# Transformer-based Entity Typing in Knowledge Graphs

**Zhiwei Hu**[♣] **Víctor Gutiérrez-Basulto**[♢] **Zhiliang Xiang**[♢]
**Ru Li**[♣*] **Jeff Z. Pan**[♠*]
♣ School of Computer and Information Technology, Shanxi University, China
♢ School of Computer Science and Informatics, Cardiff University, UK
♠ ILCC, School of Informatics, University of Edinburgh, UK
♣ zhiweihu@whu.edu.cn, liru@sxu.edu.cn
♢{gutierrezbasultov, xiangz6}@cardiff.ac.uk
♠j.z.pan@ed.ac.uk

## Abstract

We investigate the knowledge graph entity typing task which aims at inferring plausible entity types. In this paper, we propose a novel **T**ransformer-based **E**ntity **T**yping (TET) approach, effectively encoding the content of neighbors of an entity. More precisely, TET is composed of three different mechanisms: a ***local transformer*** allowing to infer missing types of an entity by independently encoding the information provided by each of its neighbors; a ***global transformer*** aggregating the information of all neighbors of an entity into a single long sequence to reason about more complex entity types; and a ***context transformer*** integrating neighbors content based on their contribution to the type inference through information exchange between neighbor pairs. Furthermore, TET uses information about class membership of types to semantically strengthen the representation of an entity. Experiments on two real-world datasets demonstrate the superior performance of TET compared to the state-of-the-art.

## 1 Introduction

A knowledge graph (KG) (Pan et al., 2016) is a multi-relational graph encoding factual knowledge, with the form $(h, r, t)$ where $h$, $t$ are the head and tail entities connected via the relation $r$. In this paper, we consider KGs with *minimal schema information*, i.e., those containing entity type assertions, as the only schema information, of the form $(e, has\_type, c)$ stating that the entity $e$ has type $c$; e.g., to capture that Barack Obama has type President. Entity type knowledge is widely used in NLP tasks, e.g., in relation extraction (Liu et al., 2014), entity and relation linking (Gupta et al., 2017; Pan et al., 2019), question answering (ElSahar et al., 2018; Hu et al., 2022), and fine-grained entity typing on text (Onoe et al., 2021; Qian et al., 2021; Liu et al., 2021). However, entity types are far from complete, since in real-world applications they are continuously emerging. For example, about 10% of entities in FB15k (Bordes et al., 2013) have the type */music/artist*, but do not have */people/person* (Moon et al., 2017).

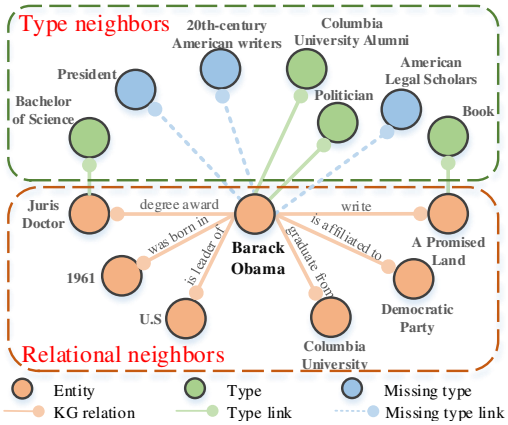


Figure 1: A KG with its entity type information.

In light of this, it has been recently investigated the *Knowledge Graph Entity Typing (KGET)* task, aiming at inferring missing entity types in a KG. Most existing approaches to KGET use methods based on either embeddings or graph convolutional networks (GCN). Despite the huge progress these methods have made, there are still some important challenges to be solved. On the one hand, most embedding-based models (Moon et al., 2017; Zhao et al., 2020; Ge et al., 2021; Zhuo et al., 2022) encode all neighbors of a target entity into a single vector, but in many cases only some neighbors are necessary to infer the correct types. For example, as shown in Figure 1, to predict that the entity *Barack Obama* has type *President*, only the neighbor $\xrightarrow{is\_leader\_of}$ *U.S* is needed. Indeed, using too many neighbors, such as $\xrightarrow{graduate\_from}$ *Columbia University*, will introduce noise. The CET model (Pan et al., 2021) overcomes this problem by encoding each neighbor independently. However, since entities and relations are repre-

*Contact Authors

sented by TransE (Bordes et al., 2013), there is a restriction on the direction of the representation of entities and relations direction, fixing it from entity to relation or vice versa. As a consequence, ***certain interactions between neighbor entities and relations are ignored***. Also, to predict more complex types, CET directly adds and averages the neighbor representations, ***weakening the contribution of different neighbors***, since it ignores that the contribution of different neighbors to different types might not be the same. For example, as shown in Figure 1, the inference of the type *20th-century American writer* involves multiple semantic aspects of *Barack Obama*, it requires to jointly consider the neighbors $\xrightarrow{write}$ *A Promised Land*, $\xrightarrow{was\_born\_in}$ *1961*, and $\xrightarrow{is\_leader\_of}$ *U.S*, but the neighbor $\xrightarrow{degree\_award}$ *Juris Doctor* should get less attention. On the other hand, GCN frameworks for KGET use expressive representations for entities and relations based on their neighbor entities and relations (Jin et al., 2019; Zhao et al., 2022; Zou et al., 2022; Vashishth et al., 2020; Pan et al., 2021). However, a common problem of GCN-based models is that they aggregate information only along the paths starting from neighbors of the target entity, ***limiting the representation of interdependence between neighbors that are not directly connected.*** For example, in Figure 1 the entities *Juris Doctor* and *U.S* are not connected, but combining their information could help to infer that *American Legal Scholars* is a type of *Barack Obama*. This could be fixed by increasing the number of layers, but with an additional computational cost.

The main objective of this paper is to introduce a transformer-based approach to KGET that addresses the highlighted challenges. The transformer architecture (Vaswani et al., 2017) has been essential for NLP, e.g., in pre-trained language models (Devlin et al., 2019; Reimers and Gurevych, 2019; Lan et al., 2020; Wu et al., 2021a), document modeling (Wu et al., 2021b), and link prediction (Wang et al., 2019; Chen et al., 2021). Transformers are well-suited for KGET as entities and relations in a KG can be regarded as tokens, and using the transformer as encoder, one can thus achieve bidirectional deep interaction between entities and relations. Specifically, we propose **TET**, a **T**ransformer-based **E**ntity **T**yping model for KGET, composed of the following three inference modules. A ***local transformer*** that independently encodes the relational and type neighbors of an entity into a sequence, facilitating bidirectional interaction between elements within the sequence, addressing the first problem. A ***global transformer*** that aggregates all neighbors of an entity into a single long sequence to simultaneously consider multiple attributes of an entity, allowing to infer more 'complex' types, thus addressing the third problem. A ***context transformer*** that aggregates neighbors of an entity in a differentiated manner according to their contribution while preserving the graph structure, thus addressing the second problem. Furthermore, we use semantic knowledge about the known types in a KG. In particular, we find out that types are normally clustered in classes. For example, the types *medicine/disease*, *medicine/symptom*, and *medicine/drug* belong to the class *medicine*. We use this class membership information for replacing the 'generic' relation *has_type* with a more fine-grained relation that captures to which class a type belongs to, enriching the semantic content of connections between entities and types. To sum up, our contributions are:

- We propose a novel transformer-based framework for inferring missing entity types in KGs, encoding knowledge about entity neighbors from three different perspectives.
- We use class membership of types to replace the single *has_type* relation with class-membership relations providing fine-grained semantic information.
- We conduct empirical and ablation experiments on two real-world datasets, demonstrating the superiority of TET over existing SoTA models.

