Semantics-driven Refinement for Relational Data

Nitisha Jain
Hasso Plattner Institute
University of Potsdam
Potsdam, Germany
Nitisha.Jain@hpi.de

Ralf Krestel
ZBW - Leibniz Centre for Economics
Kiel University
Kiel, Germany
R.Krestel@zbw.eu

Abstract

The semantics of relations are crucial for comprehending and analyzing multi-relational data. Polysemous relations between different types of entities (i.e. those that represent multiple semantics) are common in real-world relational datasets represented by knowledge graphs. For numerous use cases, such as entity type classification, question answering and knowledge graph completion, the correct semantic interpretation of these relations is necessary. In this work, we provide a method for discovering the different semantics associated with abstract relations and deriving many sub-relations with fine-grained meaning. To do this, we leverage the types of the entities associated with the relations and cluster the vector representations of entities and relations. The suggested method automatically discovers the best number of sub-relations for a polysemous relation and determine their semantic interpretation, according to our empirical evaluation.

1 Introduction

Relations between different words or phrases are important for the semantic understanding of text. Popular knowledge graphs (KGs) such as Yago (Mahdisoltani et al., 2014), NELL (Mitchell et al., 2018) and DBpedia (Lehmann et al., 2015) are formulated in terms of relational databases where the entities are linked to each other with different relations or predicates. In real-world textual data, the relations are often polysemous by nature, i.e., they exhibit distinct meanings in different contexts. For example, the relation '*part of*' has different semantics in '*..Sahara is part of Africa*' and '*humans part of mammals*'. Similar to the task of word sense disambiguation, which is required to understand different contextual meanings of words, relation disambiguation is needed to interpret the specific, contextual semantics of relations. Relation polysemy occurs frequently in open texts and has been studied and discussed by previous works on automatic relation extraction from texts (Min et al., 2012; Galárraga et al., 2014; Han and Sun, 2016). Relation semantics are particularly important in the context of knowledge graphs which are widely used for systematic representation of data in the form of $\langle subject, predicate, object \rangle$ triples. Here, *subject* and *object* are chosen from a set of entities, while the *predicate* that links the entities to each other belongs to a set of relations. As the triples in KGs are derived from real-world facts, ambiguity from texts often makes it way into the KG triples as well. Specifically, the relations may represent multiple meanings depending on the context, which in the case of KG triples, is defined by the types of the entities being connected by the relations. The underlying idea of defining the role of words by their context is quite old in Linguistics, advocated by Firth : 'a word is characterized by the company it keeps' (Firth, 1957). In the context of KGs, one could say *'a relation is characterized by the entity types it connects'*.

In order to gauge the issue of multiple relation semantics in popular KGs, we analysed the relations in the Yago3 (Mahdisoltani et al., 2014) dataset in terms of the number of unique entity types pairs that were found in the associated triples. The results are plotted in Figure 1. It can be seen that the majority of the relations have multiple entity types associated with them. Among these, many relations such as $owns$ and $created$ are generic in their semantics and exhibit very high plurality of entity types. Similar insights were also derived from the NELL knowledge graph. Some examples of the actual entity types associated with polysemous relations from these KGs are shown in Table 1.

In this work we advocate that for such relations that are associated with a number of different entity type pairs, it would be prudent to instead split them and create new sub-relations that have a more distinct meaning according to the context. The exact meanings of the sub-relations could be clearly

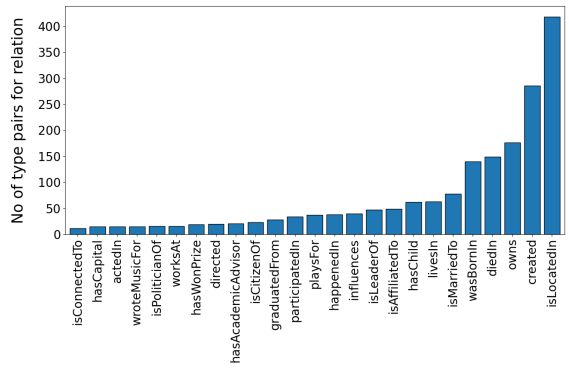

Figure 1: The type pairs associated with different relations in Yago.

defined based on the distinct types of the associated entities. However, this approach can be tricky since entity types can vary widely. While some types such as *television* and *movie* for the *created* relation in Yago are semantically similar to one another, other types are quite different, for instance *company* and *writer*. If the relation is split based on the different entity types in a straightforward manner, without taking the similarity of these types into account, the resultant sub-relations would end up being semantically similar to each other. Due to the complex hierarchy of classes (entity types)[1] in the underlying schema (or ontology), entity types belong to different granularity levels (Jain et al., 2021) leading to a wide range of semantic similarity between them. It is, therefore, a non-trivial task to decide how a relation should be split into sub-relations based on the semantics of the entity types associated with it, both in terms of the number of sub-relations as well the subset of entity types that the sub-relations should encompass.

