Knowledge Graph Error Detection with Contrastive Confidence Adaption

Xiangyu Liu[1], **Yang Liu**[1], **Wei Hu**[1,2,*]

[1] State Key Laboratory for Novel Software Technology, Nanjing University, China
[2] National Institute of Healthcare Data Science, Nanjing University, China
{xyl, yliu20}.nju@gmail.com, whu@nju.edu.cn

Abstract

Knowledge graphs (KGs) often contain various errors. Previous works on detecting errors in KGs mainly rely on triplet embedding from graph structure. We conduct an empirical study and find that these works struggle to discriminate noise from semantically-similar correct triplets. In this paper, we propose a KG error detection model CCA to integrate both textual and graph structural information from triplet reconstruction for better distinguishing semantics. We design interactive contrastive learning to capture the differences between textual and structural patterns. Furthermore, we construct realistic datasets with semantically-similar noise and adversarial noise. Experimental results demonstrate that CCA outperforms state-of-the-art baselines, especially in detecting semantically-similar noise and adversarial noise.

Introduction

A knowledge graph (KG) is composed of triplets in the form of $(head\ entity, relation, tail\ entity)$, which finds extensive applications in downstream tasks like question answering (Saxena, Tripathi, and Talukdar 2020) and recommender systems (Guo et al. 2022). Existing KGs such as NELL (Carlson et al. 2010) and Knowledge Vault (Dong et al. 2014) continuously extract triplets in an automatic way, which inevitably introduces noise. Detecting these errors holds the potential to improve the quality of KGs.

Existing works on KG error detection can be classified into embedding-based and path-based models. The former (Bordes et al. 2013; Yang et al. 2015; Trouillon et al. 2016) learns confidence scores based on the representations of entities and relations. The latter (Lin et al. 2015; Jia et al. 2019) uses paths between entities to evaluate the confidence of triplets. Different from the task of link prediction (Chen et al. 2021) or triplet classification (Yao, Mao, and Luo 2019), error detection focuses on detecting the error triplets in the whole unsupervised KG, aiming to capture the variance of triplets and give an accurate estimate of their confidence.

Current KG error detection models face a significant challenge due to the unavailability of noise patterns and the difficulty in acquiring accurately labeled noise samples for robust supervision. Negative sampling by replacing entities is widely used in previous works, especially the embedding-based models. However, real-world scenarios often introduce confusing noise semantically related to correct samples. Let us see two real error triplets in the FB15K-237 dataset (Toutanova et al. 2015): *(George Lopez, profession, Disc jockey)* and *(Majel Barrett, profession, Writer)*. In the former, the correct tail entity should be ‘Comedian’, and in the latter, the correct tail entity should be ‘Actress’. Notably, ‘Disc jockey’ and ‘Writer’ both represent professions, mirroring common human errors. This form of noise is harder to differentiate and aligns closely with human error tendencies.

Models	Noise types	FB15K-237		WN18RR	
		K=1%	K=5%	K=1%	K=5%
CAGED	Random	0.945	0.758	0.795	0.486
	Similar	0.633	0.367	0.657	0.384
TransE	Random	0.904	0.726	0.630	0.434
	Similar	0.611	0.303	0.503	0.361

Table 1: Results of our empirical study. Error triplets are divided into the “random” and “similar” groups, based on the methods of replacing entities. We show the top-K precision of two typical models on FB15K-237 and WN18RR.

We conduct a further empirical study to explore the performance of existing works on two specific types of noise: random noise and semantically-similar noise. Random noise is generated by randomly replacing the head or tail entity of a correct triplet, which is used in previous works (Dong et al. 2023). Semantically-similar noise uses entities that have co-occurrence with their relations, which means that it is semantically related to the relation. We test two typical models CAGED (Zhang et al. 2022) and TransE (Bordes et al. 2013) on the FB15K-237 (Toutanova et al. 2015) and WN18RR (Dettmers et al. 2018) datasets to verify whether existing works can deal with more realistic noise. As shown in Table 1, we add 5% of the two types of noise to the datasets and show the precision of the top-1% and top-5% detected triplets. The results show that, although the existing methods CAGED and TransE perform well on random noise, their effects are greatly reduced on semantically-similar noise. This is because existing models predominantly rely on graph structure, ignoring the rich textual information of KGs. The