Data, code, and an extended version with appendix are available at `https://github.com/zhiweihu1103/ET-TET`.

## 2 Related Work

The knowledge graph completion (KGC) task is usually concerned with predicting the missing head or tail entities of a triple. KGET can thus be seen as a specialization of KGC. Existing KGET methods can be classified in embedding- and GNC-based.

**Embedding-based Methods.** ETE (Moon et al., 2017) learns entity embeddings for KGs by a standard representation learning method (Bordes et al., 2013), and further builds a mechanism for information exchange between entities and their types.

ConnectE (Zhao et al., 2020) jointly embeds entities and types into two different spaces and learns a mapping from the entity space to the type space. CORE (Ge et al., 2021) utilizes the models RotatE (Sun et al., 2019) and ComplEx (Trouillon et al., 2016) to embed entities and types into two different complex spaces, and develops a regression model to link them. However, the above methods do not fully consider the known types of entities while training the entity embedding representation, which seriously affects the prediction performance of missing types. Also, the representation of types in these methods is such that they cannot be semantically differentiated. CET (Pan et al., 2021) jointly utilizes information about existing type assertions in a KG and about the neighborhood of entities by respectively employing an independent-based mechanism and an aggregated-based one. It also utilizes a pooling method to aggregate their inference results. AttEt (Zhuo et al., 2022) designs an attention mechanism to aggregate the neighborhood knowledge of an entity using type-specific weights, which are beneficial to capture specific characteristics of different types. A shortcoming of these two methods is that, unlike our TET model, they are not able to cluster types in classes, and are thus not able to semantically differentiate them in a fine-grained way.

**GCN-based Methods.** Graph Convolutional Networks (GCNs) have proven effective on modeling graph structures (Kipf and Welling, 2017; Hamilton et al., 2017; Dettmers et al., 2018). However, directly using GCNs on KGs usually leads to poor performance since KGs have different kinds of entities and relations. To address this problem, RGCN (Schlichtkrull et al., 2018) proposes to apply relation-specific transformations in GCN's aggregation. HMGCN (Jin et al., 2019) proposes a hierarchical multi-graph convolutional network to embed multiple kinds of semantic correlations between entities. CompGCN (Vashishth et al., 2020) uses composition operators from KG-embedding methods by jointly embedding both entities and relations in a relational graph. ConnectE-MRGAT (Zhao et al., 2022) proposes a multiplex relational graph attention network to learn on heterogeneous relational graphs, and then utilizes the ConnectE method for infering entity types. RACE2T (Zou et al., 2022) introduces a relational graph attention network method, utilizing the neighborhood and relation information of an entity for type inference. A common problem with these methods is that they follow a simple single-layer attention formulation, restricting the information transfer between unconnected neighbors of an entity.

**Transformer-based Methods.** To the best of our knowledge, there are no transformer-based approaches to KGET. However, two transformer-based frameworks for the KGC task have been already proposed: CoKE (Wang et al., 2019) and HittER (Chen et al., 2021). Our experiments show that they are not suitable for KGET.

## 3 Method

In this section, we describe the architecture of our TET model (cf. Figure 2). We start by introducing necessary background (Sec. 3.1), then present in detail the architecture of TET (Sec. 3.2). Finally, we describe pooling and optimization strategies (Sec. 3.3 and 3.4).

### 3.1 Background

In this paper, a knowledge graph (Pan et al., 2016) is represented in a standard format for graph-structured data such as RDF (Pan, 2009). A *knowledge graph (KG)* $\mathcal{G}$ is a tuple $(\mathcal{E}, \mathcal{R}, \mathcal{C}, \mathcal{T})$, where $\mathcal{E}$ is a set of entities, $\mathcal{C}$ is a set of entity types, $\mathcal{R}$ is a set of relation types, and $\mathcal{T}$ is a set of triples. Triples in $\mathcal{T}$ are either *relation assertions* $(h, r, t)$ where $h, t \in \mathcal{E}$ are respectively the *head* and *tail* entities of the triple, and $r \in \mathcal{R}$ is the *edge* of the triple connecting head and tail; or *entity type assertions* $(e, \mathit{has\_type}, c)$, where $e \in \mathcal{E}$, $c \in \mathcal{C}$, and *has_type* is the instance-of relation. For $e \in \mathcal{E}$, the *relational neighbors of* $e$ is the set $\{(r, f) \mid (e, r, f) \in \mathcal{T}\}$. The *type neighbors of* $e$ are defined as $\{(\mathit{has\_type}, c) \mid (e, \mathit{has\_type}, c) \in \mathcal{T}\}$. We will simply say *neighbors of* $e$ when we refer to the relational and type neighbors of $e$. The goal of this paper is to address KGET task which aims at inferring missing types from $\mathcal{C}$ in entity type assertions.

### 3.2 Model Architecture

In this section, we introduce the local, global and context transformer-based modeling components of our TET model. Before defining these components, we start by discussing an important observation.

#### 3.2.1 Class Membership

A key observation is that in a KG *all* type assertions are uniformly defined using the relation *has_type*.

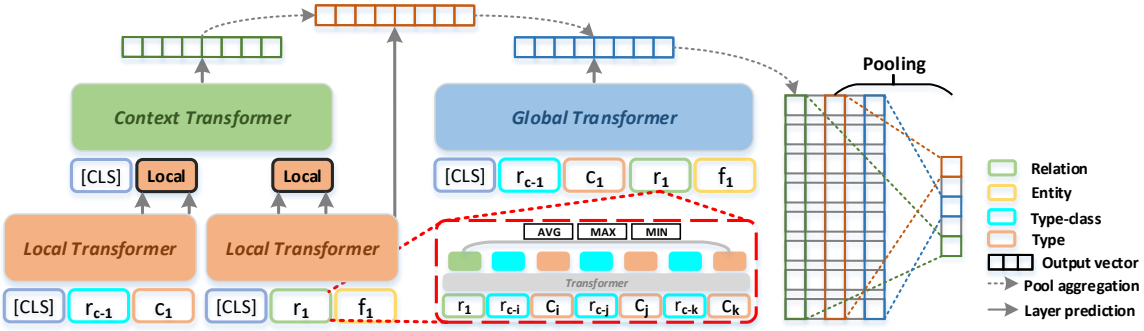


Figure 2: An overview of the TET model. The red dotted box part is only performed on the YAGO43kET dataset. Note that $r_{c-i}$ is an abbreviation of $r_{class_i}$. Box with Local text indicates the output of the local transformer module.