In this work, we define the problem statement of *fine-grained relation refinement* which refers to the disambiguation of polysemous relations in relational datasets and propose a data-driven and scalable method named ***FineGReS (Fine-Grained Relation Semantics)*** as a first solution towards the same. The fine-grained semantics for the relations are derived by relying on the multiple semantics of the relations evident from the types of the associated entities. The proposed approach leverages knowledge graph embeddings, that provide representations of the KG entities and relations in a continuous vector space. We find optimal clusters in the latent space to identify the underlying semantic features such that polysemous relations can be represented in terms of multiple sub-relations with well-defined semantics. This approach automatically determines not only the optimal number of sub-relations (corresponding to the number of clusters), but also the entity types that should be associated with each of them so as to have clearly defined semantic representation. Experimental evaluation performed on popular relational datasets reveals the benefits of defining fine-grained semantics and brings forth the efficacy of the proposed approach in the face of the challenging nature of this task.

[1] the terms *class* and *entity type* will be used interchangeably in the paper from this point on

Table 1: Examples of Multiple Semantics of Relations.

Yago *created*	**NELL *agentBelongsTo-Organization***
(writer, movie)	(politician, politicalparty)
(player, movie)	(country, sportsleague)
(artist, movie)	(sportsteam, sportsleague)
(officeholder,movie)	(coach—sportsleague)
(writer,fictional_character)	(person, charactertrait)
(artist,computer_game)	(televisionstation, company)
(artist,medium)	
(writer,television)	
(company,computer_game)	

Contributions. 1. We formally define the task of *fine-grained relation refinement* in relational datasets and motivate its importance and benefits. 2. We propose the data-driven and scalable method *FineGReS* to identify multiple sub-relations that capture the different semantics of the relations via clustering in the latent space. 3. We perform empirical evaluation and illustrate the benefits of the method on downstream applications, such as entity classification.

2 Fine-Grained Relation Semantics

Relation polysemy is quite common in knowledge graphs for two primary reasons. Firstly, the schema for most large scale KGs that are in use today have been constructed through manual or semi-automated efforts, where the relations between the entities are curated from text. Relations are often abstracted in such KGs for simplification and avoidance of redundancies. This may result in cases where a single relation serves as a general notion between various different types of KG entities and has more than one semantic meaning associated with it. In addition to this, the fact that these KGs represent real-world facts that are expressed in natural language having inherent ambiguities, contributes further to the relation polysemy in KGs. For instance,

the relation phrase '*part of*' represents varied semantics based on its context of biology (*finger part of hand*), organizations (*Google part of Alphabet*), geography (*Amazon part of South America*) and many others. Even KGs that have a large number of different relations can suffer from ambiguous relations. For instance, DBpedia has around 300 relations that are relatively well-defined in terms of their entity types, and yet there exist relations such as *award* and *partOf* that still convey ambiguity. The determination of fine-grained relation semantics in relational data is an important task which can bring substantial benefits to a wide range of NLP and semantic use cases as discussed further in this section.

The task of **relation extraction** is essential for information extraction from texts and it continues to be challenging due to the varied semantics of the evolving language. For identifying patterns and extracting relation mentions from text, unsupervised techniques typically rely on the predefined types of relation arguments (Hasegawa et al., 2004; Shinyama and Sekine, 2006; Chen et al., 2005). Given an existing KG and schema, with the goal to extract facts for a particular relation from a new corpus of text, a distant supervision approach will leverage relation patterns based on the types of entities over the text. As an example, if the relation *created* has been established between a *painter* and *artwork*, then the identification of this relation can be aided by specific patterns in text. However, if the relation *created* is generically defined between any *person* entity and any *work* entity, then the resulting text patterns for this relation will be noisy and varied, therefore may fail to identify the correct fact triples from text. Identifying the different meanings of a relation in different contexts can help with defining concrete patterns for extraction of relation phrases.

This is also useful for identification and **classification of entities** by their types in a knowledge graph. E.g. the target entity of the relation *directed* is likely to be of type *movie* or *play*. If the relations have a wider semantic range, the type of entities cannot be identified at a fine-grained level. For instance, it might be only possible to identify the entity type as *work* and not specifically *movie*, which could adversely affect the performance of further applications such as **entity linking** and **question answering**. Numerous question answering systems that use knowledge graphs as back-end data repositories (KBQA) (Cui et al., 2019) rely on the type information of the entities to narrow down the search space for the correct answers. Thus, distinct relation semantics in terms of the types of connected entities are essential for supporting QA applications over KGs.

We would like to emphasize that task of defining fine-grained relation semantics is important in the context of **KG refinement**, not being merely limited to already existing datasets but in general. KGs usually evolve over time and often in a fragmented fashion, where new facts might be added to a KG that do not strictly conform to or cannot be correctly encapsulated by the existing ontology. Addition of such new facts might easily lead to noisy and abstracted semantics in previously well-defined KG relations. Relation disambiguation would therefore play a important role in identifying new fine-grained sub-relations with precise semantics. The proposed *FineGReS* method is generally applicable and could prove to be incredibly useful in all the above scenarios.

In the next section, we provide the necessary background and present the problem statement.

