# Knowledge Structure Driven Prototype Learning and Verification for Fact Checking

Shuai Wang, Wenji Mao*, Penghui Wei, Daniel D. Zeng

*State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China*

*School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing 100049, China*

**Abstract**

To inhibit the spread of rumorous information, fact checking aims at retrieving relevant evidence to verify the veracity of a given claim. Previous work on fact checking typically uses knowledge graphs (KGs) as external repositories and develop reasoning methods to retrieve evidence from KGs. Domain knowledge structure, including category hierarchy and attribute relationships, can be utilized as discriminative information to facilitate KG based learning and verification. However, in previous fact checking research, category hierarchy and attribute information was often scattered in a KG and treated as the ordinary triple facts in the learning process like other types of information, or was utilized in a limited way without the consideration of category hierarchy or the combination of category hierarchy with the learning process. Thus to better utilize category hierarchy and attribute relationships, in this paper, we propose an end-to-end knowledge structure driven prototype learning and verification method for fact checking. For improving intra-category compactness and inter-category sparation, we develop a hierarchical prototype learning technique that jointly learns a prototype for each sub-category to enhance entity embeddings and optimize embedding representations using high-level category. For further enhancing embedding learning, we propose a graph attention network to aggregate information from neighboring attribute nodes. We construct a real-world

*Corresponding author

dataset on food domain, and experimental results on the benchmark datasets and our domain dataset show the effectiveness of our method compared to both previous fact checking methods and representative KG reasoning methods.

*Keywords:* fact checking, knowledge structure, hierarchical prototype learning, relation enhancement, verification

## 1. Introduction

Internet and social media enable every individual to be a publisher, communicating true or false information instantly and globally. The spread of knowledge-based rumorous information causes severe consequences to individuals and society. False knowledge affects science progress and societal development, undermining trust in science and the capacity of individuals to make evidence-informed choices, including on life-or-death issues [1]. Among the false information on the Web, knowledge-based misinformation accounts for a large portion, based on the statistics reported by an authority website [1].

To inhibit the spread of knowledge-based rumors, considerable research efforts have been devoted to fact checking, which aims at retrieving relevant evidence to verify the truthfulness of a given claim. Previous methods on fact checking typically use KGs as external repositories and develop reasoning methods to retrieve evidence from KGs. These methods can be classified into two classes: path-based methods [2–5] and embedding-based methods [6, 7]. Path-based methods extract evidence from the paths between head and tail entity pairs, but they cannot always find effective paths to support verification due to the incompleteness problem in real-world KGs [8]. To induce the inner connections in KGs, embedding-based methods map the KG components (i.e. entities and relations) into a vector space for effective verification based on the semantic information, which can alleviate the incompleteness issue. However, due to the long-tailed distribution in real-world KGs [9], these methods often suffer from

[1] www.xinhuanet.com

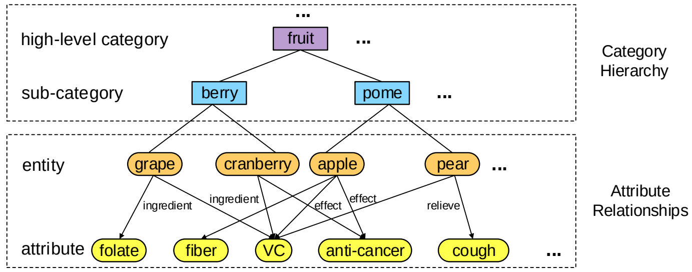



Figure 1: An illustrative example of domain knowledge structure.

the overfitting problem with insufficient training triples.

In order to reduce overfitting and enhance KG based learning and verification for fact checking, knowledge structure plays an important role. Figure 1 illustrates an example of domain knowledge structure, which includes category hierarchy and attribute relationships. The upper part in Figure 1 shows an example of category hierarchy, where *grape* and *apple* are entity nodes belonging to sub-categories *berry* and *pome* respectively, which belong to the high-level category *fruit*. This hierarchical structure of categories can be utilized as additional information to improve intra-category compactness and inter-category discrimination in KG based learning. In fact, when human fact-checkers examine an assertion, they would attempt to understand the generalized notion of the assertion by taking advantage of entity category [4]. The lower part in Figure 1 gives an example of attribute relationships, where entity nodes have attributes such as ingredients *fiber* and *folate* and effects *anti-cancer* and *cough-relief*. Leveraging the neighboring attribute relationships can enrich the semantic information of entities. The above knowledge structure can provide important semantic and discriminative information to facilitate KG based learning and verification.

However, in previous fact checking research, this information was often scattered in KGs and treated as the ordinary triple facts in the learning process, just the same as other types of information. Although several works on fact checking [4, 5] and KG reasoning [10] has incorporated categories into their methods, they either only utilized the limited category information (i.e. one level of categories) without the consideration of category hierarchy [4, 5], or they did not combine hierarchical category information with the learning process [10].

To make better use of category hierarchy, prototype learning (PL) is an excellent fit for acquiring discriminative category representation. Previously, PL was typically used to find representatives (i.e. prototypes) in the input space and then predict the class label based on its distance to the prototypes [11, 12]. Recent research combines PL with deep learning to develop deep prototype learning models that learn discriminative class prototypes for robust pattern classification [13–15]. Different from previous research that uses PL only for classification, our focus is on using category hierarchy as discriminative information for developing prototype based embedding learning. Thus we learn the prototypes for each category and take them as representatives to improve the semantic representation of entity embedding learning.

In this paper, we propose an end-to-end **K**nowledge **S**tructure driven **P**rototype **L**earning and **V**erification method (**KS-PLV**) for fact checking. To achieve intra-category compactness and inter-category discrimination in KG based learning, we propose a hierarchical prototype learning method, which learns prototypes for each sub-category and pulls entity embeddings closer to its corresponding prototypes, and by designing the loss function, further pulls closer the embedding clusters of sub-categories belonging to the same high-level category. We then propose a graph attention network, which aggregates the neighboring attribute nodes to enhance the semantic representations of entities. Finally, we design multiple loss functions to conduct KG based embedding learning and verification jointly for fact checking.

The main contributions of our work are as follows:

- We propose an end-to-end knowledge structure driven method for fact checking, which can effectively utilize category hierarchy and attribute relationships in KG based learning and verification.
- We develop the first hierarchical prototype learning method to improve the robustness of learning entity embeddings at both entity level and category level.
- We construct a real-world dataset on food domain, and experimental re-

sults on two benchmark datasets and our domain dataset show the effectiveness of our method compared to previous fact checking methods and representative KG reasoning methods.

The rest of the paper is organized as follows. Section 2 introduces the related work on KG reasoning and fact checking. Section 3 describes our proposed method in detail. In Section 4, we conduct intensive empirical studies to evaluate our work and analyze the experimental results. Section 5 concludes the paper and raises some future work.

## 2. Related Work

Fact checking can be broadly viewed as a reasoning task, typically using KGs as external repositories. In this section, we first review the related research on KG reasoning, and then review the related work on fact checking, focusing on KG based methods.

### *2.1. KG Reasoning*

The fundamental purpose of KG reasoning task is to automatically deduce and add the missing knowledge to KGs (i.e. knowledge graph completion). For instance, given a query $(h, r, ?)$, where $h$ is the head entity and $r$ is the relation, the goal of knowledge graph completion for the tail entity is to perform an efficient search over $\mathcal{G}$ and collect the set of possible answers $T = \{t_1, ..., t_n\}$, s.t. $(h, r, t_i) \notin \mathcal{G}, i \in [1, n]$. Currently, the main approach for KG reasoning is link prediction, which aims at predicting the missing link between two entities on KGs.