[*]Corresponding author

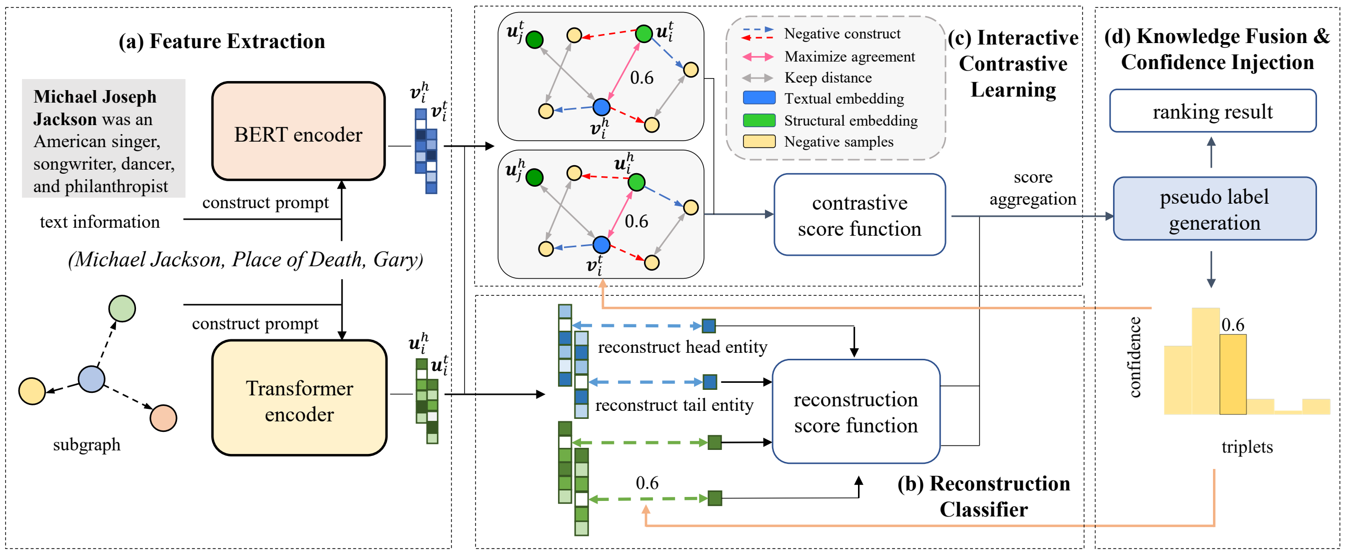

Figure 1: An overview of the proposed model CCA. (a) BERT and Transformer-based graph encoders extract textual and graph structural information, respectively. (b) The reconstruction module classifies error triplets by reconstructing head and tail entities in textual and structural embedding. (c) Interactive contrastive learning aligns the projection of textual and structural embeddings and recognizes errors by inter-model difference. (d) The knowledge fusion module takes pseudo labels generated from aggregated results as triplet confidence, which is further injected into the training process.

incompleteness of KGs leads to the lack of important information that can distinguish semantically-similar noise, which exhibits greater semantic relevance and shares a similar graph structure with correct triplets.

Furthermore, we consider the prevalent scenarios of KG error detection to build adversarial noise. For automatic KG construction and completion, noise is inevitably introduced. An effective error detection model should possess the capability to pinpoint triplets that have been inaccurately completed or constructed. We filter out the error results from the construction model to constitute adversarial noise. Subsequent experiments also show that textual information plays a key role in the noise generated by completion which relies on graph structure.

Existing works need a potent method to discern more realistic noise and extract the full potential of textual information within KGs for error detection. To achieve this goal, we opt to leverage a pre-trained language model (PLM) as the encoder of textual information in our model. We refer to KG-BERT (Yao, Mao, and Luo 2019), PKGC (Lv et al. 2022), MLMLM (Clouatre et al. 2021), and CoLE (Liu et al. 2022), which perform well in KG embedding and link prediction. A PLM uses a large number of open-domain corpora for training and can supplement rich information for triplets.

We propose a novel KG error detection model CCA. It leverages the reconstruction of triplets to comprehend noise patterns from both textual and graph structural perspectives. Also, we design interactive contrastive learning to align the latent representations of textual and structural information. It facilitates noise identification based on disparities between these two forms of information. CCA combines the reconstruction and contrastive learning output and generates pseudo-labels to represent triplet confidence. This adaptive confidence guides model training by alleviating noise interference and transferring knowledge between reconstruction and contrastive learning. With the utilization of textual information, CCA not only outperforms the state-of-the-art methods on random noise but also performs more prominently in semantically-similar noise and adversarial noise, validating the effectiveness of CCA in complex real-world scenarios.

In summary, this paper makes the following contributions:

- We propose an end-to-end KG error detection model, which fully leverages both textual and structural information by reconstructing triplets, and alleviates the interference of noise. It transfers the knowledge between reconstruction and contrastive learning.
- We design interactive contrastive learning to align the latent representations of textual and structural information. We use different negative sampling policies to mine anomalous features on the alignment of latent spaces.
- We construct two kinds of noise, semantically-similar noise and adversarial noise, to evaluate the performance of our model in more realistic scenarios. Experiments show that CCA not only surpasses the state-of-the-art competitors on random noise but also achieves more significant results on the datasets with semantically-similar noise and adversarial noise. Datasets and source code are available at https://github.com/nju-websoft/CCA.

Related Work

KG Error Detection

The embedding-based models, e.g., TransE (Bordes et al. 2013), DistMult (Yang et al. 2015), and ComplEx (Trouillon et al. 2016), evaluate triplets by designing score functions according to the representations of entities and relations. TransE employs $\|\mathbf{h} + \mathbf{r} - \mathbf{t}\|$ as the score function to assess triplets. Subsequent works such as CKRL (Xie et al. 2018), CAGED (Zhang et al. 2022), and SCEF (Zhao and Liu 2019) adopt the TransE's score function as a component. The embedding-based models heavily rely on negative sampling, and their effectiveness is hampered by the challenge of accurately modeling the real noise distribution. Consequently, the negative samples often deviate from the real noise, undermining the error detection performance.

Some works exploit path information in KGs as a reference. PTransE (Lin et al. 2015) integrates relation path information into triplet embedding learning. KGTtm (Jia et al. 2019) leverages path information for error detection.

Several works recognize that relying solely on graph structure may not be sufficient for robust KG error detection. Integrating additional information can offer more semantic insights into triplets or provide more accurate supervision signals, thereby enhancing model capabilities. Defacto (Lehmann et al. 2012) uses the information from relevant web pages to verify triplets, while the categorical information of entities and relations is also useful to assess the trustworthiness of triplets (Paulheim and Bizer 2014). CrossVal (Wang, Ma, and Gao 2020) introduces human-curated knowledge repositories to assess triplet confidence in the target KG through the correlation of triplets in two KGs. Crowdsourcing and active learning models, such as KAEL (Dong et al. 2023), show promising results. CKRL (Xie et al. 2018) and PGE (Cheng et al. 2022) also explore confidence-aware methods. Their confidence constraints are based solely on the score function of the TransE's loss.