3 Preliminaries

Knowledge Graph. For a knowledge graph $\mathcal{G}$, the set of unique relations is denoted as $\mathcal{R}$. A KG fact (or triple) $F = \langle e_h, r, e_t \rangle$ consists of the head entity e_h, the tail entity e_t and the relation r that connects them, where e_h and e_t belong to the set of entities $\mathcal{E}$. Any given relation $r \in \mathcal{R}$ appears in several triples, forming a subset $\mathcal{G}_r$ of $\mathcal{G}$.

Entity Types. The semantic types or classes of the entities are defined in an ontology associated with a KG that defines its schema. The entities $e \in \mathcal{E}$ are connected with their types by ontological triples such as $\langle e, typeOf, t \rangle$, where $t \in T$, the set of entity types in the ontology.

We define a type pair as the tuple $\langle t_h, t_t \rangle$ where $\langle e_h, typeOf, t_h \rangle$ and $\langle e_t, typeOf, t_t \rangle$. A set of unique type pairs (for a given relation r and corresponding $\mathcal{G}_r$ is denoted as P_r. Thus we have, $P_r = \{\langle t_h, t_t \rangle | \langle e_h, typeOf, t_h \rangle, \langle e_t, typeOf, t_t \rangle, \langle e_h, r, e_t \rangle \in \mathcal{G}_r\}$. The total number of such unique type pairs for relation r, i.e. $|P_r|$ is denoted by $\mathcal{L}_r$.

KG Embeddings. Knowledge graph embeddings have gained immense popularity and success for representation learning of relational data. They provide an efficient way to capture latent semantics of the entities and relations in KGs. The main advantage of these techniques is that they enable

easy manipulation of KG components when represented as vectors in low dimensional space. For example in TransE (Bordes et al., 2013), for a triple $\langle h, r, t \rangle$ the vectors **h**, **r** and **t** satisfy the relation **h** + **r** = **t** or **r** = **t** - **h**. In this work, we leverage the representational abilities of the embeddings to obtain the semantic vectors for relations expressed in terms of the entities associated with them. For vectors **h**, **r** and **t** as obtained from an embedding corresponding to a KG triple $\langle e_h, r, e_t \rangle$, we define a vector $\boldsymbol{\Delta_{r_i}}$ which is a function of $\boldsymbol{t_i}$ and $\boldsymbol{h_i}$. Every $\boldsymbol{\Delta_{r_i}}$ vector is mapped to a type pair P_{r_i} as per the entities e_h, e_t that they are both derived from.

Problem Definition. Given a relation $r \in \mathcal{R}$ in $\mathcal{G}$, the set of $\boldsymbol{\Delta_r}$ vectors and the corresponding set of type pairs P_r, the goal is to find for this relation an optimal configuration of clusters $\mathcal{C}_{opt}$ = $\{\mathcal{C}_1, \mathcal{C}_2 ... \mathcal{C}_N\}$, where the $\boldsymbol{\Delta_{r_i}}$ vectors are uniquely distributed among the clusters i.e. each $\boldsymbol{\Delta_{r_i}} \in \mathcal{C}_j$, $i = 1...|\mathcal{G}_r|$, $j = 1...N$, s.t. an objective function $\mathcal{F}(\mathcal{C}_{opt})$ is maximized.

Further, each cluster $\mathcal{C}_j$ represents the semantic union of the subset of type pairs P_r where $\exists\ \boldsymbol{\Delta_{r_i}} \in \mathcal{C}_j$ s.t. $\boldsymbol{\Delta_{r_i}}$ is mapped to one of the type pairs in P_r. Thus, the optimal configuration of clusters corresponds to the optimal number of sub-relations and their fine-grained semantics as defined by the type pairs that they represent. The proposed *FineGReS* method can derive this optimal configuration for the relations of a KG.

4 Method

In this section, we describe in detail the design and implementation details of the proposed *FineGReS* method for a relation that can easily scaled to any number of relations in the datset.

4.1 Semantic Mapping for Facts

For every unique relation r in $\mathcal{G}$, we firstly find the subset of facts $\mathcal{G}_r$ where r appears. To understand the semantics of the entities associated with r, the entities are mapped to their corresponding classes as defined in the underlying ontology. By doing so, we obtain a list of entity type pairs $\langle t_h, t_t \rangle$ for the relation. Note that several entities in $\mathcal{G}_r$ might map to the same type and therefore, a single type pair tuple would be obtained several times. Therefore in the next step, we identify the unique type pairs for a relation r as the set P. At this stage, every fact in $\mathcal{G}_r$ is associated with a type pair $\langle t_h, t_t \rangle \in P$ that represents the semantics of this fact. For example, for the $created$ relation, a triple $\langle DaVinci, created, MonaLisa \rangle$ would be mapped to $\langle artist, painting \rangle$ as per the types of the head and tail entities.