Link prediction can be classified as path-based methods and KG embedding methods. Path-based methods aim at finding new triples through the inference over the existing paths in KG [9, 16–25], while KG embedding methods aim to reason about the plausibility of triples by matching the embeddings of entities and relations in vector spaces. For path-based methods, a conventional method is to utilize Inductive Logic Programming (ILP) [16–19] for inferring over KGs. However, ILP based methods are easily overwhelmed by amount of data. To develop a robust method that is scalable to large knowledge graph, Lao et al. [20, 21] presented a simple model PRA that learned to infer relations by combining the results of different random walks through KG. To make complex inferences, Das et al. [22] conducted multi-step inference of symbolic logical reasoning using recurrent neural networks. Recent work employs deep reinforcement learning models, such as DeepPath [23] and MINERVA [24], to explore predictive paths in KG based on the input query. To further advance MINERVA, Lin et al. [25] proposed the Multi-hop method, which adopted a pre-trained embedding model to estimate unobserved facts for improving the quality of rewards in reinforcement learning. Moreover, to overcome the long-tail issue in real-world KGs, Lv et al. [9] proposed Meta-KGR, which adopted meta learning to learn effective meta parameters from high-frequency relations that could quickly adapt to few-shot relations. Their model achieved the state-of-the-art performance among path-based methods.

KG embedding methods can be roughly categorized into translational distance models and semantic matching models. Translational distance models rely on distance-based scoring functions. The main stream work is TransE [26] (which defines its scoring function as the distance between $h + r$ and $t$) and its extensions, including TranH [27], TransR [28], TransD [29], TransM [30], TransF [31], etc. Additional information can be incorporated to improve KG embeddings. Xie et al. [10] proposed the TKRL model that projected embeddings to their corresponding category spaces using the type-specific projection matrices. Although TKRL did not combine hierarchical category information with the learning process, it achieved relatively good performance among translational distance based models. Other translational distance models are based on Gaussian distribution including KG2E [32] and TransG [33].

Semantic matching models match the latent semantics of entities and relations in vector spaces. Early work relies on similarity-based functions (i.e., similarity based models), such as RESCAL [34] and NTN [35]. RESCAL [34] represented relations as matrices, which modeled pairwise interactions between two entities, and its subsequent work extended RESCAL to efficiently capture relational semantics [36], improve the capability of complex embeddings [37, 38] or model the analogical properties of entites and relations [39]. For example, DistMult [36] utilized matrix multiplication to model the compositional relations, which performed the best among RESCAL and its other variants. However, similarity based models focus on shallow, fast models with less expressive features. Recent work adopts deep learning models to learn more distinctive features automatically. Dettmers et al. [40] proposed a deep neural model ConvE that used two-dimensional convolution over embeddings with multiple layers. Schlichtkrull et al. [41] applied GCN to incorporate connectivity structure and model the relational data (i.e., R-GCN). As it is beneficial to compose embeddings from query-relevant nodes in KG, Bansal et al. [42] proposed the A2N model, which used a bi-linear attention on the graph neighbors to generate dynamic embeddings based on the query. A2N outperformed ConvE and R-GCN, and achieved the state-of-the-art performance on two benchmark datasets in KG reasoning task.

As the main purpose of KG reasoning is for knowledge graph completion, the focus of these previous methods is on ranking candidate entities and relations by their scores. Another line of work on KG reasoning is triple classification [6], the purpose of which is similar to that of fact checking. We shall introduce this work together with fact checking methods in the next section.

### *2.2.* ***Fact Checking***

For fact checking methods, there are two means to collect evidence from external sources. One is to search from *Web texts* such as WikiPedia and then develop reasoning methods to find relevant evidence [43–45]. The other commonly used means makes use of *knowledge repositories* as external sources, typically using KGs, which contain plentiful high quality facts.

Similar to KG reasoning, fact checking also consists of path-based [2–5] and embedding-based [6, 7] methods. Path-based fact checking methods mine the paths between a head entity $h$ and a tail entity $t$ to predict whether there is a relation $h \xrightarrow{r} t$ in KG. Ciampaglia et al. [2] proposed an original method to explore node connectivity in KG via finding the single shortest path between head and tail entities. To utilize richer information, Shiralkar et al. [3] proposed a method (called Knowledge Stream) to extend the work in [2], which employed a flow network to find multiple paths in KG. Knowledge Steam computed the truth score of a claim triple by considering all the paths between head and tail entities. To effectively discover discriminative paths, Shi et al. [4] proposed Predpath, which used head and tail entity categories as anchors and defined the mined rules for path extraction. Predpath achieved the state-of-the-art performance on several real-world and synthetic datasets for fact checking. To better utilize category and relation information, Fionda et al. [5] built a schema graph to generate candidate evidence patterns for fact checking via leveraging domain of predicates and entity categories along with various RDFS inference rules and optimization techniques for KG loading in memory. However, both work [4, 5] only utilized the limited category information, that is, they used one level of categories without the consideration of category hierarchy.

In real world KGs, connections between entities are usually sparse. Due to the incompleteness of real world KGs, path-based methods can not always find valid paths for fact checking. To induce the inner connections in KGs, embedding-based fact checking methods map entities and relations into continuous vector spaces to calculate the truthfulness of the claim triple, which can alleviate the incompleteness issue. The representative embedding based fact checking methods include TEKE [6] and Dual TransE [7]. As embedding based methods are brittle in 1-to-N relations, to solve this problem, Dong et al. [6] proposed a region-based approach TEKE that extended the embeddings into regions, in which tail regions of 1-to-N relations were located in the head region. Although TEKE performs well on the constructed KGs that contain only several types of relations, it is not salable to larger KGs typically with more relation types for fact checking. To make better distinction between true facts and false ones, Pan et al. [7] constructed two KGs based on true and false triples separately, and developed Dual TransE to evaluate the claim triple by comparing the max bias of these two KGs. Dual TransE achieved the best performance on a large scale real-world dataset in fact checking task.

To develop robust end-to-end method for fact checking task, in this paper, we take the advantage of knowledge structure, including category hierarchy and attribute relationship, to enhance KG based learning and verification. Unlike previous methods that utilize knowledge structure in a limited way, we focus on developing a hierarchical prototype learning technique with graph attention network to effectively incorporate knowledge structure as distinctive information for fact checking. We also conduct experimental studies to validate our proposed method.

## 3. Proposed Method

Given an unverified claim triple $(h, r, t)$ composed of a head entity $h$, a relation $r$ and a tail entity $t$, the goal of a fact checker $f(\cdot)$ is to compute a truth value $f(h, r, t)$ for the triple, with the help of a corresponding knowledge graph (KG) $\mathcal{G}$ that contains a large number of triple facts. To verify the truthfulness of claim triples, we expect that the fact checking model $f(\cdot)$ can produce a higher score for a true claim than that for a false one.

Figure 2 gives an overview of our method KS-PLV, which consists of three modules: (1) a *Hierarchical Prototype Learning* (HPL) module for improving entity representation learning, which learns the prototypes for each entity category in KG and pull entity embeddings to their corresponding prototypes in the vector space at both entity-level and category-level; (2) a *Relation Enhancement* module for relation-based embedding enhancement, which uses attribute relationship information to augment entity embeddings with a graph attention network (named *Attribute-GNN*); (3) a *Fact Verification* module for scoring the truth values of triples by semantic matching. We detail our KS-PLV method in the following sections.