As a consequence, we do not have a way to fully differentiate the contribution of different types of an entity during inference, as we cannot capture the relationship between them and their relevance, weakening thus the contribution of type supervision on entities. However, in practice types are clustered together in classes (i.e., root types in a domain); e.g., the types *medicine/disease*, *medicine/symptom*, and *medicine/drug* belong to the class *medicine*. This allows us to identify that these types are related as all of them talk about something related to medicine, providing us therefore with fine-grained semantic information. With this insight in mind, for each class, we create a relation that will be used to model that a type is an element of that class. For instance, for the class *medicine*, we introduce the relation *belongs_class_medicine* . We will then replace a type neighbor $(has\_type, c)$ of an entity $e$ with $(r_{class}, c)$, where $r_{class}$ is the relation modeling class membership, i.e. belonging to the class medicine. We define the *type-class neighbors of an entity* as expected. We will use below this semantically-enriched representation in our local and global transformers.

### 3.2.2 Local Transformer

The main intuition behind the local component is that the neighbors of an entity might help to determine its types, and that the contribution of each neighbor is different. For instance, if the entity *Liverpool* has the relational neighbor (*places_lived*, *Daniel Craig*), it is plausible to infer *Liverpool* has type */location/citytown*. On the other hand, the neighbor (*sports_team*, *Liverpool F.C.*) may help to infer that it has type */sports/sports_team_location*. To encode type-class neighbors $(r_{class}, c)$, similar to the input representations of BERT (Devlin et al., 2019), we build the input sequence $H = (\texttt{[CLS]}, r_{class}, c)$, where `[CLS]` is a special token, and for each element $h_i$ in $H$, we construct its input vector representation $\mathbf{h}_i$ as:

$$\mathbf{h}_i = \mathbf{h}_i^{word} + \mathbf{h}_i^{pos}$$

$\mathbf{h}_i^{word}$ and $\mathbf{h}_i^{pos}$ are randomly initialized word and position embeddings of $r_{class}$ or $c$. We apply a local transformer to each type-class neighbor sequence to model the interaction between the class relations and types of an entity. The output embedding corresponding to `[CLS]`, denoted as $\mathbf{H}^{cls} \in \mathbb{R}^{d\times 1}$, is then used to infer missing types of the target entity, where $d$ represents the dimension of the embedding. For an entity with $n$ type-class neighbors, they are denoted as $[\mathbf{H}_1^{cls}, \mathbf{H}_2^{cls}, ..., \mathbf{H}_n^{cls}]$ after the local transformer representation.

Type-class neighbors are not capable to fully capture the structural information within the KG. To alleviate this problem, we also consider relational neighbors. As for type-class neighbors, to encode the relational neighbors $(r, f)$ of an entity, we build a sequence $Q = (\texttt{[CLS]}, r, f)$, and aggregate the word and position embeddings and further apply a local transformer. The output embedding of `[CLS]` is denoted as $\mathbf{Q}^{cls} \in \mathbb{R}^{d\times 1}$, and for an entity with $m$ relational neighbors, they are represented as $[\mathbf{Q}_1^{cls}, \mathbf{Q}_2^{cls}, ..., \mathbf{Q}_m^{cls}]$ after the local transformer representation.

The local transformer mainly pays attention to a single existing neighbor at a time in the inference process, reducing the interference between unrelated types. We perform a non-linear activation on neighbors, and then perform a linear layer operation to unify the dimension to the number of types, the final local transformer score $\mathbf{S}^{loc} \in \mathbb{R}^{L\times(m+n)}$ is defined as:

$$\mathbf{W} Relu([\mathbf{H}_1^{cls}, \ldots, \mathbf{H}_n^{cls}, \mathbf{Q}_1^{cls}, \ldots, \mathbf{Q}_m^{cls}]) + b \quad (1)$$

$\mathbf{W} \in \mathbb{R}^{L \times d}$ and $b \in \mathbb{R}^{L}$ are the learnable parameters, where $L$ is the number of types. $[,]$ denotes the concatenation function, $\mathbf{H}_i^{cls} \in \mathbb{R}^{d \times 1}$ and $\mathbf{Q}_j^{cls} \in \mathbb{R}^{d \times 1}$ respectively represent the *i-th* and *j-th* embedding of the type-class and relational neighbors after the transformer representation.

An important observation is that the number of relations available vary from one KG to another. For instance, the YAGO43kET KG has substantially fewer relations than the FB15kET KG (cf. the dataset statistics in the Experiments Section), making the discrimination among relations in relational triples harder. To tackle this problem, for the YAGO43kET KG, we semantically enrich the representation of relations by using the type-class membership information. Specifically, for a relational neighbor $(r, f)$ of an entity, we use the types of $f$ belonging to a certain class to enhance the relation $r$ in the sequence $(\texttt{[CLS]}, r, f)$ using the following steps:

1. Let $\Gamma = \{(\textit{has\_type}, c_1), (\textit{has\_type}, c_2), \ldots, (\textit{has\_type}, c_\ell)\}$ be the set of all type neighbors of $f$. We replace $\Gamma$ with the set $\Gamma'$ of corresponding type-class neighbors: $\{(r_{\textit{class}_1}, c_1), (r_{\textit{class}_2}, c_2), \ldots, ((r_{\textit{class}_\ell}, c_\ell)\}$, i.e., representing that $c_i$ is a member of $\textit{class}_i$.

2. Based on $r$ and $\Gamma'$, we construct a sequence $P = (r, r_{\textit{class}_1}, c_1, r_{\textit{class}_2}, c_2, \ldots, r_{\textit{class}_\ell}, c_\ell)$. For each element $p_i$ of $P$, we assign randomly initialized word and position embeddings to capture sequence order. We then apply a transformer to capture the interaction between tokens. The output token embeddings are denoted as $[\mathbf{p}_0, \mathbf{p}_1, \ldots, \mathbf{p}_\ell]$.

3. For the output token embeddings, we use three different operations to obtain the final representation of relation $r$: average, maximum, and minimum. For the YAGO43kET KG, we replace the word embedding $r$ in sequence $Q$ with $\mathbf{P}_{avg} = \sum_{i=0}^{\ell} \mathbf{p}_i$, $\mathbf{P}_{max} = \textit{Max}(\mathbf{p}_i)$, or $\mathbf{P}_{min} = \textit{Min}(\mathbf{p}_i)$.