The existing models mainly focus on utilizing graph structure while under-utilizing valuable textual information. By considering this, our proposed model CCA seeks to overcome these limitations by jointly leveraging textual and graph structural information, aiming to achieve more accurate and comprehensive KG error detection.

PLMs for KGs

While KG error detection using PLMs is relatively scarce, the use of PLMs has demonstrated success in various tasks like KG completion. KG-BERT (Yao, Mao, and Luo 2019) and PKGC (Lv et al. 2022) achieve good results by directly fine-tuning PLMs by the triplet classification tasks using carefully constructed input templates. MMLLM (Clouatre et al. 2021) effectively employs the [MASK] token in prompts to enhance link prediction.

To fuse language models with structural models, (Nadkarni et al. 2021; Wang et al. 2021; Chen et al. 2023) leverage the text enhancement method to represent entities by combining textual and structural features. CoLE (Liu et al. 2022) explores the use of the two types of information to construct models independently and uses knowledge distillation to complement and improve each other's performance.

The Proposed Model

In this section, we describe the proposed model CCA in detail. The framework is shown in Figure 1. The goal of the proposed model is to better leverage both textual and graph structural information at the same time. Given a triplet (h, r, t), we first construct its input sequence by its subgraph and the descriptions of the entities and the relation. Then, BERT and a Transformer-based encoder are used to extract textual features and graph structural features, respectively. Reconstruction loss is employed to construct the head or tail entity in the triplet and evaluate the triplet's trustworthiness. Interactive contrastive learning utilizes the projection of textual and structural representations to discover anomalous features in the two alignment spaces. The scores from contrastive learning and reconstruction classifier are aggregated to generate a pseudo label as the confidence to dynamically constrain the training process.

P_t^T = [CLS] [SEP]$_1^r$ N_h [SEP]$_2^r$ N_r [SEP]$_3^r$ [MASK] [SEP]$_4^r$ D_h
P_h^T = [CLS] [SEP]$_1^r$ [MASK] [SEP]$_2^r$ N_r [SEP]$_3^r$ N_t [SEP]$_4^r$ D_t
P_t^S = [CLS] [h] [SEP] [r] [SEP] [MASK] [SEP] [h'] [r']...
P_h^S = [CLS] [MASK] [SEP] [r] [SEP] [t] [SEP] [t'] [r']...

Table 2: Inputs used in the textual and structural encoders

Feature Extraction

Text encoder. Currently PLMs like BERT (Kenton and Toutanova 2019) have been widely used to extract textual features from KGs. For each entity e, following the previous work (Liu et al. 2022), we leverage its description D_e and human-readable literal name N_e to learn the textual representation. We first add the corresponding representation of the new entity into the PLM's vocabulary. To better leverage the description of the entity, pre-trained prompts are constructed for initializing the new token embedding $\mathbf{e}$:

$$\mathbf{e} = \mathrm{BERT}(\text{“The description of [MASK] is } D_e\text{”}), \quad (1)$$

where [MASK] denotes the missing entity and we use its embedding to initialize the token embedding $\mathbf{e}$. After pre-training on the description, we construct two prompts through the masked head and tail entities to better leverage textual information.

Given a triplet (h, r, t), we use the prompts presented in Table 2 as the input sequence. In P_h^T, the [MASK] token is used to replace the original head entity, and the tail entity t in P_t^T is also masked, so that h and t can be reconstructed respectively by the encoder. [CLS] and [SEP] are two special tokens used to separate the different parts of the prompt. [SEP]$_i^r$ is the i-th adjustable soft prompt token for the relation r, as a more expressive separator. We provide the textual description D_t of the tail entity in P_h^T, and D_h of the head entity is used in P_t^T. If the tail entity t is an anomalous entity, it is hard to reconstruct t through the text description of h in P_t^T, so the reconstruction loss can be used to evaluate the error possibility of triplets from the textual view.

Structure encoder. To make full use of the graph structure in KGs, we build an entity's subgraph by random sampling and use Transformer to encode it. Given a triplet (h, r, t), we also create two input sequences P_h^S and P_t^S by masking h and t, respectively. For P_h^S, let $InNeighbor(h) = \{(h', r') \mid (h', r', h) \in \Gamma\}$ and $OutNeighbor(h) =$

$\{(t',r') \,|\, (h,r',t') \in \Gamma\}$ denote two neighbor sets of entity h, where Γ is the triplet set. We randomly select some neighbors and separate them by the [SEP] token, as shown in Table 2. P_t^S is constructed in the same way.

The structure encoder can learn the differences between t and the subgraph of h. Therefore, by reconstructing entities from the subgraphs, errors can be distinguished.

Reconstruction Classifier

Given a triplet (h,r,t), we can get the output representation $\mathbf{e}_h^T$ of the masked head entity from the text encoder:

$$\mathbf{e}_h^T = \mathrm{BERT}(P_h^T), \tag{2}$$

where P_h^T is the input sequence that we construct in Table 2.

