should emphasize that such configurations are not essential for our model to generate satisfactory entity typing results. The estimated times for modules of SSET to converge (or finish) are listed in Table 8.

Module	FB15kET	YAGO43kET
SEM (w/ BERT)	7 hours	17.5 hours
SEM (w/ RoBERTa)	7 hours	19.7 hours
SKA (1-hop)	1.3 hours	6.1 hours
SKA (2-hop)	1.5 hours	7.3 hours
SKA (3-hop)	4.3 hours	21.4 hours
UTR (inference only)	< 10 seconds	4 mins

Table 8: Estimated time for different modules in SSET to achieve its best results.

The statistics above show that the main computational overhead of our model arises from language model fine-tuning. Nevertheless, compared to other language model fine-tuning tasks (Liu et al., 2020), the computational cost of the SEM module remains reasonable. Despite this, considering the evident performance improvement, we maintain the belief that incorporating textual information in KGET is both necessary and feasible. In the future, we plan to explore the integration of efficient parameter tuning techniques (e.g., LoRA (Hu et al., 2021)) in the domain of KGs. On the other hand, the computation cost of the SKA module exhibits a noticeable border effect. Expanding the scope of knowledge aggregation to more hops is unlikely to yield remarkable performance improvements, but will substantially increase the training time.

[7] Only 1 GPU is used in our experiments.

F Additional Results with Different Hyperparameter Settings.

Impact of Re-ranking Weights. We conduct additional ablation experiments to analyze the effects of using different re-ranking weights α for module balancing. A higher value of α allows our model to prioritize semantic knowledge in the text, whereas a lower value of α directs our model to focus more on the structural knowledge of the target entity. From Figure 3 (top) we can see that a moderate value of α (e.g. 0.5 on the FB15kET dataset and 0.6 on the YAGO43kET dataset) maximizes the performance of type re-ranking. This reaffirms the substantial complementary relationship between semantic and structural knowledge.

Impact of Selecting Different Numbers of Type Candidates for Re-ranking. We further investigate the impact of using different numbers of type candidates for re-ranking. From Figure 3 (bottom) we can conclude that increasing the number of type candidates generally enhances the model performance. Specifically, selecting an insufficient number of candidates excludes plausible types supported by high probabilities derived from semantic knowledge. However, employing an excessive number of re-ranking candidates may degrade the model performance as structural knowledge cannot provide sufficient support for type candidates with relatively lower probabilities.