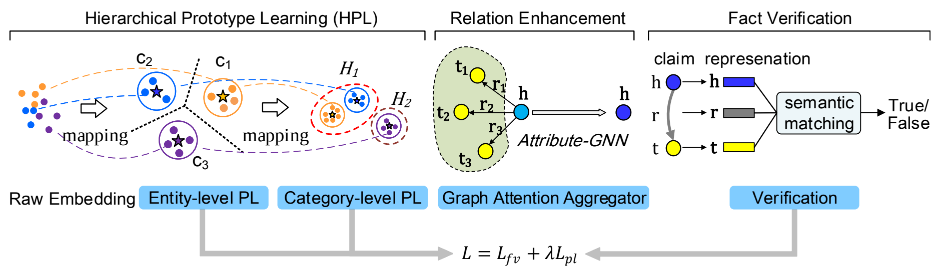



Figure 2: Overview of our proposed end-to-end knowledge structure driven prototype learning and verification method KS-PLV (Here the sub-categories $c_1$ and $c_2$ belong to the high-level category $H_1$, and $c_3$ belongs to $H_2$. Stars denote the learned prototypes of corresponding categories).

### 3.1. Hierarchical Prototype Learning with Category Hierarchy

To make better use of the rich information in knowledge structure, we encode the category hierarchy into the entity embedding learning process. An example category hierarchy of entities is shown in Figure 1. Each high-level category contains a number of sub-categories, and each entity belongs to a sub-category[2].

Based on this hierarchical category structure, we propose a hierarchical prototype learning method for entity embedding learning, which is composed of entity-level and category-level prototype learning (PL).

#### 3.1.1. Entity-level Prototype Learning

Entity-level prototype learning leverages entity's sub-category information to learn entity embeddings. Intuitively, if two entities belong to the same sub-category, their embeddings should be close to each other in the vector space.

Based on this consideration, we build a learnable *prototype vector* for each sub-category to perform entity-level PL. We only consider leaf sub-categories (a leaf sub-category is the lowest-level sub-category of entities that has no sub-categories under it), and a prototype serves as the representative of the corresponding leaf sub-category. We utilize these prototypes to regularize the entity

[2] In a large-scale KG, due to the existence of some general categories, an entity may belong to more than one sub-category. To effectively incorporate the distinctive category information for improving entity embedding learning, we choose the smallest sub-category of an entity as its specific sub-category in our work.

representation learning process via pulling the entity embeddings closer to their prototypes in the vector space. The prototype vectors are optimized together with other learnable parameters in our KS-PLV method.

Specifically, given a KG $\mathcal{G}$, its entity set $\mathcal{E}$ and an entity $e_j^{(i)} \in \mathcal{E}$, let $\mathcal{H}_l$ denote the set of leaf sub-category in $\mathcal{G}$, $c_i \in \mathcal{H}_l$ denote the leaf sub-category $e_j^{(i)}$ belongs to, and $\boldsymbol{p}^{(i)}$ denote the prototype vector of $c_i$. For entity $e_j^{(i)}$, we pull its embedding $\boldsymbol{e}_j^{(i)}$ closer to its leaf sub-category's prototype vector $\boldsymbol{p}^{(i)}$ by minimizing their 2-norm distance, where the prototype serves as the constraint of embedding position in the vector space:

$$d(e_j^{(i)}, c_i) = \|\boldsymbol{e}_j^{(i)} - \boldsymbol{p}^{(i)}\|_2 \tag{1}$$

Further, to improve the inter-category discrimination of entity representations, we incorporate inter-category constraints to avoid the overlapping between the embeddings of different sub-categories. Given entity embedding $\boldsymbol{e}_j^{(i)}$, we first find the closest prototype $\boldsymbol{p}_{\text{closest}}$ from other leaf sub-categories $\mathcal{H}_l - \{c_i\}$: $\forall c_j \in \mathcal{H}_l - \{c_i\}, c_{\text{closest}} \in \mathcal{H}_l - \{c_i\}$, we have

$$\|\boldsymbol{e}_j^{(i)} - \boldsymbol{p}_{\text{closest}}\|_2 \leq \|\boldsymbol{e}_j^{(i)} - \boldsymbol{p}^{(i)}\|_2 \tag{2}$$

Thus sub-category $c_{\text{closest}}$ can be regarded as the most competitive to $c_i$ for entity $e_j^{(i)}$. We then build the second objective, which maximizes the 2-norm distance between $\boldsymbol{e}_j^{(i)}$ and the closest prototype $\boldsymbol{p}_{\text{closest}}$:

$$d(e_j^{(i)}, c_{\text{closest}}) = \|\boldsymbol{e}_j^{(i)} - \boldsymbol{p}_{\text{closest}}\|_2 \tag{3}$$

By combining the above two objectives, the training objective for entity-level PL is to minimize the following function:

$$\mathcal{L}_{pl-entity} = \sum_{e_j^{(i)} \in \mathcal{E}} \left( \|\boldsymbol{e}_j^{(i)} - \boldsymbol{p}^{(i)}\|_2^2 - \|\boldsymbol{e}_j^{(i)} - \boldsymbol{p}_{\text{closest}}\|_2^2 \right). \tag{4}$$

#### 3.1.2. ***Category-level Prototype Learning***

Category-level prototype learning further leverages high-level category information of sub-categories to improve entity representation learning. Intuitively, if two sub-categories belong to the same high-level category, their corresponding prototypes should be relatively close to each other in the vector space. To better represent a sub-category and model the distribution of its entities, we use *multivariate Gaussian distribution* (including mean and covariance) as the representation of the sub-category (i.e., the representative prototype of this sub-category).

Specifically, given two sub-categories $c_U$ and $c_V$ that belong to the same high-level category, their corresponding Gaussian distributions are denoted as prototypes $p_U = \mathcal{N}(\boldsymbol{\mu}_U, \Sigma_U)$ and $p_V = \mathcal{N}(\boldsymbol{\mu}_V, \Sigma_V)$. We pull the two distributions closer to each other by minimizing their 2-Wasserstein distance (Note that Wasserstein distance can represent the distance when there is no overlapping between two distributions while KL-divergence fails in this case):

$$\begin{aligned} W(p_U, p_V) = &\|\boldsymbol{\mu}_U - \boldsymbol{\mu}_V\|_2^2 \\ &+ \mathrm{Tr}(\Sigma_U + \Sigma_V - 2(\Sigma_U^{\frac{1}{2}} \Sigma_V \Sigma_U^{\frac{1}{2}})^{\frac{1}{2}}) \end{aligned} \tag{5}$$

where the Tr is the trace of the matrix, $\mu_i$ and $\Sigma_i$ are the unbiased estimation of mean and covariance respectively (let $|c_i|$ denote the number of entities belonging to the sub-category $c_i$):

$$\boldsymbol{\mu}_i = \frac{1}{|c_i|} \sum_{e_j^{(i)} \in c_i} \boldsymbol{e}_j^{(i)}, \tag{6}$$

$$\Sigma_i = \frac{1}{|c_i| - 1} \sum_{e_j^{(i)} \in c_i} (\boldsymbol{e}_j^{(i)} - \boldsymbol{\mu}_i)(\boldsymbol{e}_j^{(i)} - \boldsymbol{\mu}_i)^\top. \tag{7}$$

As the covariance is a symmetric matrix, which satisfies $\Sigma_U \Sigma_V = \Sigma_V \Sigma_U$.