Then, we can calculate the prediction logits against all candidate entities and get the cross-entropy loss $\mathcal{L}_h^T$ according to the corresponding labels:

$$\mathcal{L}_h^T = CE\Big(\mathrm{softmax}\big(\mathrm{MLP}(\mathbf{e}_h^T), \mathbf{E}^T\big), \mathbf{y}_h\Big), \tag{3}$$

where $\mathbf{E}^T$ is the matrix of all entity embeddings and $\mathbf{y}_h$ is the corresponding labels. $\mathbf{e}_h^T$ is transferred by a multi-layer perceptron and used to calculate the prediction logits with all embeddings in $\mathbf{E}^T$. $CE()$ denotes the cross-entropy loss. Note that the tail entity should also be reconstructed, and we calculate $\mathbf{e}_t^T$ and $\mathcal{L}_t^T$ in the same way.

Next, we introduce adaptive confidence to the reconstruction loss and obtain the final text reconstruction loss:

$$\mathcal{L}_{\text{text}} = \sum_{(h,r,t)\in\Gamma} c(h,r,t)\big(\mathcal{L}_h^T + \mathcal{L}_t^T\big), \tag{4}$$

where the triplet confidence $c(h,r,t)$ is calculated in Section shortly. Note that we use the same way on the Transformer encoder to get the masked entity losses $\mathcal{L}_h^S$ and $\mathcal{L}_t^S$, and the overall reconstruction loss $\mathcal{L}_{struct}$ similar to Eqs. (2), (3), and (4). We add $\mathcal{L}_{text}$ and $\mathcal{L}_{struct}$ and jointly train the final reconstruction loss $\mathcal{L}_{\text{reconstruct}}$ as follows:

$$\mathcal{L}_{\text{reconstruct}} = \alpha\mathcal{L}_{\text{text}} + (1-\alpha)\mathcal{L}_{\text{struct}}, \tag{5}$$

where α is a hyperparameter to balance the training process.

Finally, we use the cross-entropy loss to compute the scores of the triplet (h,r,t):

$$score_{\text{text}} = \mathcal{L}_h^T + \mathcal{L}_t^T, \tag{6}$$

$$score_{\text{struct}} = \mathcal{L}_h^S + \mathcal{L}_t^S, \tag{7}$$

where $score_{\text{text}}$ and $score_{\text{struct}}$ are the confidence scores from text and structure reconstruction, respectively.

Interactive Contrastive Learning

We use interactive contrastive learning (Yang et al. 2022) to learn the differences between textual and structural information. For a triplet (h,r,t), $\mathbf{v}_i^h$ and $\mathbf{v}_i^t$ represent the token embeddings generated from the PLM by P_h^T and P_t^T, respectively. $\mathbf{u}_i^h$ and $\mathbf{u}_i^t$ represent the embeddings from the structure encoder. $\mathbf{v}_i^h$ contains the textual information of t and the reconstruction of h from the PLM. $\mathbf{u}_i^t$ has the structural information of h and the reconstruction of t from Transformer. Intuitively, $\mathbf{v}_i^h$ and $\mathbf{u}_i^t$ should be aligned, so that the prediction of h can match its structural information and the prediction of t can match its textual information. Therefore, we use $\mathbf{v}_i^h$ and $\mathbf{u}_i^t$ as one anchor pair and $\mathbf{v}_i^t$ and $\mathbf{u}_i^h$ as the other pair for contrastive learning.

Negative sampling. In interactive contrastive learning, we maximize the agreement between the anchor pair like $\mathbf{v}_i^h$ from the textual encoder and its structural representation $\mathbf{u}_i^t$ for one triplet (h,r,t). We use two negative sampling policies to support stable training.

First, we construct negative samples to align the latent spaces from the two encoders. For the anchor sample $\mathbf{v}_i^h$ from the text encoder, another sample in the structure encoder like $\mathbf{u}_j^t$ should keep a distance from the anchor sample. Thus, we randomly take another sample from the structure encoder as one of the negative samples.

To improve the sensitivity and robustness of the model against noise, we construct error samples as negative examples based on the anchor. We randomly replace the head (or tail) entity of the anchor sample (h,r,t) to construct a typical error sample (h,r,t') (or (h',r,t)), and generate its representations $\mathbf{m}_{i_k}^h$ and $\mathbf{n}_{i_k}^t$ from different encoders, respectively. Note that (h,r,t') is in fact the noise generated from disturbance. Therefore, $\mathbf{m}_{i_k}^h$ and $\mathbf{n}_{i_k}^t$ should not be aligned.

To reduce the training cost, we directly use the representation of the entity obtained by the encoder in each training batch to disturb original embeddings. For a triplet (h,r,t), the similarity of its negative samples is calculated as

$$\begin{aligned} neg\_sim(h,r,t) = &\sum_{j\neq i}\exp\big(sim(\mathbf{v}_i^h,\mathbf{u}_j^t)\big) \\ &+\sum_{x=1}^{X}\exp\big(sim(\mathbf{m}_{i_x}^h,\mathbf{n}_{i_x}^t)\big), \end{aligned} \tag{8}$$

where $\mathbf{v}_i^h$ denotes the masked head triplet embedding of the i-th sample and $\mathbf{u}_j^t$ denotes the masked tail triplet embedding of the j-th sample. $sim()$ denotes the cosine similarity used to calculate the distance of the two embeddings. $\mathbf{m}_{i_x}^h$ and $\mathbf{n}_{i_x}^t$ represent the randomly replaced embeddings from $\mathbf{v}_i^h$ and $\mathbf{u}_i^t$, and we replace them for $X = 4$ times randomly.