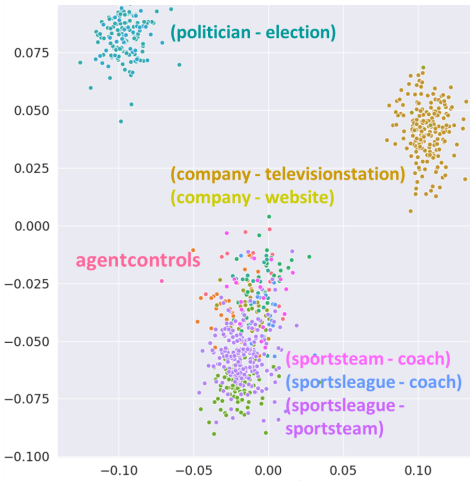

Figure 2: Visualization of vectors for *agentcontrols* relation in NELL with associated type pairs.

4.2 Vector Representations for Relations

For representing the semantics of r in terms of the associated entities, we leverage pre-trained KG embeddings. As proposed in previous work (Jiang et al., 2020), we obtain a representation for r from **h** and **t** corresponding to every fact in $\mathcal{G}_r$ and denote this vector as $\boldsymbol{\Delta_r}$. In this way, for every relation r, a set of $\boldsymbol{\Delta_r}$ vectors is obtained from the KG embeddings, in addition to the actual **r** vector that the embedding already provides. These $\boldsymbol{\Delta_r}$ vectors are already mapped to the unique type pairs $P_i \in P$ for the relation r (according to the fact triples they were calculated from)[2], such that each unique type pair is represented by a set of $\boldsymbol{\Delta_r}$ vectors. These $\boldsymbol{\Delta_r}$ vectors encode the information conveyed by both the head and tail entity types together and represent the relationship between the entities, they therefore represent the latent semantics of the relations in different facts. The $\boldsymbol{\Delta_r}$ vectors become our data points (with the associated type pairs P_i serving as their labels).

Relation Semantics. While it is believed that KG embeddings are able to capture relation similarity in the embedding space, i.e., relations having similar semantics occur close together in the vector space (Do et al., 2018; Kalo et al., 2019), we found that relations having multiple semantics (based on the context of their entities) are, in fact, not represented well in the vector space. In fact,

[2] we denote P_r as P when the relation r is clear from the context

for polysemous relations, the vectors obtained for a single relation (from the different facts that it appears in) form separate clusters in the vector space that do not overlap with the actual relation vector **r** obtained from the embeddings. This happens due to the fact that multiple entity pairs connected by the same relation are semantically different from one another. Figures 2 illustrates an example from the NELL dataset where this behaviour of the embedding vectors for relations is clearly visible. We leverage this semantically-aware behaviour of the embedding vectors to determine meaningful clusters of $\mathbf{\Delta_r}$ vectors that represent the distinct latent semantics exhibited by different entity type pairs connected by the same relation, as described next.

4.3 Clustering for Fine-grained Semantics

For each relation r, the total number of unique type pairs $\mathcal{L} = |P_r|$ is theoretically the maximum number of possible semantic sub-relations or clusters that could be obtained for r. This is the *maximal* splitting that will assign a different sub-relation for every different type pair. However, in practice, it is rare that all the type pairs would have completely different semantics. For example, the $created$ relation in Yago has type pairs $\langle artist, painting \rangle$ and $\langle artist, music \rangle$ that have the same head entity type, while the type pair $\langle organization, software \rangle$ conveys quite a different meaning. While a single relation is not sufficient to be representative of the semantics of all facts which is it appears in, at the same time, a naive *maximal* splitting of the relation as per the unique type pairs would also be inefficient and lead to a large number of unnecessary sub-relations.

The *FineGReS* method aims to find an optimal number and composition of clusters $\mathcal{C}_{opt}$ for the type pairs that can convey distinct semantics of the relations based on the data, by combining similar type pairs while separating the dissimilar ones. Each of the clusters having one or more than one semantically similar type pairs represents a potential sub-relation. In order to obtain this configuration, various compositions of the clusters need to be analysed for optimality. Clustering is performed in an iterative manner with a predefined number of clusters and combinations of type pairs within each cluster. Since it is not feasible or practical to consider all possible clusters of the type pairs, *FineGReS* leverages the semantic similarity of type pairs to narrow down the search space for obtaining the optimal clusters. For this, the similarity scores between all combinations of the unique type pairs (t_{h_i}, t_{t_i}), (t_{h_j}, t_{t_j}) are calculated. First, the vector representations for the types are obtained. Subsequently, the similarity scores are obtained by calculating the similarity scores between the vectors corresponding to t_{h_i} and t_{h_j} as well as t_{t_i} and t_{t_j} and then taking their mean value.

Iterative Clustering. The clustering begins with $\mathcal{L}$ clusters, with each cluster corresponding to one type pair for the relation. The cluster labels are regarded as type pairs themselves. Following this, the similarity scores of the type pairs are calculated, and the ones with the highest similarity are considered as candidate pairs to be merged together and placed in a single cluster. To generate the 'ground truth', the data points ($\mathbf{\Delta_r}$ vectors) corresponding to both type pairs are assigned the same distinct label and this dataset is separately used for evaluating the cluster at every iteration. The number of clusters is given as $\mathcal{L}$ - 1 during the next iteration of clustering, and the cluster labels consist of $\mathcal{L}$ - 2 original type pairs and one merged type pair. If two combinations of type pairs have the same similarity score in any iteration, ties are broken arbitrarily. This process of selecting the most similar pair of class combinations for lowering the number of clusters and obtaining cluster labels as ground truth is repeated until all type pairs have been gradually merged back together in a single cluster. For each iteration, the quality of the clusters is evaluated and compared to the engineered ground truth, and the best performing cluster configuration is chosen as the optimal set of clusters $\mathcal{C}_{opt}$ that best represent the sub-relations with fine-grained semantics[3].