#### 3.2.3 Global Transformer

The local transformer mechanism is suitable for types that can be inferred by looking at simple structures, and for which independently considering neighbors is thus enough. However, inferring 'complex' types requires to capture the interaction between different neighbors of an entity. For instance, if we would like to infer that the entity *Birmingham_City_L.F.C.* has type *Women's_football_clubs_in_England*, we need to simultaneously consider different sources of information to support this, such as the type neighbor (*has_type*, *Association_football_clubs*) and relational neighbor (*isLocatedIn*, *England*) of *Birmingham_City_L.F.C.*, and that (*playsFor*, *Birmingham_City_L.F.C.*) and (*hasGender*, *female*) are relational neighbors of the entity *Darla_Hood*. To this aim, we introduce a global transformer module capturing the interaction between type-class and relational neighbors by comprehensively representing them as the input of a transformer as follows:

1. For a target entity $e$, we define the set $\Gamma'$ as done in Section 3.2.2. Further, let $\Xi = \{(r_1, f_1), \ldots, (r_m, f_m)\}$ denote the set of all relational neighbors of $e$.

2. We uniformly represent $\Gamma'$ and $\Xi$ as a single sequence $G = (\texttt{[CLS]}\ r_{\textit{class}_1}, c_1, r_{\textit{class}_2}, c_2, \ldots, r_{\textit{class}_\ell}, c_n, r_1, f_1, r_2, f_2, \ldots r_m, f_m)$.

3. For each element in the sequence $G$, we assign randomly initialized word and position embeddings, and input it into a transformer. The output embedding of `[CLS]` is denoted $\mathbf{G}^{cls} \in \mathbb{R}^{d \times 1}$. Similar to Equation (1), we define the prediction score $\mathbf{S}^{glo} \in \mathbb{R}^{L \times 1}$ as $\mathbf{W}\mathit{Relu}([\mathbf{G}^{cls}]) + b$.

#### 3.2.4 Context Transformer

For complex types, the global transformer uniformly serializes the information about the neighbors of the target entity. However, the neighbors of the target entity are pairs, and this structural information might be useful for inference. For instance, to infer that the entity *Barack Obama* has type *20th-century American writers*, we need to consider different aspects of its relational neighbors, e.g., the neighbor (*bornIn*, *Chicago*) focuses on the birthplace, while the neighbor (*write*, *A Promised Land*) is concerned with possible careers. The global transformer serialization of pairs as a sequence may lead to two problems: First, serializing neighbors disregards the structure of the graph. Second, the importance of each element in the sequence is the same, and even elements that are not relevant for the inference will exchange information, e.g., *bornIn* and *A Promised Land* in the example above. To realize a differentiated aggregation between different neighbor pairs while preserving the graph structure, we use a context

transformer module as in (Chen et al., 2021). Intuitively, given the output of the local transformer and the `[CLS]` embedding, the context transformer contextualizes the target entity with type-class and relational neighbors knowledge from its neighborhood graph, details of the context transformer can be found in (Chen et al., 2021). The output embedding of `[CLS]`, denoted as $\mathbf{C}^{cls} \in \mathbb{R}^{d\times 1}$, is used for the final entity type prediction, which is defined as $\mathbf{S}^{ctx} = \mathbf{W}Relu([\mathbf{C}^{cls}]) + b$, where $\mathbf{S}^{ctx} \in \mathbb{R}^{L\times 1}$.

### 3.3 Pooling

For an entity $e$, the local, global, and context transformers may generate multiple entity typing inference results. To address this, we adopt an exponentially weighted pooling method to aggregate prediction results (Pan et al., 2021; Stergiou et al., 2021), formulated as follows:

$$\mathbf{S}_e = pool(\{\mathbf{S}_0^{loc}, \mathbf{S}_1^{loc}, ..., \mathbf{S}_{m+n-1}^{loc}, \mathbf{S}^{glo}, \mathbf{S}^{ctx}\})$$

$\mathbf{S}_e \in \mathbb{R}^L$ represents the relevance score between $e$ and its types, and $n$ ($m$) is the number of type-class (relational) neighbors of $e$ respectively. For simplicity, we will omit the identifiers (*loc,glo,ctx*). We unify the numerical order of the output results of the local, global, and context transformers as follows:

$$\begin{aligned}\mathbf{S}_e &= pool(\{\mathbf{S}_0, \mathbf{S}_1, ..., \mathbf{S}_{m+n-1}, \mathbf{S}_{m+n}, \mathbf{S}_{m+n+1}\}) \\ &= \sum_{i=0}^{m+n+1} w_i \mathbf{S}_i,\ w_i = \frac{\exp \alpha \mathbf{S}_i}{\sum_{j=0}^{m+n+1} \exp \alpha \mathbf{S}_j} \mathbf{S}_i\end{aligned}$$

We further apply a sigmoid function to $\mathbf{S}_e$, denoted as $\mathbf{s}_e = \sigma(\mathbf{S}_e)$, to map the scores between 0 and 1, where the higher the value of $\mathbf{s}_{e,k}$ of $\mathbf{s}_e$, the more likely is $e$ to have type $k$.

### 3.4 Optimization Strategy

To train a model with positive sample score $\mathbf{s}_{e,k}$ (representing that $(e, \textit{has\_type}, k)$ exists in a KG) and negative sample score $\mathbf{s}'_{e,k}$ (representing that $(e, \textit{has\_type}, k)$ does not exist in KG), usually binary cross-entropy (BCE) is used as the loss function. However, there may exist a serious false negative problem, i.e., some $(e, \textit{has\_type}, k)$ are valid, but they are missing in existing KGs. To overcome this problem, false-negative aware loss functions (FNA) have been proposed (Pan et al., 2021). Basically, they assign lower weight to negative samples with too high or too low relevance scores. We introduce a steeper false-negative aware (SFNA) loss function which gives more penalties to negative samples with too high or too low relevance scores. The negative sample score is defined as:

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

For the positive score $\mathbf{s}_{e,k}$ and negative score $\mathbf{s}'_{e,k}$, the SFNA loss is defined as follows:

$$\mathcal{L} = -\sum f(\mathbf{s}'_{e,k}) \log(1 - \mathbf{s}'_{e,k}) - \sum \log(\mathbf{s}_{e,k})$$

## 4 Experiments

In this section, we discuss the evaluation of TET relative to twelve baselines on a wide array of entity typing benchmarks. We first describe datasets and baseline models (Sec. 4.1). Then we discuss the experimental results (Sec. 4.2). Finally, we present ablation study experiments (Sec. 4.3).

### 4.1 Datasets and Baselines

**Datasets.** We evaluate our proposed TET model on two real-world knowledge graphs: FB15k (Bordes et al., 2013) and YAGO43k (Moon et al., 2017) which are the subgraphs of Freebase (Bollacker et al., 2008) and YAGO (Suchanek et al., 2007), respectively. FB15kET and YAGO43kET provide entity type instances which map entities from FB15k and YAGO43k to corresponding entity types. For fairness of the experimental comparison, we followed the standard train/test split as in the baselines. The basic statistics of all datasets are shown in Table 2.