Adaptive contrastive learning. We employ the InfoNCE loss (van den Oord, Li, and Vinyals 2018) to train interactive contrastive learning. For the anchor pair $\mathbf{v}_i^h$ and $\mathbf{u}_i^t$, the loss $\mathcal{L}_{\text{ICL}}^1(h,r,t)$ is

$$\mathcal{L}_{\text{ICL}}^1(h,r,t) = -\log\frac{\exp(sim(\mathbf{v}_i^h,\mathbf{u}_i^t))}{\exp(sim(\mathbf{v}_i^h,\mathbf{u}_i^t))+neg\_sim(h,r,t)}, \tag{9}$$

where $neg\_sim(h,r,t)$ is the similarity of negative samples. Note that if we choose $\mathbf{v}_i^t$ and $\mathbf{u}_i^h$ as the anchor pair, we can get $\mathcal{L}_{\text{ICL}}^2(h,r,t)$ in the same way. Therefore, the final contrastive training loss is

$$\mathcal{L}_{\text{contr}} = \sum_{(h,r,t)\in\Gamma} c(h,r,t)\big(\mathcal{L}_{\text{ICL}}^1(h,r,t) + \mathcal{L}_{\text{ICL}}^2(h,r,t)\big), \tag{10}$$

where $c(h,r,t)$ is the adaptive confidence of (h,r,t). The score function of contrastive learning of (h,r,t) is

$$score_{\text{contrastive}} = sim(\mathbf{v}_i^h,\mathbf{u}_j^t). \tag{11}$$

Datasets	Correct triplets	Wrong triplets	Avg. deg.
FB15K-237	310,116	16,321	44.89
WN18RR	93,003	4,894	4.78

Table 3: Statistics of the modified datasets

Knowledge Fusion

Since $score_{\text{text}}$ and $score_{\text{struct}}$ are both logits from triplet reconstruction, we directly add them to obtain the reconstruction score:

$$score_{\text{reconstruct}} = score_{\text{text}} + \lambda\, score_{\text{struct}}, \tag{12}$$

where λ is the hyperparameter to balance the scores. Note that the scores from reconstruction and contrastive learning have quite different distributions, so we use ranking to combine them. By ranking the error scores $score_{\text{reconstruct}}$ and $score_{\text{contrastive}}$, we obtain the error ranks $R_1(h,r,t)$ and $R_2(h,r,t)$ of each triplet, respectively. The final score is

$$score(h,r,t) = \frac{1}{\lceil \frac{R_1(h,r,t)}{\gamma} \rceil} + \frac{1}{\lceil \frac{R_2(h,r,t)}{\gamma} \rceil^{\beta}}, \tag{13}$$

where γ is the hyperparameter to control the number of triplets with the same score. β is used to balance the scores.

We generate a pseudo label from the final score. Let $Z = normalize\big([z_1,\ldots,z_n]\big)$ be the pseudo label set, where $z_i \sim N(\mu,\rho)$ and Z is sorted in ascending order, the adaptive confidence of (h,r,t) is as follows:

$$c(h,r,t) = z_{R(h,r,t)}, \tag{14}$$

where $R(h,r,t)$ is the rank of $score(h,r,t)$ and we use $c(h,r,t)$ as the final result. Note that $c(h,r,t)$ is the result integrated from textual and structural reconstruction and contrastive learning. We use $c(h,r,t)$ directly constrain the training process according to Eqs. (4) and (10) to transfer knowledge across components.

Experiments and Results

Dataset Construction

We conduct our experiments on FB15K-237 (Toutanova et al. 2015) and WN18RR (Dettmers et al. 2018). We add 5% noise into these two datasets. The statistics of the new datasets are shown in Table 3. We construct three types of noise to better evaluate the performance of our model:

- **Random noise**, drawn from a uniform distribution that does not depend on the data and is not predictable. For a correct triplet (h,r,t), random noise is constructed by randomly replacing one of the entities or the relation.
- **Semantically-similar noise**, which is more realistic because errors primarily stem from semantic confusion in real-world scenarios. To better evaluate semantically-related error detection, we introduce semantically-similar noise. Given a triplet (h,r,t), we form a candidate set S_t by selecting other tail entities linked to r. For each entity $e \in S_t$, we employ BERT to encode its description, acquiring the semantic embedding $\mathbf{e}$ of e. We calculate the sampling probability distribution of semantic similarity as follows and utilize it as a sampling probability to replace the original t:

$$\mathbf{P}(t) = \text{softmax}\Big(\big[\mathbf{t}\cdot\mathbf{e}_1, \mathbf{t}\cdot\mathbf{e}_2, \ldots, \mathbf{t}\cdot\mathbf{e}_i\big]\Big), \tag{15}$$

  where $\mathbf{t}$ denotes the embedding of the original tail entity t, $\mathbf{e}_i$ denotes the embedding of the i-th candidate entity, and $\cdot$ denotes the dot product.
- **Adversarial noise**, which is adversarially generated from KG construction models. Since error detection models are frequently employed to discern errors in automatically constructed KGs, we use TransE (Bordes et al. 2013) for adversarial noise generation and evaluate the ability of models to recognize errors during KG construction. Given a dataset D, we randomly divide it into training and testing sets, D_{train} and D_{test}. After training on D_{train}, we randomly select one entity within the top-10 prediction results of the triplet in D_{test} to construct its error triplet. We iteratively repeat this process until we get an adequate amount of noise.

Settings

We use the ranking measures for evaluation. All triplets in a KG are ranked based on the confidence scores in ascending order. Triplets with lower confidence scores are more likely to be noisy. We follow CAGED (Zhang et al. 2022) and use precision@top-K and recall@top-K to assess the performance, where K denotes the ratio (e.g., 5%).