Equation 5 can be simplified as:

$$W(p_U, p_V) = \|\boldsymbol{\mu}_U - \boldsymbol{\mu}_V\|_2^2 + \|\Sigma_U^{\frac{1}{2}} - \Sigma_V^{\frac{1}{2}}\|_2^2 \tag{8}$$

Further, we also minimize the 2-form distance between their prototypes $\boldsymbol{p}_U$ and $\boldsymbol{p}_V$ as a constraint. By combining the two objectives, the training objective for category-level PL is to minimize the following function:

$$\mathcal{L}_{pl-category} = \sum_{c_U, c_V \in \mathcal{L}_l} \Big( \|\boldsymbol{p}_U - \boldsymbol{p}_V\|_2^2 + W(p_U, p_V) \Big). \tag{9}$$

In the intermediate levels of category hierarchy, each node is both the sub-category of its upper-level parents and the high-level category of its lower-level children. Thus the above category-level PL process is recursive and can be adapted to multi-level category hierarchy.

Finally, by weighting the sum of entity-level and category-level losses, the training objective for hierarchical prototype learning is to minimize $\mathcal{L}_{pl}$:

$$\mathcal{L}_{hpl} = \mathcal{L}_{pl-entity} + \gamma \mathcal{L}_{pl-category} \tag{10}$$

where $\gamma$ is the trade-off parameter.

### *3.2.* ***Relation Enhancement with Attribute-GNN***

Relation based enhancement leverages attribute relationship information to further enhance embedding learning for fact checking. Intuitively, an entity's attributes (structurally embodied in its neighboring entities and its corresponding relations) can enrich the semantic representations of entities and provide relevant evidence for fact verification. To model this local graph structure, we encode neighboring attributes into entity embeddings via a relation based graph attention network *Attribute-GNN*. We then utilize these enhanced embedding representations to help better predict the truth value of the claim triple.

Specifically, given a head entity embedding $\boldsymbol{h}$ and one tail entity embedding $\boldsymbol{t}_j$ with its relation embedding $\boldsymbol{r}_j$ (i.e., $(h, r_j, t_j)$ is a triple fact in $\mathcal{G}$), a GNN layer aggregates the head entity's neighboring information with attention mechanism to learn an updated embedding for the head entity:

$$
\begin{aligned}
s_j &= \mathbf{w}^\top \mathrm{ReLU}\left([\mathbf{W}\boldsymbol{h}; \mathbf{W}(\boldsymbol{t}_j \odot \boldsymbol{r}_j)]\right) \\
\alpha_j &= \mathrm{softmax}(s_j) \\
\hat{\boldsymbol{h}} &= \tanh\left(\sum_{(h,r_j,t_j)\in\mathcal{N}(h)} \alpha_j \mathbf{W}\boldsymbol{t}_j\right) \\
\boldsymbol{h} &\leftarrow \mathbf{W}_h[\hat{\boldsymbol{h}}; \boldsymbol{h}]
\end{aligned}
\tag{11}
$$

The first two equations compute the attention score $\alpha_j$ for the $j$-th neighbor (i.e., the tail entity $t_j$ with the relation $r_j$). We integrate the relation information into the tail entity via element-wise multiplication on their embeddings. Thus the third equation aggregates all the one-hop neighbors to update the head entity representation, where $\mathcal{N}(h)$ denotes all the triples in which $h$ is the head entity. Finally, in the fourth equation, we concatenate the aggregated embedding $\hat{\boldsymbol{h}}$ and the original embedding $\boldsymbol{h}$, and then project the concatenated vector into the initial dimension with $\mathbf{W}_h$ to obtain the ultimate entity representation.

### *3.3.* ***Fact Verification***

We compute the score of a triple fact $(h, r, t)$ by semantic matching, which measures the plausibility of the fact via matching the latent semantics of the two entities with their relation. Our prototype based learning and verification described above has incorporated rich knowledge structure information on category hierarchy and attribute relationship to effectively support this semantic matching process for fact verification.

Specifically, we adopt the following score function [36]:

$$
f(h, r, t) = \boldsymbol{h}^\top \mathrm{diag}(\boldsymbol{r})\boldsymbol{t} \tag{12}
$$

where $\boldsymbol{h}$, $\boldsymbol{r}$ and $\boldsymbol{t}$ denote the embeddings of the head entity $h$, relation $r$ and tail entity $t$ respectively, and $\mathrm{diag}(\boldsymbol{r})$ is a diagonal matrix formed by the elements in $\boldsymbol{r}$.

The objective of fact verification is to minimize the regularized logistic loss, which encourages $f(\cdot)$ to produce a higher score for a true triple than that for a false one:

$$\mathcal{L}_{fv} = \sum_{(h,r,t)\in\Delta} \log(1+\exp(-f(h,r,t))) + \sum_{(h',r',t')\in\Delta'} \log(1+\exp(f(h',r',t'))) \tag{13}$$

where $\Delta$ and $\Delta'$ are the set of correct triples and that of incorrect triples, respectively.

### *3.4.* ***Overall Objective Function of KS-PLV***

Our KS-PLV method can be optimized in an end-to-end fashion via minimizing the overall loss $\mathcal{L}$ defined as follows:

$$\mathcal{L} = \mathcal{L}_{fv} + \lambda\mathcal{L}_{hpl} \tag{14}$$

where $\lambda$ is the trade-off parameter.

## 4. Experiments

In this section, we validate our KS-PLV method by comparing it with the previous fact checking methods and representative KG reasoning methods in the related work. We also analyze the experimental results and discuss on the related issues.

### *4.1.* ***Experimental Setup***

#### *4.1.1.* ***Datasets and Hyperparameters***

***Datasets.*** We conduct our experiments on three datasets. To demonstrate the advantage of our method, we construct a real-world knowledge graph on food domain (*FOOD*). To maintain the completeness of category hierarchy, we collect the category information for each entity and sub-category from the Chinautri website [3]. We first collect the ingredients of food from this website. To collect the common effects of food, we extract food names and their effects originally presented as structured descriptions from another popular food website, Meishichina[4]. The extracted food effects have two forms, verb-noun phrase (e.g., *reduce cholesterol*) or simple noun phrase (e.g., *antiobesity*). For each verb-noun phrase, we take its verb and noun as relation and tail entity respectively to construct a triple (e.g., (*broccoli, reduce, cholesterol*)). For each noun phrase, we treat it as tail entity and add the relation *effect* to construct a triple. By this means, we build the KG in food domain with 4192 entities and 86 types of relations[5].

We also use two benchmark KG datasets *FB15K* [26] and *FB15K-237*. *FB15K* is a dense graph extracted from *Freebase*, with each entity and relation in it having 100+ associative triples in *Freebase*. *FB15K-237* is constructed by deleting all the inverse triples in *FB15K* to increase the data sparsity of the graph, so that the reasoning task becomes more challenging. Currently *FB15K* and *FB15K-237* are the most commonly used KG datasets. We utilize the category hierarchy of entities and sub-categories for them collected by [10]. Table 1 gives the statistics of the three KG datasets.

We construct the negative triples as false facts for all the datasets. As the start point, we adopt the approach typically used by most previous KG based reasoning methods (e.g. [6–10, 16–42]), which replaces a head or tail entity of a positive fact in a KG with another randomly selected entity (i.e., random negative sampling approach). Moreover, to make the fact verification task harder, following [10, 35], we only allow entities of the constructed negative triples to appear in the positions where they have occurred in the KGs. For example, given a correct triple (*pear, moisten, lung*), an allowable negative example is (*chili, moisten, lung*) or (*pear, moisten, stomach*). Besides, to better alleviate

[3]National Institute for Nutrition and Health of Chinese CDC: www.chinanutri.cn
[4]The Fine Food in China: www.meishichina.com
[5]https://github.com/Wangshuaiia/fact_checking

| Dataset | #entity | #relation type | #train | #valid | #test |
|---|---|---|---|---|---|
| FOOD | 4,192 | 86 | 26,767 | 1,986 | 1,080 |
| FB15K-237 | 14,505 | 237 | 272,115 | 20,000 | 20,000 |
| FB15K | 14,951 | 1,345 | 483,142 | 50,000 | 59,071 |

Table 1: Statistics of the three KG datasets.

the errors in constructing negative triples, we restrict the random sampling process by selecting the newly replaced entity whose category is different from that of the original entity. Intuitively, the farther the distance between the categories of two entities in the category hierarchy, the smaller the possibility to construct erroneous negative triples by the above random sampling approach.