**Baselines.** We compare TET with twelve state-of-the-art entity typing methods, and their variants. We consider the embedding-based models ETE (Moon et al., 2017), ConnectE (Zhao et al., 2020), CORE (Ge et al., 2021), AttEt (Zhuo et al., 2022) and CET (Pan et al., 2021). We consider the GCN-based models HMGCN (Jin et al., 2019), RACE2T (Zou et al., 2022), ConnectE-MRGAT (Zhao et al., 2022), CompGCN (Vashishth et al., 2020) and RGCN (Pan et al., 2021). We also use as baselines two transformer-based methods for KGC, CoKE and HittER (Wang et al., 2019; Chen et al., 2021). It should be noted that in all reported experimental results, the **bold** numbers denote the *best results* while the underlined ones the *second best*.

### 4.2 Experimental Results

Table 1 presents the evaluation results of entity type prediction on FB15kET and YAGO43kET. We

| Datasets | FB15kET | | | | YAGO43kET | | | |
|---|---|---|---|---|---|---|---|---|
| **Metrics** | **MRR** | **Hit@1** | **Hit@3** | **Hit@10** | **MRR** | **Hit@1** | **Hit@3** | **Hit@10** |
| *Embedding-based methods* | | | | | | | | |
| ETE (Moon et al., 2017)♢ | 0.500 | 0.385 | 0.553 | 0.719 | 0.230 | 0.137 | 0.263 | 0.422 |
| ConnectE (Zhao et al., 2020)♢ | 0.590 | 0.496 | 0.643 | 0.799 | 0.280 | 0.160 | 0.309 | 0.479 |
| CORE-RotatE (Ge et al., 2021)♢ | 0.600 | 0.493 | 0.653 | 0.811 | 0.320 | 0.230 | 0.366 | 0.510 |
| CORE-ComplEx (Ge et al., 2021)♢ | 0.600 | 0.489 | 0.663 | 0.816 | 0.350 | 0.242 | 0.392 | 0.550 |
| AttEt (Zhuo et al., 2022)♢ | 0.620 | 0.517 | 0.677 | 0.821 | 0.350 | 0.244 | 0.413 | 0.565 |
| CET-BCE (Pan et al., 2021)♢ | 0.682 | 0.593 | 0.733 | 0.852 | 0.472 | 0.362 | 0.540 | 0.669 |
| CET-FNA (Pan et al., 2021)♢ | 0.697 | 0.613 | 0.745 | 0.856 | 0.503 | 0.398 | 0.567 | **0.696** |
| *GCN-based methods* | | | | | | | | |
| HMGCN (Jin et al., 2019)♦ | 0.510 | 0.390 | 0.548 | 0.724 | 0.250 | 0.142 | 0.273 | 0.437 |
| ConnectE-MRGAT (Zhao et al., 2022)♢ | 0.630 | 0.562 | 0.662 | 0.804 | 0.320 | 0.243 | 0.343 | 0.482 |
| RACE2T (Zou et al., 2022)♢ | 0.640 | 0.561 | 0.689 | 0.817 | 0.340 | 0.248 | 0.376 | 0.523 |
| CompGCN-BCE (Vashishth et al., 2020)♦ | 0.657 | 0.568 | 0.704 | 0.833 | 0.357 | 0.274 | 0.384 | 0.520 |
| CompGCN-FNA (Vashishth et al., 2020)♦ | 0.665 | 0.578 | 0.712 | 0.839 | 0.355 | 0.274 | 0.383 | 0.513 |
| RGCN-BCE (Pan et al., 2021)♢ | 0.662 | 0.571 | 0.711 | 0.836 | 0.357 | 0.266 | 0.392 | 0.533 |
| RGCN-FNA (Pan et al., 2021)♢ | 0.679 | 0.597 | 0.722 | 0.843 | 0.372 | 0.281 | 0.409 | 0.549 |
| *Transformer-based methods* | | | | | | | | |
| CoKE (Wang et al., 2019)♦ | 0.465 | 0.379 | 0.510 | 0.624 | 0.344 | 0.244 | 0.387 | 0.542 |
| HittER (Chen et al., 2021)♦ | 0.422 | 0.333 | 0.466 | 0.588 | 0.240 | 0.163 | 0.259 | 0.390 |
| TET-BCE | 0.699 | 0.615 | 0.748 | 0.862 | 0.492 | 0.385 | 0.554 | 0.684 |
| TET-FNA | 0.701 | 0.608 | 0.761 | **0.873** | 0.508 | 0.405 | 0.567 | **0.696** |
| TET-SFNA-no-class | 0.706 | 0.626 | 0.749 | 0.862 | 0.472 | 0.375 | 0.525 | 0.654 |
| TET-SFNA | **0.717** | **0.638** | **0.762** | 0.872 | **0.510** | **0.408** | **0.571** | 0.695 |

Table 1: Evaluation of different models on FB15kET and YAGO43kET. ♢ results are from the original papers. ♦ results are from our implementation of the corresponding models. TET-SFNA-no-class means that type-class neighbors were not used, and for YAGO43kET in addition no semantic enhancement on relations is used.

| Datasets | FB15kET | YAGO43kET |
|---|---|---|
| # Entities | 14,951 | 42,335 |
| # Relations | 1,345 | 37 |
| # Types | 3,584 | 45,182 |
| # Clusters | 1,081 | 6,583 |
| # Train.triples | 483,142 | 331,686 |
| # Train.tuples | 136,618 | 375,853 |
| # Valid | 15,848 | 43,111 |
| # Test | 15,847 | 43,119 |

Table 2: Statistics of Datasets.

can observe that our model TET outperforms all baselines in terms of basically all metrics. These results demonstrate that transformers more effectively encode the neighbor information of an entity. Specifically, when using the BCE and FNA loss functions, TET meets or exceeds the CET model (the best performing baseline). By using the SFNA loss function, we can get further performance improvement, especially in the MRR and Hit@1 metrics on FB15kET. Furthermore, TET has different gains compared to CET with respect to the Hit metrics. The improvement on Hit@1 is higher than on Hit@3 and Hit@10 because by using three different transformer modules TET can encode the neighborhood information of an entity at three different levels of granularity. Further, if we do not use type-class neighbors and for the YAGO43kET dataset the type-class enrichment on relations is not present (TET-SNFA-no-class), we note that the performance of TET on the YAGO43kET dataset decreases considerably. Intuitively, the decrease on the YAGO43kE is larger than on FB15k because the graph structure of YAGO43k is sparser, has fewer relations, and a large number of types, making the semantic type-class knowledge crucial.