All experiments are conducted on two Intel Xeon Gold 6326 CPUs, 512GB RAM, and one NVIDIA RTX A6000 GPU. We leverage the BERT-base model from huggingface as the PLM. We use PyTorch to implement our model and employ the AdamW optimizer and a cosine decay scheduler with a linear warm-up for optimization. The grid search is used for hyperparameter tuning. The results of KG embedding models are obtained from μKG (Luo, Sun, and Hu 2022), a recent open-source library for KG embedding.

Baseline Models

We compare our model with eight baselines, including five structural models and three textual models.

- **Structural models.** We choose three typical embedding models TransE (Bordes et al. 2013), DistMult (Yang et al. 2015), and ComplEx (Trouillon et al. 2016) and two state-of-the-art structural error detection models KGTtm (Jia et al. 2019) and CAGED (Zhang et al. 2022) for comparison. Both KGTtm and CAGED utilize graph structural information based on the TransE's score function.
- **Textual models.** We choose one textual classification model KG-BERT (Yao, Mao, and Luo 2019), and two recent models StAR (Wang et al. 2021) and CSProm-KG (Chen et al. 2023), which leverage both textual and structural information, for comparison. KG-BERT takes entity and relation descriptions of a triplet as input and computes scores by binary classification. StAR and CSProm-KG use the representations obtained from PLMs to aid structural models. CCA also falls into this category.

	Model types	Models	FB15K-237					WN18RR				
			K=1%	K=2%	K=3%	K=4%	K=5%	K=1%	K=2%	K=3%	K=4%	K=5%
Precision@top-K	Struct models	TransE	0.946	0.774	0.606	0.498	0.423	0.690	0.576	0.501	0.437	0.400
		DistMult	0.764	0.630	0.530	0.463	0.410	0.687	0.633	0.526	0.438	0.374
		ComplEx	0.802	0.633	0.521	0.446	0.393	0.774	0.699	0.550	0.449	0.384
		KGTtm	0.857	0.687	0.631	0.467	0.437	0.789	0.644	0.541	0.473	0.417
		CAGED	0.863	0.666	0.602	0.543	0.467	0.753	0.620	0.536	0.470	0.421
	Text models	KG-BERT	0.966	0.799	0.660	0.584	0.498	0.973	**0.968**	**0.938**	0.829	0.710
		StAR	**0.970**	**0.835**	0.681	0.571	0.490	0.971	0.918	0.842	0.739	0.647
		CSProm-KG	0.961	0.798	0.689	0.574	0.509	0.977	0.927	0.869	0.773	0.680
		CCA (ours)	0.969	0.812	**0.707**	**0.599**	**0.534**	**0.986**	0.959	0.920	**0.834**	**0.733**
Recall@top-K	Struct models	TransE	0.189	0.310	0.364	0.399	0.423	0.138	0.231	0.300	0.350	0.400
		DistMult	0.153	0.252	0.318	0.371	0.410	0.137	0.253	0.316	0.350	0.374
		ComplEx	0.161	0.254	0.313	0.357	0.393	0.155	0.279	0.330	0.359	0.384
		KGTtm	0.171	0.275	0.378	0.374	0.437	0.158	0.257	0.324	0.378	0.417
		CAGED	0.173	0.266	0.362	0.435	0.467	0.150	0.248	0.321	0.376	0.421
	Text models	KG-BERT	0.193	0.319	0.396	0.467	0.498	0.195	**0.387**	**0.563**	0.663	0.710
		StAR	**0.194**	**0.334**	0.409	0.457	0.490	0.194	0.367	0.505	0.591	0.647
		CSProm-KG	0.192	0.319	0.413	0.459	0.509	0.195	0.371	0.521	0.618	0.680
		CCA (ours)	**0.194**	0.325	**0.424**	**0.479**	**0.534**	**0.197**	0.384	0.552	**0.667**	**0.733**

Table 4: Results of precision and recall at top-K on FB15K-237 and WN18RR

	Models	K=1%	K=3%	K=5%
FB15K-237	CCA (full)	.969 / .194	.707 / .424	.535 / .535
	– adapt conf.	.951 / .190	.664 / .398	.496 / .496
	– inter contr.	.961 / .192	.679 / .407	.509 / .509
	– struct recon.	.797 / .159	.582 / .349	.475 / .475
	– text recon.	.778 / .156	.543 / .325	.414 / .414
WN18RR	CCA (full)	.986 / .197	.920 / .552	.733 / .733
	– adapt conf.	.971 / .194	.858 / .515	.632 / .632
	– inter contr.	.979 / .196	.911 / .546	.722 / .722
	– struct recon.	.974 / .195	.914 / .549	.726 / .726
	– text recon.	.577 / .115	.513 / .308	.423 / .423

Table 5: Ablation results of precision and recall at top-K

	Models	Random	Similar	Adversarial
FB15K-237	TransE	0.726	0.304	0.125
	ComplEx	0.734	0.306	0.150
	DistMult	0.662	0.328	0.125
	KGTtm	0.730	0.318	0.128
	CAGED	0.758	0.331	0.126
	KG-BERT	0.674	0.336	0.199
	StAR	0.728	0.371	0.164
	CSProm-KG	0.732	0.419	0.211
	CCA (ours)	**0.768**	**0.453**	**0.240**
WN18RR	TransE	0.434	0.351	0.314
	ComplEx	0.384	0.303	0.366
	DistMult	0.316	0.338	0.285
	KGTtm	0.448	0.391	0.359
	CAGED	0.486	0.388	0.373
	KG-BERT	0.806	0.633	**0.599**
	StAR	0.794	0.610	0.527
	CSProm-KG	0.791	0.634	0.536
	CCA (ours)	**0.807**	**0.657**	0.545

Table 6: Precision at top-5% on three different error types

Main Results

Table 4 presents the comparison results of our model and eight baseline models on FB15K-237 and WN18RR, where we add 5% noise, containing three types of noise in equal quantities. Overall, our model outperforms eight baseline models on both datasets. We have three observations below:

First, compared with the KG embedding models, the error detection models generally perform better. This is because the KG embedding models assume that all triplets are correct. They learn representations of entities and relations without alleviating disturbance from noise, making it difficult to discriminate error triplets. With adaptive confidence, CCA can effectively improve performance.