***Hyperparameters.*** For the benchmark datasets FB15K and FB15K-237, we set the dimension of embeddings to 100, learning rate to 0.01 and batch size to 128, the configurations widely used in many previous models (e.g. [26–31, 36, 40]). For our constructed dataset FOOD, we also set the learning rate to 0.01 and batch size to 128, and the dimension of embeddings is set to 30 so as to balance between overfitting and the powerfulness of representations. The trade-off parameters $\lambda$ and $\gamma$ are set to 0.1 and 0.01 respectively, based on the observation that our model performs best on these values of $\lambda$ and $\gamma$. We adopt the AdaGrad optimizer. We shall provide the detailed analyses of different values of $\lambda$ and $\gamma$ on model performance in Subsection 4.3.1.

#### *4.1.2.* ***Comparative Methods***

We compare our method with the existing KG based fact checking methods. We also compare our method with the representative KG reasoning methods. Though KG reasoning methods are designed to rank candidate entities or relations for knowledge graph completion, but not for the fact checking task, we adjust them to the verification of unseen triples based on their ranking scores.

The existing fact checking methods we compared with include:

(1) **Knowledge Stream** [3] regards fact checking as a network-flow problem

and computes the truth score by summing the net flows of paths between head and tail entities.

(2) **PredPath** [4] uses head and tail entity categories as anchors to effectively find discriminative paths and classifies the selected paths using logistic regression.

(3) **Dual TransE** [7] constructs two KGs based on true triples and false triples, and evaluates the claim triple by comparing the max bias of these two KGs.

The representative KG reasoning methods we compared with include:

(1) **DistMult** [36] is a classical semantic matching method that models compositional relations using matrix multiplication and measures the plausibility of facts using a bi-linear score function.

(2) **TKRL** [10] is a TransE based model that projects entity embeddings to their corresponding category spaces via the type-specific projection matrices.

(3) **MINERVA** [24] uses reinforcement learning (RL) to efficiently search the graph by sequentially extending the inference path based on the input query.

(4) **Multi-Hop** [25] improves the performance of RL by adopting a pre-trained embedding model to ensure the quality of rewards, and designs action dropout technique to find discriminative paths.

(5) **Meta-KGR** [9] is a RL based model, which employs meta-learning to solve the problem of few-shot relations.

(6) **A2N** [42] adaptively composes neighbors with bi-linear attention to generate dynamic embeddings based on the input query.

Among the above fact checking methods, Knowledge Stream and PredPath are path-based methods, which extract evidence from the paths between head and tail entity pairs. Compared with Knowledge Stream, PredPath additionally utilizes entity type of head and tail entities to find discriminative paths. Dual TransE is an embedding-based method, which develops a TransE-based model to predicate the truth score of the input triple.

Among the above KG reasoning methods, MINERVA, Multi-Hop and Meta-KGR are path-based methods and use reinforcement learning as the reasoning framework. Multi-Hop improves MINERVA by shaping its reward function. On this basis, Meta-KGR adds meta-learning technique to learn a better representation for the few-shot relation [9]. DistMult, TKRL and A2N are embedding based methods. Compared with the classical method DistMult, TKRL uses additional entity type information and A2N composes neighbors with bi-linear attention.

For the implementation of methods Knowledge Stream, PredPath, MINERVA, Multi-Hop and Meta-KGR above, we use the codes published in the corresponding papers and tune the hyperparameters based on what are reported in the papers to get the best results. We reimplement the other compared methods and tune the hyperparameters in similar way. As each method produces a score for the claim triple, to verify its truthfulness, we use an automatically generated threshold that is derived by maximizing the accuracy of the method on validation set. We use accuracy and $F_1$ as the evaluation metrics.

## *4.2.* ***Experimental Results***

To validate our KS-PLV method, we conduct experiments on fact checking by comparing with the related methods and evaluating each component of our method. We provide the experimental results and analyses in this section.

### *4.2.1.* ***Comparison with Related Methods***

Table 2 summarizes the experimental results on fact checking by different methods. We can see from the table that our KS-PLV method outperforms all the comparative methods on the three datasets. Reinforcement learning based methods, including MINERVA, Multi-Hop and Meta-KGR, outperform the conventional path-based methods Knowledge Stream and PredPath. One reason for this is that RL based methods are generally more efficient in finding the discriminative paths. Among embedding based methods, A2N outperforms the early methods DistMult and TKRL. As A2N and our KS-PLV use neighbor-

| Method | FB15K | | FB15K-237 | | FOOD | |
|---|---|---|---|---|---|---|
| | Acc. | $F_1$ | Acc. | $F_1$ | Acc. | $F_1$ |
| Knowledge Stream | 0.835 | 0.829 | 0.801 | 0.729 | 0.725 | 0.734 |
| PredPath | 0.813 | 0.806 | 0.802 | 0.806 | 0.728 | 0.746 |
| Dual TransE | 0.912 | 0.903 | 0.874 | 0.875 | 0.764 | 0.726 |
| DistMult | 0.895 | 0.884 | 0.878 | 0.879 | 0.750 | 0.753 |
| TKRL | 0.948 | 0.936 | 0.854 | 0.836 | 0.768 | 0.723 |
| MINERVA | 0.903 | 0.883 | 0.886 | 0.873 | 0.754 | 0.747 |
| Multi-Hop | 0.950 | 0.913 | 0.902 | 0.893 | 0.781 | 0.776 |
| Meta-KGR | 0.933 | 0.901 | 0.907 | 0.899 | 0.783 | 0.784 |
| A2N | 0.946 | 0.937 | 0.912 | 0.908 | 0.806 | 0.807 |
| KS-PLV | **0.954** | **0.941** | **0.931**$^*$ | **0.925**$^*$ | **0.846**$^*$ | **0.838**$^*$ |

Table 2: Experimental results on fact checking by different methods (Among these compared models, Dual TransE and A2N are the SOTA methods for fact checking and KG reasoning respectively. Here "$*$" denotes that our KS-PLV significantly outperforms the other methods).

ing attribute relations to supply discriminative information, these two methods achieve obvious performance gains than all the path-based methods as well as early embedding-based methods. Compared to A2N, KS-PLV takes full advantage of category hierarchy of entities to further improve the performance. Although PredPath and TKRL also utilize the category information, PredPath only uses one level of categories and ignores the category hierarchy. Compared with TKRL, which learns a projecting matrix for each category, our KS-PLV benefits from hierarchical prototype learning to learn the robust entity representations. This indicates that incorporating category hierarchy and attribute relationships is indeed beneficial for KG-based fact verification.