5 Experimental Analysis

To evaluate the performance of the *FineGReS*, we performed intrinsic empirical analysis in terms of the quality of the optimal clusters as obtained from the method. Further, to show the performance gains from *FineGReS* for a relevant use case, we performed extrinsic evaluation for the task of entity classification.

Datasets. We prepared datasets derived from Yago3 and NELL-995 knowledge graphs for relation analysis and disambiguation in this work. The

[3]It is to be noted that the labeling of newly proposed sub-relations is a separate task on its own. In this work, we merely utilize names of entity types to derive basic labels, however a proper naming scheme for these relations is a complex task in the context of ontology design and out of the scope of the current work.

Table 2: Quality of *FineGReS* clusters (C_{FGReS}) in comparison with baselines.

		Yago				NELL			
	Clustering Technique	C_{max}	C_{head}	C_{tail}	C_{FGReS}	C_{max}	C_{head}	C_{tail}	C_{FGReS}
TransE	KMC	.245	.199	.190	**.269**	.384	.304	.318	**.463**
	OPC	.456	.506	422	**.524**	.192	.197	.203	**.258**
	SPC	**.040**	.027	.012	.031	.332	.185	.190	**.337**
	HAC	.217	.195	.182	**.254**	.335	.245	.293	**.374**
DistMult	KMC	**.186**	.101	.166	.183	.348	.212	.298	**.369**
	OPC	.430	.424	.423	**.451**	.342	340	.349	**.370**
	SPC	.316	**.469**	.031	.332	.287	.163	.196	**.307**
	HAC	.212	.208	.176	**.237**	.283	.173	.227	**.292**

Yago dataset consists of 1,492,078 triples, with 31 relations and 917,325 unique entities. The NELL dataset includes 154,213 triples, with 200 relations and 75,492 unique entities. For both, the entities were augmented with their types as derived from the respective ontologies of the KGs.

Vectors for Entity Types. In order to obtain word vector representations of the class types, we use the pre-trained ConVec embeddings (Ehsan Sherkat, 2017). We also leveraged the pre-trained *Sentence-BERT* (Reimers et al., 2019) models from the HuggingFace library (Wolf et al., 2019). Cosine similarity measure was used for calculating the vector similarities (Euclidean similarity measure provided very similar results).

Knowledge Graph Embeddings. We perform our experiments on the following widely used KG embedding models : TransE (Bordes et al., 2013) and DistMult (Yang et al., 2014). These models are chosen to serve as prominent examples of embeddings using translation distance and semantic matching techniques respectively. We use the model implementations from the LibKGE library (Broscheit et al., 2020) for the Yago3-10 dataset and from OpenKE library (Han et al., 2018) for the NELL-995 dataset. It is important to note that in this work, we have purposefully chosen those embedding techniques that showed the promise of being able to differentiate the multiple semantics of a relation. Indeed there have been some embedding models (Xiao et al., 2016) that instead aim to encapsulate the different semantics of the relation in a single representative vector, however such techniques are not fruitful towards our goal of fine-grained relation refinement.

Baselines. We establish several baselines to evaluate and compare the performance of the optimal clusters (hence the corresponding sub-relations) derived from *FineGReS* to naive approaches in each experimental setting.

max - Sub-relations are obtained on the basis of every different type pair that is found associated with a relation, this naive setting corresponds to the maximum number of clusters.

head - Sub-relations represent all type pairs associated with a common head entity and different tail entities. It is equivalent to grouping the type pairs in the *max* setting by head entity types.

tail - Similar to *head*, just replacing head entity with tail entity instead so that sub-relations represent all type pairs associated with a common tail entity and different head entities.

Clustering Techniques. To explore the affect of different techniques during the clustering step of *FineGReS* method, we employed several algorithms : KMeans clustering(KMC), Spectral (SPC), Optics (OPC) and Hierarchical Agglomerative clustering(HAC).

5.1 Cluster Quality Evaluation

The quality of the clusters, and thereby, the resultant sub-relations is measured in terms of homogeneity score (Jain et al., 2018), this metric favors a clustering scheme where every cluster represents a unique dominant label that corresponds to one or more unique type pairs in our case. Thus, this metric best represents the distinct semantics of the clusters. We measure and report the weighted (as per the number of data points) homogeneity scores for the clusters of all relations in the different settings of KG embeddings and clustering techniques in Table 2. It can be seen that in most cases the $\mathcal{C}_{FGReS}$ clusters obtained by the proposed *FineGReS* method obtain higher scores than all the other baselines that were considered for defining

Table 3: Examples of Fine-grained Sub-relations.