Second, benefiting from PLMs, the textual models outperform the structural models on the two datasets. They both can learn noise patterns in the textual view with the descriptions of entities and relations, considering that PLMs can capture factual knowledge from a large amount of open-domain corpora. CCA combines both textual information and structural information and uses a Transformer encoder to learn structural information from scratch, which is more effective than other textual models.

Third, the performance gap between CCA and KG-BERT on WN18RR is less significant than that on FB15K-237. We think the reason is that WN18RR is much sparser than FB15K-237. As shown in Table 3, the average degree of entities in WN18RR is far less than that of FB15K-237, which indicates that there is less structural information in WN18RR. Thus, our model obtains less improvement by combining with graph structure on sparse KGs.

			CCA	KG-BERT	CAGED
Case 1.	Noise	Razzie Award for Worst Actor, award winner, Richard D. Zanuck	2.9%	7.6%	3.4%
	Truth	Razzie Award for Worst Actor, award winner, Marlon Wayans			
Case 2.	Noise	Tony Award for Best Choreography, ceremony, 54th Academy Awards	3.2%	10.6%	11.6%
	Truth	Academy Award for Best Sound Mixing, ceremony, 54th Academy Awards			
Case 3.	Noise	Lauren Tom, place of birth, Buenos Aires	2.2%	3.2%	35.1%
	Truth	Sebastian Krys, place of birth, Buenos Aires			
Case 4.	Noise	Viola Davis, award nominee, Carrie-Anne Moss	1.0%	10.9%	4.3%
	Truth	Viola Davis, award nominee, Meryl Streep			

Table 7: Case study on FB15K-237

Ablation Study

We conduct an ablation study to assess the impact of each component in CCA. We have four variants of our model by removing confidence adaption, interactive contrastive learning, structure reconstruction, or text reconstruction. We report their performance under the same settings.

Table 5 shows that all components contribute to our model on both datasets, where text reconstruction contributes the most. On FB15K-237, the precision improvement of text construction is from 0.414 to 0.535. This shows that applying PLMs to KG error detection is a promising way, and it deserves further research. The improvement is more obvious on WN18RR. The reason is that the structure-based models perform poorly, as WN18RR is sparser than FB15K-237.

Removing adaptive confidence affects differently on the two datasets. On WN18RR, adaptive confidence contributes more than FB15K-237, as the performance gap between text and structure reconstruction is larger. Without adaptive confidence, structure reconstruction is hard to gain benefits from the knowledge of PLMs, which provides scores with lower accuracy and disturbs the final results. Therefore, adaptive confidence can be more effective in improving overall performance when components have a larger performance gap.

Performance on Different Error Types

To investigate the robustness of all models to different noise types, we add three types of 5% noise separately to the datasets. Note that precision@top-5% and recall@top-5% have the same value as we add 5% noise. Table 6 presents precision@top-5% on FB15K-237 and WN18RR.

On FB15K-237, CCA outperforms all baselines. For random noise, the structure-based models generally outperform KG-BERT, as this type of noise can be well distinguished by graph structure alone. For semantically-similar noise, all baseline models perform closely. As for adversarial noise, the textual models outperform the structural ones, because this type of noise consists of entities that are wrongly predicted as correct entities by TransE, which is a structural model. Our CCA takes advantage of the knowledge from PLMs and graph structure at the same time, so it outperforms all baselines, especially for semantically-similar noise.

On WN18RR, CCA is comparable to KG-BERT, which is different from the observations on FB15K-237. The models with PLMs largely outperform the structural models on all three types of noise, mainly due to that WN18RR is sparser than FB15K-237. For adversarial noise, our model underperforms KG-BERT. Given that the structure-based models do not work well on WN18RR, Transformer in CCA provides less reliable knowledge for the PLM, hindering it from distinguishing the noise correctly.

Case Study

To explore how textual information and graph structure act on error detection, we perform a case study on CCA, KG-BERT, and CAGED. Table 7 shows the position of the error triplet in a ranking, where a smaller percentage is better.

In the first two cases, CCA leverages both textual and graph structural information, and outperforms KG-BERT and CAGED, which solely use one type of information. For example, in Case 1, the description shows that Richard D. Zanuck is an American film producer, and the graph structural information records films that he has produced. Evidence from text and graph structure are complementary to each other. In Case 3, CAGED is not effective compared with CCA and KG-BERT, which is caused by the lack of graph structural information. The degrees of entities "Lauren Tom" and "The Venture Bros" are 15 and 17, respectively, which are much smaller than the average degree of 44.89 in FB15K-237. In Case 4, entities in the noise and correct triplets all have abundant graph structural information, so CAGED can achieve a better effect.