It can be also seen from the table that for the dense knowledge graph *FB15K*, all the methods get fairly good results. This is consistent with the previous study on the impact of the KG sparsity on the model performance [9]. In a dense KG, it is relatively easy to learn generalized embeddings or find effective paths, while in a sparse KGs, since a large portion of entities have few relations with other entities, there are not sufficient triples to train the general representations

| Method | FB15K | | FB15K-237 | | FOOD | |
|---|---|---|---|---|---|---|
| | Acc. | $F_1$ | Acc. | $F_1$ | Acc. | $F_1$ |
| KS-PLV (full model) | 0.954 | 0.941 | 0.931 | 0.925 | 0.846 | 0.838 |
| −HPL | 0.939 | 0.931 | 0.914 | 0.904 | 0.811 | 0.803 |
| −Attribute-GNN | 0.920 | 0.907 | 0.908 | 0.899 | 0.765 | 0.786 |
| −Attribute-GNN −HPL | 0.895 | 0.884 | 0.878 | 0.879 | 0.750 | 0.753 |
| −Attribute-GNN +A2N | 0.942 | 0.936 | 0.921 | 0.910 | 0.816 | 0.807 |

Table 3: Experimental results of the ablation study.

for these entities. Since *FB15K-237* is sparser than *FB15K*, the performances of all the methods decline on *FB15K-237*. On *FB15K-237*, KS-PLV obtains 1.9% and 1.7% improvement on accuracy and $F_1$ respectively compared to the second best-performing method (i.e., A2N). The real-world dataset *FOOD* is the sparsest among the three datasets and thus makes the fact checking task on it more challenging. Nonetheless, our KS-PLV method can still get reasonably good results on *FOOD* and outperforms the other methods by a large margin. This indicates that KS-PLV is a robust end-to-end fact checking method for the sparse KG. In general, the experimental results verify the effectiveness of our method.

#### *4.2.2.* ***Ablation Study***

We conduct an ablation study to verify the effectiveness of each component in our method, by constructing four variations of KS-PLV:

- **−HPL**: excludes the *HPL* module (i.e., corresponding loss function $\mathcal{L}_{hpl}$) from KS-PLV.
- **−Attribute-GNN**: excludes the *Attribute-GNN* module from KS-PLV.
- **−Attribute-GNN −HPL**: excludes both *HPL* and *Attribute-GNN* modules from KS-PLV.
- **−Attribute-GNN +A2N**: replaces *Attribute-GNN* to *A2N* [42] in KS-PLV.

Table 3 gives the experimental results of the ablation study. We can see

| Level | FB15K | | FB15K-237 | | FOOD | |
|---|---|---|---|---|---|---|
| | Acc. | $F_1$ | Acc. | $F_1$ | Acc. | $F_1$ |
| Raw Embedding (w/o PL) | 0.939 | 0.931 | 0.914 | 0.904 | 0.811 | 0.803 |
| Entity-Level PL | 0.949 | 0.937 | 0.923 | 0.911 | 0.834 | 0.822 |
| Category-Level PL | 0.953 | 0.940 | 0.928 | 0.920 | 0.846 | 0.838 |

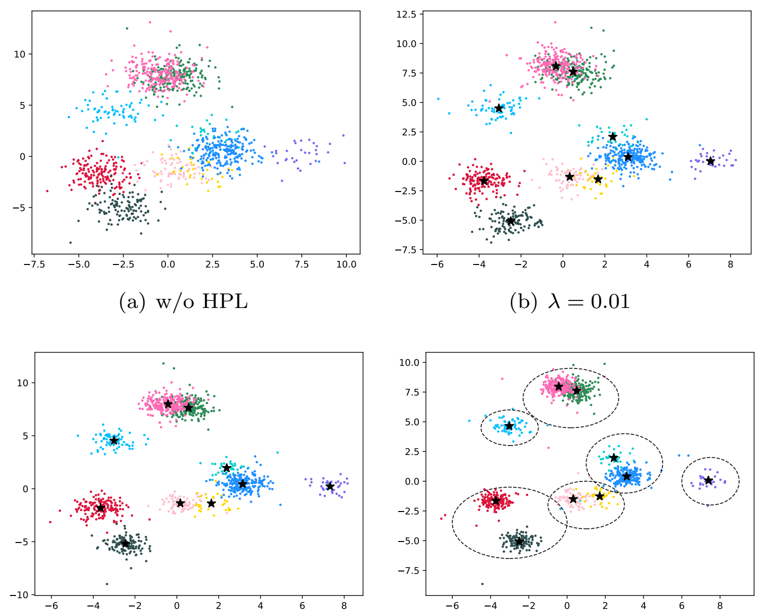
Table 4: Experimental results of entity-level and category-level prototype learning.

from the table that excluding *HPL* or *Attribute-GNN* from KS-PLV will cause significant performance drop. By comparing the results of "−Attribute-GNN" and "−HPL −Attribute-GNN", we can clearly see the performance gained by *HPL*. To further compare the results of "−Attribute-GNN +A2N" with those of the full model, we can also see that by aggregating the information of neighbors to improve the performance, our *Attribute-GNN* is obviously more effective than A2N's bi-linear attention model.

To further evaluate *HPL* at different levels, we conduct an additional experiment for entity-level and category-level prototype learning. Table 4 shows the experimental results. For category-level PL, we use the 4-layer category hierarchy for *FB15K* and *FB15K-237*, as the category hierarchy in them (with 92% of all the entities) is within 4 layers. The category hierarchy in *FOOD* has 3 layers. From the table, we can see that both entity-level PL and category-level PL can steadily increase the performances of fact checking. Compared to the performances gained by entity-level and category-level PL on *FB15K* and *FB15K-237*, the performance gains are much greater on *FOOD*, whose graph is the sparsest among the three datasets. The experimental results on the ablation study further verify the usefulness of each component in our method.

### *4.3.* ***Further Analyses***

We further analyze the learning and verification effect of our method in this section. As *HPL* is the major component in our method, we first analyze its learning properties. We then discuss the effect of our method on fact checking



(a) w/o HPL (b) $\lambda = 0.01$

(c) $\lambda = 0.1$ (d) $\lambda = 1$

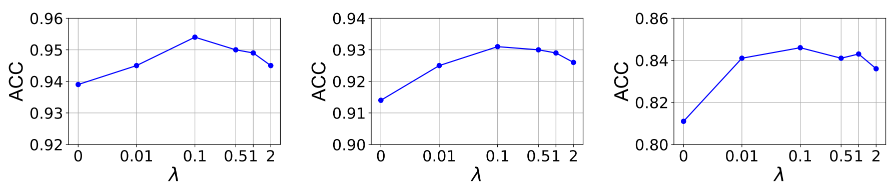
Figure 3: The learned representations of ten sub-categories of entities in *FOOD* (The colors represent different sub-categories, the black stars represent the corresponding prototypes of each sub-category and the dashed circles in (d) represent the high-level categories. Here $\lambda$ is the trade-off parameter in Equation 14).

in KGs with long tail.

#### *4.3.1.* ***Effect of Hierarchical Prototype Learning***

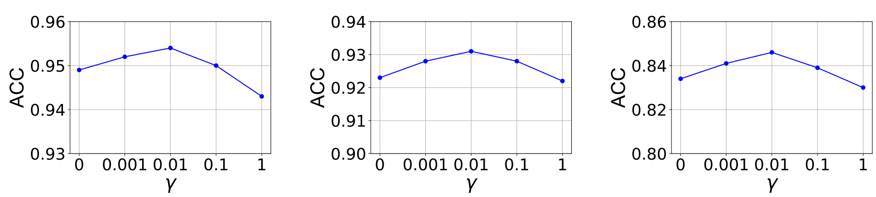
To analyze the impact of *HPL* on embedding learning, we choose ten largest sub-categories of entities in *FOOD* to visualize their embeddings with t-SNE. Figure 3 (a) shows that without *HPL*, the embeddings of entities in different categories tend to overlap with each other. In Figure 3 (b) and (c), with the increase of $\lambda$ in Equation 14, the learned intra-category embeddings become more and more compact and some overlapping inter-category embeddings are separated. In Figure 3(d), the distributions of the sub-categories in the same high-level category are close (illustrated by the dashed circles). This demonstrates that the representations learned by KS-PLV have the properties of intra-category compactness and inter-category separation.