Dataset - Relation (setting)	*FineGReS* Sub-relations
Yago - *owns* (TransE-HAC)	$\{\langle company, airport\rangle\ \langle organization, airport\rangle\}$, $\{\langle sovereign, building\rangle\}$, $\{\langle company, club\rangle\ \langle company, company\rangle, \langle country, club\rangle\ \}$
Yago - *created* (TransE-OPC)	$\{\langle artist, medium\rangle\ \langle of ficeholder, movie\rangle\}$, $\{\langle writer, fictional\_character\rangle\}$, $\{\langle writer, movie\rangle\ \langle writer, television\rangle\ \langle writer, fictional\_character\rangle\ \langle artist, movie\rangle\ \langle artist, computer\_game\rangle\ \langle player, movie\rangle\}$, $\{\langle company, computer\_game\rangle\}$
NELL-*agentCompetesWith* (TransE-Kmeans)	$\{\langle company, person\rangle\ \langle website, person\rangle\ \langle person, person\rangle, \langle sportsteam, sportsteam\rangle\}$, $\{\langle person, company\rangle, \langle person, website\rangle\}$ $\{\langle animal, animal\rangle, \langle bird, animal\rangle\}$, $\{\langle bank, bank\rangle\}$, $\{\langle mammal, politicsissue\rangle\}$
NELL-*subpartOfOrganization* (DistMult-Kmeans)	$\{\langle sportsteam, sportsteam\rangle\ \langle stateorprovince, sportsteam\rangle\ \langle university, sportsteam\rangle\ \langle city, sportsteam\rangle\ \}$, $\{\langle organization, organization\rangle\}$, $\{\langle televisionstation, city\rangle\}$, $\{\langle company, company\rangle\ \langle televisionstation, company\rangle\}$, $\{\langle sportsteam, sportsleague\rangle\}$, $\{\langle bank, bank\rangle\}$ $\{\langle televisionstation, televisionnetwork\rangle\}$, $\{\langle televisionstation, website\rangle\}$

Table 4: Performance Comparison for Entity Classification Task for Yago (*r* refers to original relations).

					FineGReS	
	r	*max*	*head*	*tail*	TransE	DistMult
Precision	.893	.916	.906	.322	**.923**	**.928**
Recall	.908	.925	.921	.425	**.941**	**.942**
F1 score	.894	.914	.909	.34	**.931**	**.931**

the cluster configurations for sub-relations. This indicates the efficacy of the method for finding optimal fine-grained sub-relations.

Discussion. Table 3 shows a few representative examples of the sub-relations obtained by *FineGReS* in different settings for Yago and NELL. It can be seen that semantically different entity type pairs have been clearly separated out as distinct sub-relations, e.g. the $\langle sovereign, building\rangle$ pair for *owns* relation where *sovereign* is semantically distant from other types or *agentCompetesWith* where $\langle bank, bank\rangle$ is a separate sub-relation. Other sub-relations have multiple type pairs associated with them based on their semantic proximity. Note that in a few cases, the optimal configuration could indeed be the *max* or *head/tail* setting for the relation depending on the associated type pairs. The *FineGReS* method is able to automatically determine this optimal configurations of the sub-relations for each relation relying solely on the facts in the dataset and the associated type information.

5.2 Entity Classification Use Case

In order to empirically evaluate the *FineGReS* method in terms of the usefulness of the derived sub-relations, we consider the popular use case of entity classification which is an important task for KG completion (Neelakantan and Chang, 2015). It is modeled as a supervised multi-label classification task, where the entities are assigned to their respective types. Previous works have performed type prediction for entities in KGs based on statistical features (Paulheim and Bizer, 2013), textual information (Kliegr and Zamazal, 2016) as well as embeddings (Biswas et al., 2020). Taking cue from the same, we built a CNN classifier (Zhang and Wallace, 2017) for the multi-label classification task which can jointly classify both the entities in a given triple to their respective types.

The dataset for the classification task was obtained by replacing the original polysemous relations in the KG dataset with their corresponding fine-grained sub-relations in the affected triples, obtained from the best performing setting of the *FineGReS* method as well as from the baseline techniques described in Section 5. The performance of entity classification was measured in terms of weighted precision, recall and F1 scores (averaged over 10 runs). The results for the Yago dataset are shown in Table 4 (similar results were obtained for NELL). It can be seen that with well-defined relation semantics, the performance of the entity classification task improved considerably. In particular, the gains seen over the *max* setting are indicative of the superiority of the *FineGReS* method in terms of not merely finding *any* set of sub-relations but finding the *optimal* configuration of the sub-relations that best represent fine-grained semantics for all relations.