Conclusion

In this paper, we propose a novel KG error detection model. It encodes textual and graph structural information to find noise patterns. To alleviate the disturbance of noise and integrate the knowledge from the two encoders, we design a confidence adaption model to aggregate the results and constrain the training process. To learn the noise patterns between textual and structural information, we leverage interactive contrastive learning to align latent spaces. We construct semantically-similar noise and adversarial noise for evaluation. Experiments show that our model achieves good results on semantically-similar noise and adversarial noise.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (No. 62272219).

References

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

Carlson, A.; Betteridge, J.; Kisiel, B.; Settles, B.; Hruschka, E.; and Mitchell, T. 2010. Toward an architecture for never-ending language learning. In *AAAI*, 1306–1313.

Chen, C.; Wang, Y.; Sun, A.; Li, B.; and Lam, K.-Y. 2023. Dipping PLMs sauce: Bridging structure and text for effective knowledge graph completion via conditional soft prompting. In *Findings of ACL*, 11489–11503.

Chen, S.; Liu, X.; Gao, J.; Jiao, J.; Zhang, R.; and Ji, Y. 2021. HittER: Hierarchical transformers for knowledge graph embeddings. In *EMNLP*, 10395–10407.

Cheng, K.; Li, X.; Xu, Y. E.; Dong, X. L.; and Sun, Y. 2022. PGE: Robust product graph embedding learning for error detection. *Proc. VLDB Endow.*, 15(6): 1288–1296.

Clouatre, L.; Trempe, P.; Zouaq, A.; and Chandar, S. 2021. MLMLM: Link prediction with mean likelihood masked language model. In *Findings of ACL*, 4321–4331.

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

Dong, J.; Zhang, Q.; Huang, X.; Tan, Q.; Zha, D.; and Zihao, Z. 2023. Active ensemble learning for knowledge graph error detection. In *WSDM*, 877–885.

Dong, X.; Gabrilovich, E.; Heitz, G.; Horn, W.; Lao, N.; Murphy, K.; Strohmann, T.; Sun, S.; and Zhang, W. 2014. Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In *KDD*, 601–610.

Guo, Q.; Zhuang, F.; Qin, C.; Zhu, H.; Xie, X.; Xiong, H.; and He, Q. 2022. A survey on knowledge graph-based recommender systems. *IEEE Trans. Knowl. Data Eng.*, 34(8): 3549–3568.

Jia, S.; Xiang, Y.; Chen, X.; and Wang, K. 2019. Triple trustworthiness measurement for knowledge graph. In *WWW*, 2865–2871.

Kenton, J. D. M.-W. C.; and Toutanova, L. K. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*, 4171–4186.

Lehmann, J.; Gerber, D.; Morsey, M.; and Ngomo, A.-C. N. 2012. Defacto-deep fact validation. In *ISWC*, 312–327.

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

Liu, Y.; Sun, Z.; Li, G.; and Hu, W. 2022. I know what you do not know: Knowledge graph embedding via co-distillation learning. In *CIKM*, 1329–1338.

Luo, X.; Sun, Z.; and Hu, W. 2022. μKG: A library for multi-source knowledge graph embeddings and applications. In *ISWC*, 610–627.

Lv, X.; Lin, Y.; Cao, Y.; Hou, L.; Li, J.; Liu, Z.; Li, P.; and Zhou, J. 2022. Do pre-trained models benefit knowledge graph completion? A reliable evaluation and a reasonable approach. In *Findings of ACL*, 3570–3581.

Nadkarni, R.; Wadden, D.; Beltagy, I.; Smith, N.; Hajishirzi, H.; and Hope, T. 2021. Scientific language models for biomedical knowledge base completion: An empirical study. In *AKBC*.

Paulheim, H.; and Bizer, C. 2014. Improving the quality of linked data using statistical distributions. *Int. J. Semant. Web Inf. Syst.*, 10(2): 63–86.

Saxena, A.; Tripathi, A.; and Talukdar, P. P. 2020. Improving multi-hop question answering over knowledge graphs using knowledge base embeddings. In *ACL*, 4498–4507.

Toutanova, K.; Chen, D.; Pantel, P.; Poon, H.; Choudhury, P.; and Gamon, M. 2015. Representing text for joint embedding of text and knowledge bases. In *EMNLP*, 1499–1509.

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

van den Oord, A.; Li, Y.; and Vinyals, O. 2018. Representation learning with contrastive predictive coding. *arXiv*, 1807.03748.

Wang, B.; Shen, T.; Long, G.; Zhou, T.; Wang, Y.; and Chang, Y. 2021. Structure-augmented text representation learning for efficient knowledge graph completion. In *WWW*, 1737–1748.

Wang, Y.; Ma, F.; and Gao, J. 2020. Efficient knowledge graph validation via cross-graph representation learning. In *CIKM*, 1595–1604.

Xie, R.; Liu, Z.; Lin, F.; and Lin, L. 2018. Does William Shakespeare really write Hamlet? Knowledge representation learning with confidence. In *AAAI*, 4954–4961.

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

Yang, C.; An, Z.; Cai, L.; and Xu, Y. 2022. Mutual contrastive learning for visual representation learning. In *AAAI*, 3045–3053.

Yao, L.; Mao, C.; and Luo, Y. 2019. KG-BERT: BERT for knowledge graph completion. *arXiv*, 1909.03193.

Zhang, Q.; Dong, J.; Duan, K.; Huang, X.; Liu, Y.; and Xu, L. 2022. Contrastive knowledge graph error detection. In *CIKM*, 2590–2599.

Zhao, Y.; and Liu, J. 2019. SCEF: A support-confidence-aware embedding framework for knowledge graph refinement. *arXiv*, 1902.06377.