To further analyze the impact of *HPL* on fact checking, we first vary the values of $\lambda$ from 0 to 2. Figure 4 shows the fact checking results on the three datasets. With the growth of $\lambda$, the performances gradually go up at first and then decrease, and KS-PLV reaches the best results when $\lambda$ is around 0.1. We also vary the values of $\gamma$ from 0 to 1. Figure 5 shows the results on the three datasets. With the growth of $\gamma$, the performances also increase first and then decrease, and our model achieves the best results when $\gamma$ is around 0.01. The results reveals that too large $\lambda$ or $\gamma$ would cause the learned intra-category embeddings too close to maintain their discrimination property.




(a) FB15K (b) FB15K-237 (c) FOOD

Figure 4: Fact checking performances w.r.t the trade-off parameter $\lambda$.




(a) FB15K (b) FB15K-237 (c) FOOD

Figure 5: Fact checking performances w.r.t the trade-off parameter $\gamma$.

#### *4.3.2.* ***Effect on Verification in KGs with Long Tail***

In real-world scenarios, a large portion of relations in KGs has long tail distributions [9]. To analyze the effect of our method in handling the long tail issue, we construct the long tail datasets for our study, following the convention provided by [10]. We add 6,607 new triples with 510 new relations from [10] to *FB15K-237* and construct a new dataset *FB15K-237+* with many sparse

| **Long Tail Dataset** (with relation frequency) | #**Test** | **Method** | | | | |
|---|---|---|---|---|---|---|
| | | TKRL | Dual TransE | A2N | Meta-KGR | KS-PLV |
| FB15K-237+1 ($f_r \leq 100$) | 2,544 | 0.633 | 0.601 | 0.619 | 0.621 | **0.675**$^*$ |
| FB15K-237+2 ($f_r \leq 1000$) | 7,702 | 0.724 | 0.693 | 0.725 | 0.733 | **0.784**$^*$ |
| FB15K-237+ (total) | 23,303 | 0.864 | 0.832 | 0.915 | 0.922 | **0.934** |

Table 5: Experimental results of accuracies on the long tail datasets (Here "$*$" denotes that KS-PLV significantly outperforms other methods).

relations. We then construct two datasets *FB15K-237+1* and *FB15K-237+2* by filtering out the triples in *FB15K-237+* with the constraints of relation frequency $f_r \leq 100$ and $f_r \leq 1000$ respectively.

We compare our method with four selected methods, TKRL[10], Dual TransE [7], Meta-KGR [9] and A2N [42]. Besides the original TKRL model in [10], we choose the state-of-the-art methods Dual TransE and A2N for fact checking and KG reasoning respectively. We also choose Meta-KGR, a RL based method particularly designed to solve the long tail problem. Table 5 shows the accuracies of different methods on the three constructed long tail datasets *FB15K-237+1*, *FB15K-237+2* and *FB15K-237+*. We can see from the table that KS-PLV outperforms the other methods on all the datasets, especially on the sparer datasets *FB15K-237+1* ($f_r \leq 100$) and *FB15K-237+2* ($f_r \leq 1000$). As the second best-performing method Meta-KGR is aimed at handling this problem, this demonstrates that our prototype learning based method KS-PLV can effectively alleviate the long tail issue in real-world KGs.

### *4.4. Illustration of Interpretable Evidence*

One merit of using HPL is to process the compactness and semantic similarity of entity embeddings of the same sub-category, so as to increase the ability of inducing the knowledge from homogeneous entities as evidence. In addition, *Attribute-GNN* is capable of further enhancing entity embeddings using neighboring information with attention mechanism. Table 6 illustrates several true examples of input claim triples from our realistic experimental data and the triples most similar to them as fact evidence, acquired by our method using

| Input claim triple | Most similar evidence |
|---|---|
| (passion fruit, prevent, cold)<br>(百香果，预防，感冒) | (passion fruit, strengthen, immunity)<br>(百香果，增强，免疫力) |
| (arkshell, lower, blood_lipid)<br>(赤贝，降，血脂) | (arkshell, reduce, cholesterol)<br>(赤贝，降低，胆固醇) |
| (alburnus, effect, diuresis)<br>(刁子鱼，作用，利尿) | (alburnus, nourish, kidney)<br>(刁子鱼，滋补，肾脏) |
| (pear, relieve, cough)<br>(梨，止，咳) | (pear, moisten, lung)<br>(梨，润，肺) |

Table 6: Illustrative *true* input triples (left column) and their most similar evidence computed by our KS-PLV (right column).

the highest attention scores. We can see from the table that the acquired fact evidence is quite similar to the input claim triples, which can effectively support fact checking.

## 5. Conclusions and Future Work

This paper proposes an end-to-end knowledge structure driven method KS-PLV for fact checking, which aims to utilize category hierarchy and attribute relationship to facilitate KG based learning and verification. We develop the first hierarchical prototype learning method to jointly learn a prototype for each sub-category and improve entity embeddings using high-level category information. We then propose a relation enhanced graph attention network, which can effectively induce neighboring attribute information to further enrich the semantic representations of entities. We also construct a new publicly available dataset on food domain, and conduct experimental studies based on our constructed domain dataset and two benchmark datasets. Our method outperforms all the comparison methods and achieves the state-of-the-art performances on the three datasets.

Our knowledge structure driven learning enhancement and verification approach can be broadly applied to domains such as knowledge-based rumor detection and textual-level fact verification, and can be extended to multi-claim fact checking [46] as well. In the future, we shall refine our work in several directions. We shall model the interrelation of entities and relations implied in a claim to develop better semantic matching functions, and consider to utilize more complex knowledge structures in real-world scenarios for fact checking.

## Acknowledgments

This work is supported in part by the Ministry of Science and Technology of China under Grant #2020AAA0108405, and National Natural Science Foundation of China under Grants #11832001 and #71621002.

## References

[1] H. Hopf, A. Krief, G. Mehta, S. A. Matlin, Fake science and the knowledge crisis: ignorance can be fatal, R. Soc. Open Sci. 6 (2019) 190161.

[2] G. L. Ciampaglia, P. Shiralkar, L. M. Rocha, J. Bollen, F. Menczer, A. Flammini, Computational fact checking from knowledge networks, PLoS One 10 (2015) 1–13.

[3] P. Shiralkar, A. Flammini, F. Menczer, G. L. Ciampaglia, Finding streams in knowledge graphs to support fact checking, in: Proceedings of ICDM, 2017, pp. 859–864.

[4] B. Shi, T. Weninger, Discriminative predicate path mining for fact checking in knowledge graphs, Knowl.-Based Syst. 104 (2016) 123–133.

[5] V. Fionda, G. Pirrò, Fact checking via evidence patterns, in: Proceedings of IJCAI, 2018, pp. 3755–3761.

[6] T. Dong, Z. Wang, J. Li, C. Bauckhage, A. B. Cremers, Triple classification using regions and fine-grained entity typing, in: Proceedings of AAAI, 2019, pp. 77–85.

[7] J. Z. Pan, S. Pavlova, C. Li, N. Li, Y. Li, J. Liu, Content based fake news detection using knowledge graphs, in: Proceedings of ISWC, 2018, pp. 669–683.