6 RELATED WORK

There is a large body of work in linguistics that deals with multiple semantics of words (Erk and Padó, 2008; Reisinger and Mooney, 2010; Neelakantan et al., 2014). Early work on the semantic connections between the relations and their associ-

ated entities in texts was introduced in 1963 (Katz and Fodor, 1963) with the concept of *selectional preference* for performing predicate or verb sense disambiguation (Resnik, 1997). The idea advocates that verbs can semantically restrict the types of the arguments that occur around them, thus having a preference for certain classes of entities. In contrast, our work in this paper focuses on the predicates that are generic and have insufficient constraints for their entity types. In such cases, the diverse entity types were, in fact, leveraged to identify and define the fine-grained semantics of these predicates by dividing them into multiple predicates.

In other work related to relation semantics, (Jiang et al., 2020) explores the entailment between relations, e.g. the relation *creator* entails *author* or *developer* in the sense that *creator* subsumes the other relations. Similar to our work, the authors leverage the entity type information to solve the multi-classification problem of assigning the child relations to the parent ones. Our problem statement of fine-grained relation refinement is significantly more challenging and impactful in the sense that it involves the identification of novel sub-relations in an unsupervised manner.

While the idea of learning better embeddings for *words* by considering their multiple contextual semantics is not new (Vu and Parker, 2016), the semantics of *relations* have also been recently studied in regards to learning knowledge graph embeddings. In (Lin et al., 2015) the authors advocated the need for learning multiple relation vectors to capture the fine-grained semantics, however this study was limited in scope and lacked any consideration for complex entity type hierarchies in KGs. In (Zhang et al., 2018), the authors create a 3-level relation hierarchy which combines similar relations as well splits relations into sub-relations, in order to improve the embeddings for relations. The proposed approach is quite rigid and opaque in terms of the actual semantics of the relations obtained from it. In fact, the number of clusters was predefined for all relations across a dataset, in contrast to the *FineGReS* method that can determine an optimal number of clusters separately for each relation based on the associated entity types.

The diverse semantics of relations was also considered by (Ji et al., 2015) where the authors proposed two different vectors for the relations as well as entities, to capture their meanings and connections with each other. Similarly, in (Xiao et al., 2016) the authors discussed the generation of multiple translation components of relations based on their semantics with the help of a bayesian nonparametric infinite mixture model. However, they do not perform a systematic analysis of the relations semantics and a qualitative evaluation of their approach is missing.

In general, the above discussed works have leveraged relation polysemy for designing better embedding models and improving their link prediction performance on knowledge graphs. In this work, we instead approach the problem of relation polysemy and discovery of the latent relation semantics with the goal of knowledge graph refinement and improvement of the quality of the relations in underlying ontology. More importantly, none of the previous works have explored the challenges of deriving fine-grained relations from an existing polysemous relation in the presence of complex semantic relationships between the associated entity types, which is quite common for real-world datasets. Our proposed *FineGReS* method performs this task in a systematic and data-driven fashion and shows promising benefits for downstream applications.

7 Conclusion

In this paper, we have studied the need for relation disambiguation for knowledge graphs due to the inherent relation polysemy in these datasets. We have proposed a scalable, data-driven method *FineGReS* that automatically determines an optimal configuration for deriving sub-relations with concrete semantics. Empirical evaluation has demonstrated the efficacy of the method for learning fine-grained relation semantics for real-world data. The performance improvement achieved for downstream application of entity classification strongly indicates the promise of this approach. Since the method relies on the type information of the entities, *FineGReS* can currently be applied only to the KGs accompanied by their ontologies. It would be interesting to extend the proposed approach to derive entity semantics from other sources, such as text. As future work, we also plan to perform a systematic analysis of the utility and impact of this method on further NLP tasks, such as relation extraction and question answering over KGs.

Acknowledgments

We would like to thank David Tresner-Kirsch for his valuable feedback and comments.

References

Russa Biswas, Radina Sofronova, Mehwish Alam, and Harald Sack. 2020. Entity type prediction in knowledge graphs using embeddings. *arXiv preprint arXiv:2004.13702*.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in neural information processing systems*, pages 2787–2795.

Samuel Broscheit, Daniel Ruffinelli, Adrian Kochsiek, Patrick Betz, and Rainer Gemulla. 2020. LibKGE - A knowledge graph embedding library for reproducible research. In *Proc. of the 2020 Conf. on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 165–174.

Jinxiu Chen, Donghong Ji, Chew Lim Tan, and Zheng-Yu Niu. 2005. Unsupervised feature selection for relation extraction. In *Companion Volume to the Proceedings of Conference including Posters/Demos and tutorial abstracts*.

Wanyun Cui, Yanghua Xiao, Haixun Wang, Yangqiu Song, Seung-won Hwang, and Wei Wang. 2019. Kbqa: learning question answering over qa corpora and knowledge bases. *arXiv preprint arXiv:1903.02419*.

Kien Do, Truyen Tran, and Svetha Venkatesh. 2018. Knowledge graph embedding with multiple relation projections. In *2018 24th International Conference on Pattern Recognition (ICPR)*, pages 332–337. IEEE.

Evangelos Milios Ehsan Sherkat. 2017. Vector embedding of wikipedia concepts and entities.

Katrin Erk and Sebastian Padó. 2008. A structured vector space model for word meaning in context. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 897–906.

John R Firth. 1957. A synopsis of linguistic theory, 1930-1955. *Studies in linguistic analysis*.

