Jain, Nitisha; Krestel, Ralf

Conference Paper — Published Version

Discovering Fine-Grained Semantics in Knowledge Graph Relations

Suggested Citation: Jain, Nitisha; Krestel, Ralf (2022) : Discovering Fine-Grained Semantics in Knowledge Graph Relations, In: CIKM '22: Proceedings of the 31st ACM International Conference on Information & Knowledge Management, ISBN 978-1-4503-9236-5, ACM, New York, NY, pp. 822-831, https://doi.org/10.1145/3511808.3557287

This Version is available at:
http://hdl.handle.net/11108/564

Kontakt/Contact
ZBW – Leibniz-Informationszentrum Wirtschaft/Leibniz Information Centre for Economics
Düsternbrooker Weg 120
24105 Kiel (Germany)
E-Mail: info@zbw.eu
https://www.zbw.eu/de/ueber-uns/profil-der-zbw/veroeffentlichungen-zbw

Discovering Fine-Grained Semantics in Knowledge Graph Relations

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

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

ABSTRACT

Knowledge graphs (KGs) provide structured representation of data in the form of relations between different entities. The semantics of relations between words and entities are often ambiguous, where it is common to find polysemous relations that represent multiple semantics based on the context. This ambiguity in relation semantics also proliferates KG triples. While the guidance from custom-designed ontologies addresses this issue to some extent, our analysis shows that the heterogeneity and complexity of real-world data still results in substantial relation polysemy within popular KGs. The correct semantic interpretation of KG relations is necessary for many downstream applications such as entity classification and question answering. We present the problem of fine-grained relation discovery and a data-driven method towards this task that leverages the vector representations of the knowledge graph entities and relations available from relational learning models. We show that by performing clustering over these vectors, our method is able to not only identify the polysemous relations in knowledge graphs, but also discover the different semantics associated with them. Extensive empirical evaluation shows that fine-grained relations discovered by the proposed approach lead to substantial improvement in the semantics in the Yago and NELL datasets, as compared to baselines. Additional insights from qualitative analyses convey that fine-grained relation discovery is an important yet complex task, especially in the presence of complex ontologies and noisy data.

CCS CONCEPTS

• **Computing methodologies** → **Knowledge representation and reasoning**; • **Information systems** → *Information retrieval*;

KEYWORDS

knowledge graphs, polysemous relations, fine-grained semantics, embeddings

ACM Reference Format:
Nitisha Jain and Ralf Krestel. 2022. Discovering Fine-Grained Semantics in Knowledge Graph Relations. In *Proceedings of the 31st ACM International Conference on Information and Knowledge Management (CIKM '22), October 17–21, 2022, Atlanta, GA, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3511808.3557287

1 INTRODUCTION

Knowledge graphs such as Yago [20], NELL [22] and DBpedia [18] are widely used for systematic representation of real-world 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. In 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 '*finger part of hand*'. As the triples in KGs are derived from and represent factual information from such texts, ambiguity from texts often makes it way into the KG triples as well. Specifically, the KG relations may represent multiple meanings depending on the *context*, which is defined by the types of the entities being connected by the relations in the case of KG triples.

Relation polysemy in KGs is a particularly important issue due to the widespread application of KGs in several downstream tasks such as search, question answering and reasoning where semantics play a crucial role. However, it has received surprisingly little attention until now. In order to gauge the magnitude of the issue in popular KGs, we analysed the relations in the Yago3 [20] dataset in terms of the number of unique entity type pairs that are connected by a single relation (in the KG triples) without any further semantic specialization. The results are plotted in Figure 1. It can be seen that for a majority of the relations, the triples in which they occur contain subject and object entities belonging to various entity types. Among these, many relations such as $owns$ and $created$ exhibit very high plurality of entity types which indicates that they are quite generic with regards to their meaning. Similar insights were also derived from the NELL dataset. Table 1 shows some examples of the different entity types associated with relations from these KGs.