[8] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, O. Yakhnenko, Translating embeddings for modeling multi-relational data, in: Proceedings of NIPS, 2013, pp. 2787–2795.

[9] X. Lv, Y. Gu, X. Han, L. Hou, J. Li, Z. Liu, Adapting meta knowledge graph information for multi-hop reasoning over few-shot relations, in: Proceedings of EMNLP, 2019, pp. 3376–3381.

[10] R. Xie, Z. Liu, M. Sun, Representation learning of knowledge graphs with hierarchical types, in: Proceedings of IJCAI, 2016, pp. 2965–2971.

[11] T. Kohonen, The self-organizing map, Proceedings of the IEEE 78 (1990) 1464–1480.

[12] S. Geva, J. Sitte, Adaptive nearest neighbor pattern classification, IEEE Trans. Neural Netw. Learn. Syst 2 (1991) 318–322.

[13] J. Snell, K. Swersky, R. Zemel, Prototypical networks for few-shot learning, in: Proceedings of NIPS, 2017, pp. 4077–4087.

[14] H. Yang, X. Zhang, F. Yin, C. Liu, Robust classification with convolutional prototype learning, in: Proceedings of CVPR, 2018, pp. 3474–3482.

[15] K. Allen, E. Shelhamer, H. Shin, J. Tenenbaum, Infinite mixture prototypes for few-shot learning, in: Proceedings of ICML, 2019, pp. 232–241.

[16] P.-N. Tan, V. Kumar, J. Srivastava, Selecting the right interestingness measure for association patterns, in: Proceedings of KDD, 2002, pp. 32–41.

[17] L. A. Galárraga, C. Teflioudi, K. Hose, F. Suchanek, Amie: Association rule mining under incomplete evidence in ontological knowledge bases, in: Proceedings of the WWW, 2013, p. 413–422.

[18] L. Galárraga, C. Teflioudi, K. Hose, F. Suchanek, Fast Rule Mining in Ontological Knowledge Bases with AMIE+, VLDB J. 24 (2015) 707–730.

[19] F. Yang, Z. Yang, W. W. Cohen, Differentiable learning of logical rules for knowledge base reasoning, in: Proceedings of NIPS, 2017, p. 2316–2325.

[20] N. Lao, T. Mitchell, W. W. Cohen, Random walk inference and learning in a large scale knowledge base, in: Proceedings of EMNLP, 2011, pp. 529–539.

[21] N. Lao, W. W. Cohen, Relational retrieval using a combination of path-constrained random walks, Mach. Learn. 81 (2010) 53–67.

[22] R. Das, A. Neelakantan, D. Belanger, A. McCallum, Chains of reasoning over entities, relations, and text using recurrent neural networks, in: Proceedings of ACL, 2017, pp. 132–141.

[23] W. Xiong, T. Hoang, W. Y. Wang, DeepPath: A reinforcement learning method for knowledge graph reasoning, in: Proceedings of EMNLP, 2017, pp. 564–573.

[24] R. Das, S. Dhuliawala, M. Zaheer, L. Vilnis, I. Durugkar, A. Krishnamurthy, A. Smola, A. McCallum, Go for a walk and arrive at the answer: Reasoning over paths in knowledge bases using reinforcement learning, in: Proceedings of ICLR, 2018.

[25] X. V. Lin, R. Socher, C. Xiong, Multi-hop knowledge graph reasoning with reward shaping, in: Proceedings of EMNLP, 2018, pp. 3243–3253.

[26] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, O. Yakhnenko, Translating embeddings for modeling multi-relational data, in: Proceedings of NIPS, 2013, pp. 2787–2795.

[27] Z. Wang, J. Zhang, J. Feng, Z. Chen, Knowledge graph embedding by translating on hyperplanes, in: Proceedings of AAAI, 2014, pp. 1112–1119.

[28] Y. Lin, Z. Liu, M. Sun, Y. Liu, X. Zhu, Learning entity and relation embeddings for knowledge graph completion, in: Proceedings of AAAI, 2015.

[29] G. Ji, S. He, L. Xu, K. Liu, J. Zhao, Knowledge graph embedding via dynamic mapping matrix, in: Proceedings of ACL, 2015, pp. 687–696.

[30] M. Fan, Q. Zhou, E. Chang, F. Zheng, Transition-based knowledge graph

embedding with relational mapping properties, in: Proceedings of AAAI, 2014, pp. 328–337.

[31] J. Feng, M. Huang, M. Wang, M. Zhou, Y. Hao, X. Zhu, Knowledge graph embedding by flexible translation, in: Proceedings of KR, 2016.

[32] S. He, K. Liu, G. Ji, J. Zhao, Learning to represent knowledge graphs with gaussian embedding, in: Proceedings of CIKM, 2015, pp. 623–632.

[33] H. Xiao, M. Huang, X. Zhu, TransG : A generative model for knowledge graph embedding, in: Proceedings of ACL, 2016, pp. 2316–2325.

[34] M. Nickel, V. Tresp, H. Kriegel, A three-way model for collective learning on multi-relational data, in: Proceedings of ICML, 2011, p. 809–816.

[35] R. Socher, D. Chen, C. D. Manning, A. Ng, Reasoning with neural tensor networks for knowledge base completion, in: Proceedings of NIPS, 2013, pp. 926–934.

[36] B. Yang, W. Yih, X. He, J. Gao, L. Deng, Embedding entities and relations for learning and inference in knowledge bases, in: Proceedings of ICLR, 2015.

[37] M. Nickel, L. Rosasco, T. A. Poggio, et al., Holographic embeddings of knowledge graphs, in: Proceedings of AAAI, 2016, pp. 3–2.

[38] T. Trouillon, J. Welbl, S. Riedel, E. Gaussier, G. Bouchard, Complex embeddings for simple link prediction, in: Proceedings of ICML, 2016, p. 2071–2080.

[39] H. Liu, Y. Wu, Y. Yang, Analogical inference for multi-relational embeddings, in: Proceedings of ICML, 2017, p. 2168–2178.

[40] T. Dettmers, P. Minervini, P. Stenetorp, S. Riedel, Convolutional 2d knowledge graph embeddings, in: Proceedings of AAAI, 2018, pp. 1811–1818.

[41] M. Schlichtkrull, T. N. Kipf, P. Bloem, R. Van Den Berg, I. Titov, M. Welling, Modeling relational data with graph convolutional networks, in: Proceedings of ESWC, 2018, pp. 593–607.

[42] T. Bansal, D. Juan, S. Ravi, A. McCallum, A2n: Attending to neighbors for knowledge graph inference, in: Proceedings of ACL, 2019, pp. 4387–4392.

[43] J. Thorne, A. Vlachos, C. Christodoulopoulos, A. Mittal, FEVER: a large-scale dataset for fact extraction and VERification, in: Proceedings of ACL, 2018, pp. 809–819.

[44] A. Hanselowski, H. Zhang, Z. Li, D. Sorokin, B. Schiller, C. Schulz, I. Gurevych, UKP-athene: Multi-sentence textual entailment for claim verification, in: Proceedings of FEVER, 2018, pp. 103–108.

[45] J. Zhou, X. Han, C. Yang, Z. Liu, L. Wang, C. Li, M. Sun, GEAR: Graph-based evidence aggregating and reasoning for fact verification, in: Proceedings of ACL, 2019, pp. 892–901.

[46] S. Wang, W. Mao, Modeling inter-claim interactions for verifying multiple claims, in: Proceedings of CIKM, 2021.