Luis Galárraga, Geremy Heitz, Kevin Murphy, and Fabian M Suchanek. 2014. Canonicalizing open knowledge bases. In *Proceedings of the 23rd acm international conference on conference on information and knowledge management*, pages 1679–1688.

Xianpei Han and Le Sun. 2016. Global distant supervision for relation extraction. In *Thirtieth AAAI Conference on Artificial Intelligence*.

Xu Han, Shulin Cao, Lv Xin, Yankai Lin, Zhiyuan Liu, Maosong Sun, and Juanzi Li. 2018. Openke: An open toolkit for knowledge embedding. In *Proceedings of EMNLP*.

Takaaki Hasegawa, Satoshi Sekine, and Ralph Grishman. 2004. Discovering relations among named entities from large corpora. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 415–422.

Nitisha Jain, Jan-Christoph Kalo, Wolf-Tilo Balke, and Ralf Krestel. 2021. Do embeddings actually capture knowledge graph semantics? In *European Semantic Web Conference*, pages 143–159. Springer.

Prachi Jain, Pankaj Kumar, Soumen Chakrabarti, et al. 2018. Type-sensitive knowledge base inference without explicit type supervision. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 75–80.

Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Knowledge graph embedding via dynamic mapping matrix. In *Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing (volume 1: Long papers)*, pages 687–696.

Zhengbao Jiang, Jun Araki, Donghan Yu, Ruohong Zhang, Wei Xu, Yiming Yang, and Graham Neubig. 2020. Learning relation entailment with structured and textual information. In *Automated Knowledge Base Construction*.

Jan-Christoph Kalo, Philipp Ehler, and Wolf-Tilo Balke. 2019. Knowledge graph consolidation by unifying synonymous relationships. In *International Semantic Web Conference*, pages 276–292. Springer.

Jerrold J Katz and Jerry A Fodor. 1963. The structure of a semantic theory. *language*, 39(2):170–210.

Tomáš Kliegr and Ondřej Zamazal. 2016. Lhd 2.0: A text mining approach to typing entities in knowledge graphs. *Journal of Web Semantics*, 39:47–61.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al. 2015. DBpedia–A large-scale, multilingual knowledge base extracted from Wikipedia. *Semantic Web*, 6(2):167–195.

Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion. In *Twenty-ninth AAAI conference on artificial intelligence*.

Farzaneh Mahdisoltani, Joanna Biega, and Fabian Suchanek. 2014. YAGO3: A Knowledge Base from Multilingual Wikipedias. In *7th Biennial Conference on Innovative Data Systems Research*. CIDR Conference.

Bonan Min, Shuming Shi, Ralph Grishman, and Chin-Yew Lin. 2012. Ensemble semantics for large-scale unsupervised relation extraction. In *Proceedings of the 2012 Joint Conference on Empirical Methods*

in Natural Language Processing and Computational Natural Language Learning, pages 1027–1037.

Tom Mitchell, William Cohen, Estevam Hruschka, Partha Talukdar, Bishan Yang, Justin Betteridge, Andrew Carlson, Bhavana Dalvi, Matt Gardner, Bryan Kisiel, et al. 2018. Never-ending learning. *Communications of the ACM*, 61(5):103–115.

Arvind Neelakantan and Ming-Wei Chang. 2015. Inferring missing entity type instances for knowledge base completion: New dataset and methods. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 515–525.

Arvind Neelakantan, Jeevan Shankar, Alexandre Passos, and Andrew McCallum. 2014. Efficient non-parametric estimation of multiple embeddings per word in vector space. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1059–1069.

Heiko Paulheim and Christian Bizer. 2013. Type inference on noisy rdf data. In *Int. Semantic Web Conf.*, pages 510–525.

Nils Reimers, Iryna Gurevych, Nils Reimers, Iryna Gurevych, Nandan Thakur, Nils Reimers, Johannes Daxenberger, Iryna Gurevych, Nils Reimers, Iryna Gurevych, et al. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Joseph Reisinger and Raymond Mooney. 2010. Multi-prototype vector-space models of word meaning. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 109–117.

Philip Resnik. 1997. Selectional preference and sense disambiguation. In *Tagging Text with Lexical Semantics: Why, What, and How?*

Yusuke Shinyama and Satoshi Sekine. 2006. Preemptive information extraction using unrestricted relation discovery. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 304–311.

Thuy Vu and D Stott Parker. 2016. K-embeddings: Learning conceptual embeddings for words using context-embeddings: Learning conceptual embeddings for words using context. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1262–1267.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Han Xiao, Minlie Huang, and Xiaoyan Zhu. 2016. Transg: A generative model for knowledge graph embedding. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2316–2325.

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2014. Embedding entities and relations for learning and inference in knowledge bases. *arXiv preprint arXiv:1412.6575*.

Ye Zhang and Byron C Wallace. 2017. A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 253–263.

Zhao Zhang, Fuzhen Zhuang, Meng Qu, Fen Lin, and Qing He. 2018. Knowledge graph embedding with hierarchical relation structure. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3198–3207.