### 4.3 Ablation Studies

To verify the impact of each TET model component on the performance, we conduct ablation studies on FB15kET and YAGO43kET. In particular we look at the effect of: a) different transformer modules, Table 3; b) different neighbor content, Table 4; c) different integration methods on YAGO43kET, Table 5; d) different dropping rates, Table 6; e) the number of hops, Table 7.

**Effect of Transformer.** The local transformer by itself performs better than the global one by itself. This indicates that considering independently the neighbors of an entity can reduce interference be-

| Models | | | FB15kET | | | | YAGO43kET | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Global** | **Local** | **Context** | **MRR** | **Hit@1** | **Hit@3** | **Hit@10** | **MRR** | **Hit@1** | **Hit@3** | **Hit@10** |
| √ | √ | √ | **0.717** | **0.638** | **0.762** | **0.872** | **0.510** | **0.408** | **0.571** | 0.695 |
| √ | √ | | 0.713 | 0.632 | 0.759 | 0.871 | 0.503 | 0.401 | 0.561 | 0.690 |
| √ | | √ | 0.664 | 0.578 | 0.711 | 0.829 | 0.369 | 0.289 | 0.397 | 0.524 |
| | √ | √ | 0.700 | 0.614 | 0.752 | 0.864 | 0.509 | 0.407 | 0.568 | **0.697** |
| √ | | | 0.660 | 0.578 | 0.702 | 0.824 | 0.365 | 0.286 | 0.392 | 0.517 |
| | √ | | 0.684 | 0.596 | 0.732 | 0.859 | 0.494 | 0.387 | 0.555 | 0.690 |
| | | √ | 0.641 | 0.554 | 0.686 | 0.817 | 0.353 | 0.280 | 0.375 | 0.493 |

Table 3: Evaluation of ablation study with different transformer modules combinations.

tween unrelated types. By combining the global and context transformer, more complex types can be inferred from the token and graph structure level, achieving state-of-the-art results. Note that both the global and context transformers deal with complex types, but the context one further takes into account the relevance of different neighbors while preservin the structure of the KG. As one can see from the results, for the used datasets, the global transformer is already doing most of the work, i.e., the combination of local and global transformers achieves almost the same result as when the context one is also incorporated. We believe that in datasets with a more complex structure the context transformer could play a more prominent role, we leave this line of research as future work.

**Effect of Neighbor Content.** We observe that the impact of relational neighbors is greater than that of type-class neighbors. Indeed, removing relational neighbors leads to a substantial performance degradation in YAGO43kET. When both of them are available, type-class neighbors might help relational ones to distinguish between relevant and irrelevant types for an inference.

| | | FB15kET | | | |
|---|---|---|---|---|---|
| **relational** | **type-class** | **MRR** | **Hit@1** | **Hit@3** | **Hit@10** |
| √ | √ | **0.717** | **0.638** | **0.762** | **0.872** |
| √ | | 0.657 | 0.568 | 0.707 | 0.833 |
| | √ | 0.654 | 0.561 | 0.705 | 0.839 |
| | | YAGO43kET | | | |
| √ | √ | **0.510** | **0.408** | **0.571** | **0.695** |
| √ | | 0.467 | 0.372 | 0.518 | 0.642 |
| | √ | 0.373 | 0.288 | 0.405 | 0.535 |

Table 4: Evaluation of ablation study with different neighbor content.

**Effect of Integration Methods.** YAGO43kET has a sparser graph structure, fewer types of relations and a large number of types. To tackle this problem, in Section 3.2.2, we have enriched the representations of relations in relational neighbors with type-class knowledge. One can observe that the Avg operation outperforms Min and Max because the latter tend to discard useful content.

| Models | | | | YAGO43kET | | | |
|---|---|---|---|---|---|---|---|
| **No** | **Avg** | **Max** | **Min** | **MRR** | **Hit@1** | **Hit@3** | **Hit@10** |
| √ | | | | 0.491 | 0.385 | 0.554 | 0.684 |
| | √ | | | **0.510** | **0.408** | **0.571** | **0.695** |
| | | √ | | 0.505 | 0.404 | 0.564 | 0.688 |
| | | | √ | 0.505 | 0.405 | 0.563 | 0.691 |

Table 5: Evaluation of ablation study with different integration methods. Note that, "No" means without performing type-class semantic enhancement on the relations.

| **Dropping Rates** | **75%** | | | **90%** | | |
|---|---|---|---|---|---|---|
| **Models** | **MRR** | **Hit@1** | **Hit@3** | **MRR** | **Hit@1** | **Hit@3** |
| CompGCN | 0.648 | 0.559 | 0.697 | 0.633 | 0.544 | 0.679 |
| RGCN | 0.648 | 0.560 | 0.694 | 0.626 | 0.534 | 0.673 |
| CET | 0.670 | 0.580 | 0.721 | 0.646 | 0.553 | 0.698 |
| TET_RSE | 0.683 | 0.599 | **0.733** | 0.645 | 0.555 | 0.692 |
| TET | **0.689** | **0.606** | **0.733** | **0.658** | **0.574** | **0.701** |

Table 6: Evaluation with different dropping rates on FB15kET. TET_RSE represents TET with semantic enhancement on relations.

**Effect of Dropping Rates.** In real life KGs, many entities have sparse relations with other entities. In particular, they have few relational neighbors but a large number of types, so for their inference we lack structural relational information. Indeed, in YAGO43kET about 4.73% of its entities have five times more types than relational neighbors (Zhuo et al., 2022). To further test the robustness of TET under relation-sparsity, we also conduct ablation experiments on FB15kET by ran-

domly removing 25%, 50%, 75%, and 90% of the relational neighbors of entities. We find that with the continuous increase of the sparsity ratio, the performance of baselines decrease to varying degrees, but TET still achieves the best results under all sparsity conditions. We also consider TET with semantic enhancement on relations since by randomly dropping neighbors the number of relations might also be reduced. However, not enough relations are removed to have a positive effect. Another reason for not having positive effect is that the number of types in FB15kET is substantially smaller than in YAGO43kET. Table 6 shows results for 75%, and 90%, for missing results see appendix.

| Models | | | FB15kET | | | |
|---|---|---|---|---|---|---|
| 1-hop | 2-hops | 3-hops | MRR | Hit@1 | Hit@3 | Hit@10 |
| √ | | | **0.717** | **0.638** | **0.762** | **0.872** |
| | √ | | 0.677 | 0.592 | 0.720 | 0.844 |
| | | √ | 0.682 | 0.597 | 0.728 | 0.850 |
| √ | √ | | 0.709 | 0.626 | 0.759 | 0.869 |
| √ | | √ | 0.710 | 0.630 | 0.756 | 0.868 |
| | √ | √ | 0.680 | 0.598 | 0.721 | 0.845 |
| √ | √ | √ | 0.709 | 0.630 | 0.754 | 0.865 |

Table 7: Evaluation of ablation study with different number of hops on FB15kET.

**Effect of Number of Hops.** For relational neighbors, TET only considers one-hop information i.e., only the information around their direct neighbors. We also conduct an ablation study on the effect of using different number of hops. In principle multi-hop information could provide richer structural knowledge, increasing the discrimination of relational neighbors. Indeed, a positive effect of multi-hop information has been witnessed in several approaches to KGC. However, our experimental results show that the noise introduced by intermediate entities is more dominant than the additional knowledge $n$-hop entities and relations provide. Intuitively, for KGC multi-hop information makes a difference as it exploits the topological structure of the KG (i.e. how entities are related). However, in the input KG, types are not related between them and as our experiments show, one can not lift the topological structure at the entity-level to the type one, explaining why there is no gain from considering multi-hop information. It would interesting to confirm this observation by using GCNs, which more naturally capture multi-hop information.

## 5 Conclusions

In this paper, we propose a novel transformer-based model for KGET which utilizes contextual information of entities to infer missing types for KGs with minimal schema information. TET has three modules allowing to encode local and global neighborhood information from different perspectives. We also enhance the representation of entities by using class membership knowledge of types. We experimentally showed the benefits of our model.

## 6 Limitations

Our TET model currently suffers from two limitations. From the methodological viewpoint, a transformer mechanism introduces more parameters than embedding-based methods, bringing some computational burden and memory overhead, but they are tolerable. Also, there exist other important tasks related to types, e.g. fine-grained entity typing, aiming at classifying entity mentions into fine-grained semantic labels. TET is currently not appropriate for this kind of tasks.

## Acknowledgments

This work has been supported by the National Natural Science Foundation of China (No.61936012), by the National Key Research and Development Program of China (No.2020AAA0106100), by the National Natural Science Foundation of China (No. 62076155), by a Leverhulme Trust Research Project Grant (RPG-2021-140), and by the Chang Jiang Scholars Program (J2019032).

## References

Kurt D. Bollacker, Colin Evans, Praveen K. 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*, pages 1247–1250. ACM.

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*, pages 2787–2795.

Sanxing Chen, Xiaodong Liu, Jianfeng Gao, Jian Jiao, Ruofei Zhang, and Yangfeng Ji. 2021. Hitter: Hierarchical transformers for knowledge graph embeddings.

In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 10395–10407. Association for Computational Linguistics.

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*, pages 1811–1818. AAAI Press.

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)*, pages 4171–4186. Association for Computational Linguistics.

Hady ElSahar, Christophe Gravier, and Frédérique Laforest. 2018. Zero-shot question generation from knowledge graphs for unseen predicates and entity types. 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)*, pages 218–228. Association for Computational Linguistics.

Xiou Ge, Yuncheng Wang, Bin Wang, and C.-C. Jay Kuo. 2021. CORE: A knowledge graph entity type prediction method via complex space regression and embedding. *CoRR*, abs/2112.10067.

Nitish Gupta, Sameer Singh, and Dan Roth. 2017. Entity linking via joint encoding of types, descriptions, and context. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 2681–2690. Association for Computational Linguistics.

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, December 4-9, 2017, Long Beach, CA, USA*, pages 1024–1034.

Zhiwei Hu, Víctor Gutiérrez-Basulto, Zhiliang Xiang, Xiaoli Li, Ru Li, and Jeff Z. Pan. 2022. Type-aware embeddings for multi-hop reasoning over knowledge graphs. *CoRR*, abs/2205.00782.

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 2019, Hong Kong, China, November 3-7, 2019*, pages 4968–4977. 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*. OpenReview.net.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A lite BERT for self-supervised learning of language representations. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Qing Liu, Hongyu Lin, Xinyan Xiao, Xianpei Han, Le Sun, and Hua Wu. 2021. Fine-grained entity typing via label reasoning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 4611–4622. Association for Computational Linguistics.

Yang Liu, Kang Liu, Liheng Xu, and Jun Zhao. 2014. Exploring fine-grained entity type constraints for distantly supervised relation extraction. In *COLING 2014, 25th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, August 23-29, 2014, Dublin, Ireland*, pages 2107–2116. Association for Computational Linguistics.

Changsung Moon, Paul Jones, and Nagiza F. Samatova. 2017. Learning entity type embeddings for knowledge graph completion. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, CIKM 2017, Singapore, November 06 - 10, 2017*, pages 2215–2218. ACM.

Yasumasa Onoe, Michael Boratko, Andrew McCallum, and Greg Durrett. 2021. Modeling fine-grained entity types with box embeddings. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 2051–2064. Association for Computational Linguistics.

Jeff Z. Pan. 2009. Resource description framework. In *Handbook on Ontologies*, pages 71–90. Springer.

Jeff Z. Pan, Mei Zhang, Kuldeep Singh, Frank Van Harmelen, Jinguang Gu, and Zhi Zhang. 2019. Entity Enabled Relation Linking. In *Proc. of 18th International Semantic Web Conference (ISWC 2019)*, pages 523–538.

J.Z. Pan, G. Vetere, J.M. Gomez-Perez, and H. Wu. 2016. *Exploiting Linked Data and Knowledge Graphs for Large Organisations*. Springer.

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, Virtual Event / Punta Cana, Dominican Republic, 16-20 November, 2021*, pages 2240–2250. Association for Computational Linguistics.

Jing Qian, Yibin Liu, Lemao Liu, Yangming Li, Haiyun Jiang, Haisong Zhang, and Shuming Shi. 2021. Fine-grained entity typing without knowledge base. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 5309–5319. Association for Computational Linguistics.

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*, pages 3980–3990. Association for Computational Linguistics.

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*, volume 10843 of *Lecture Notes in Computer Science*, pages 593–607. Springer.

Alexandros Stergiou, Ronald Poppe, and Grigorios Kalliatakis. 2021. Refining activation downsampling with softpool. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pages 10337–10346. IEEE.

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*, pages 697–706. ACM.

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.

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*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 2071–2080. JMLR.org.

Shikhar Vashishth, Soumya Sanyal, Vikram Nitin, and Partha P. Talukdar. 2020. Composition-based multi-relational graph convolutional networks. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

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, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Quan Wang, Pingping Huang, Haifeng Wang, Songtai Dai, Wenbin Jiang, Jing Liu, Yajuan Lyu, Yong Zhu, and Hua Wu. 2019. Coke: Contextualized knowledge graph embedding. *CoRR*, abs/1911.02168.

Chuhan Wu, Fangzhao Wu, and Yongfeng Huang. 2021a. Da-transformer: Distance-aware transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 2059–2068. Association for Computational Linguistics.

Chuhan Wu, Fangzhao Wu, Tao Qi, and Yongfeng Huang. 2021b. Hi-transformer: Hierarchical interactive transformer for efficient and effective long document modeling. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 2: Short Papers), Virtual Event, August 1-6, 2021*, pages 848–853. Association for Computational Linguistics.

Yu Zhao, Anxiang Zhang, Ruobing Xie, Kang Liu, and Xiaojie Wang. 2020. Connecting embeddings for knowledge graph entity typing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 6419–6428. Association for Computational Linguistics.

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*.

Jianhuan Zhuo, Qiannan Zhu, Yinliang Yue, Yuhong Zhao, and Weisi Han. 2022. A neighborhood-attention fine-grained entity typing for knowledge graph completion. In *WSDM '22: The Fifteenth ACM International Conference on Web Search and Data Mining, Virtual Event / Tempe, AZ, USA, February 21 - 25, 2022*, pages 1525–1533. ACM.

Changlong Zou, Jingmin An, and Guanyu Li. 2022. Knowledge graph entity type prediction with relational aggregation graph attention network. In *The Semantic Web - 19th International Conference, ESWC 2022, Hersonissos, Crete, Greece, May 29 - June 2, 2022, Proceedings*, volume 13261 of *Lecture Notes in Computer Science*, pages 39–55. Springer.

# Appendix

## A Details about Experiments

In this section, we give more experimental details and discuss the evaluation protocol.

| Parameter | {FB15kET, YAGO43kET} |
|---|---|
| # Embedding dim | {100, 100} |
| # Train or valid batch size | {128, 128} |
| # Test batch size | {1, 1} |
| # Learning rate | {0.001, 0.001} |
| # Trm layers | {3, 3} |
| # Trm hidden dim | {480, 480} |
| # Trm heads | {4, 4} |
| # Trm dropout rate | {0.2, 0.2} |
| # Type neighbor sample size | {3, 3} |
| # KG neighbor sample size | {7, 8} |
| # Warmup epochs | {50, 50} |
| # Valid epochs | {25, 25} |
| # $\alpha$ | {0.5, 0.5} |
| # Epochs | {500, 500} |

Table 8: The main hyperparameters of TET model in different datasets.

**Experimental Setting.** Table 8 shows the hyperparameter settings of the TET model on the FB15kET and YAGO43kET datasets. We use Adam (Kingma and Ba, 2015) as the optimizer, the hyperparameters are tuned according to the MRR on the validation set. We only sample the entity type-class and relational neighbors during training and validation, but for testing we use all the neighbors of the entity for prediction, so the test batch size is set to 1. We adopted the warmup training strategy, keeping the initial learning rate 0.001 unchanged for the first 50 iterations, but after that we divided the learning rate by 5, and set the interval to be the current interval multiplied by 2. In Table 8 "Trm" refers to the three transformer modules, we use the same number of layers, hidden dim, and heads.

**Evaluation Protocol.** For each test sample $(e, c)$, we first calculate the correlation score $\mathbf{s}_e$ between the entity $e$ and type $c$, and then sort them in descending order. We report various metrics for evaluation, specifically, we adopt the filtered setting (Bordes et al., 2013; Pan et al., 2021) for computing mean reciprocal rank (MRR), and the proportion of correct entities ranked in the top 1/3/10 (Hit@1, Hit@3, Hit@10).

## B Additional Results

In this section, we discuss more ablation experimental results that are not included in the main part of the paper.

**Effect of Dropping Rates: Number of Relational Neighbors.** In the main body of the paper we discussed why we test the robustness of TET under relation-sparsity on FB15kET by randomly removing 25%, 50%, 75%, and 90% of the relational neighbors of entities. Due to space constraints we only presented the results for the two more extreme cases: 75%, and 90%. Table 9 shows the missing results for 25% and 50%.

| Dropping Rates | 25% | | | 50% | | |
|---|---|---|---|---|---|---|
| **Models** | **MRR** | **Hit@1** | **Hit@3** | **MRR** | **Hit@1** | **Hit@3** |
| CompGCN | 0.661 | 0.573 | 0.705 | 0.655 | 0.565 | 0.702 |
| RGCN | 0.673 | 0.590 | 0.716 | 0.667 | 0.584 | 0.708 |
| CET | 0.697 | 0.613 | 0.744 | 0.687 | 0.601 | 0.733 |
| TET_RSE | 0.699 | 0.613 | 0.748 | 0.698 | 0.613 | 0.749 |
| TET | **0.712** | **0.631** | **0.758** | **0.705** | **0.624** | **0.753** |

Table 9: Evaluation with different dropping rates on FB15kET. TET_RSE represents TET with semantic enhancement on relations.

| Dropping Rates | 25% | | | 50% | | |
|---|---|---|---|---|---|---|
| **Models** | **MRR** | **Hit@1** | **Hit@3** | **MRR** | **Hit@1** | **Hit@3** |
| CompGCN | 0.664 | 0.578 | 0.708 | 0.662 | 0.574 | 0.708 |
| RGCN | 0.676 | 0.593 | 0.719 | 0.673 | 0.590 | 0.719 |
| CET | 0.699 | 0.617 | 0.743 | 0.694 | 0.610 | 0.742 |
| TET_RSE | 0.708 | 0.628 | 0.754 | **0.712** | **0.634** | 0.755 |
| TET | **0.711** | **0.631** | **0.756** | 0.710 | 0.630 | **0.757** |
| **Dropping Rates** | **75%** | | | **90%** | | |
| **Models** | **MRR** | **Hit@1** | **Hit@3** | **MRR** | **Hit@1** | **Hit@3** |
| CompGCN | 0.653 | 0.565 | 0.699 | 0.637 | 0.546 | 0.683 |
| RGCN | 0.658 | 0.573 | 0.702 | 0.636 | 0.548 | 0.681 |
| CET | 0.675 | 0.588 | 0.721 | 0.653 | 0.564 | 0.700 |
| TET_RSE | 0.687 | 0.604 | 0.733 | 0.675 | **0.591** | 0.718 |
| TET | **0.690** | **0.608** | **0.734** | **0.677** | **0.591** | **0.722** |

Table 10: Evaluation with different dropping rates on relations on FB15kET. The TET_RSE represents TET with semantic enhancement on relations.

**Effect of Dropping Rates: Number of Relation Types.** In Section 3.2.2, we enhanced relations with semantic knowledge for the YAGO43kE KG. The main reason for doing this only for YAGO43kE is that it contains only very few relation types (cf. Table 2), making the discrimination among relation triples harder. As a first step towards having a better understanding of the interplay of the number of relation types and the number of types in a KG, we

conduct an ablation study in which on FB15kET we randomly remove 25%, 50%, 75%, and 90% of the relation types. The results in Table 10 show that in this case enhancing relation types with semantic knowledge does not have a positive effect, unlike for YAGO43kE. We believe that the main reason behind this is that YAGO43kE does not only have very few relations, but also a very large number of types. To have a precise understanding, a dedicated deep analysis of the interplay of the number of types, the number of relation types, and other structural characteristics of KGs is required - it is out of the scope of this paper, but it is an interesting question for future work.