In this work, we advocate that for such relations that are associated with a number of different entity type pairs that are semantically distant from one another, it would be prudent to replace them with sub-relations that have a more distinct meaning according to the context. The exact meanings of the sub-relations could be clearly defined based on the distinct types of the associated entities. Indeed, this underlying idea is derived from the task of word sense disambiguation in Linguistics, as advocated by Firth : 'a word is characterized by the company it keeps' [8]. In the context of KGs, one could say *'a relation is characterized by the entity types it connects'*.

CIKM '22, October 17–21, 2022, Atlanta, GA, USA
© 2022 Association for Computing Machinery.
ACM ISBN 978-1-4503-9236-5/22/10…$15.00
https://doi.org/10.1145/3511808.3557287

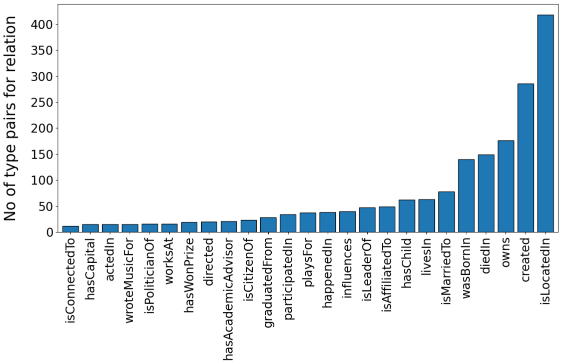


Figure 1: The number of unique type pairs associated with different relations in Yago

However, it is to be emphasized that while being intuitive, this task is extremely tricky due to a wide variance in the types of the entities in large KGs. Let us consider the relation *created* from the Yago dataset (Table 1). While some types such as *television* and *movie* for the *created* relation are semantically similar to one another, other types are quite different, for instance *company* and *writer*. If the relation is trivially replaced by multiple relations based on the different entity type pairs in a straightforward manner, without taking the similarity as well as frequency of these types into account, the resultant sub-relations would end up being remarkably similar to each other, thus leading to a high degree of duplication. Due to the complex hierarchy of classes (entity types)[1] in the underlying ontology, entity types often belong to different granularity levels [12], leading to a broad range of semantic similarity between them. The frequency with which a relation connects different type pairs is also widely variable. It is, therefore, a non-trivial task to decide how to define the sub-relations based on the semantics of the entity types associated with a relation, both in terms of the number of sub-relations as well the subset of entity types that the sub-relations should encompass.

Relational learning models have recently shown a lot of promise for the task of knowledge graph completion and refinement. In essence, these models aim to encapsulate the structure of the KG, as well as the latent semantics of entities and relations, by embedding them in low-dimensional vector space. Previous works have shown that the vector representations obtained from these models can be used for semantic analysis in KGs [14, 15]. We extend this idea further by demonstrating that these vectors also capture relation semantics such that they can be leveraged to successfully identify polysemous relations. The proposed method uses these vectors for finding representative clusters in the latent space and derive the fine-grained semantics from polysemous relations in an effective manner. To the best of authors' knowledge, the task of fine-grained relation semantics has not been systematically explored in the context of KG relations. This work establishes several feature-based baselines and shows the promise of embedding-based solution that outperforms a previous related non-embedding approach in the context of open relation extraction [21].

Table 1: Examples of Multiple Semantics of Relations

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

The main contributions of this paper are following: (1) We formally define the task of *fine-grained relation discovery* which refers to the disambiguation of polysemous relations in knowledge graphs and motivate its importance and benefits. (2) We propose a data-driven and scalable method *FineGReS (**Fine**-**G**rained **R**elation **S**emantics)* to identify multiple sub-relations that capture the different underlying semantics of the relations via clustering in the latent space. (3) We present detailed discussion and empirical evaluation on two large KG datasets (Yago and NELL) that illustrates the effectiveness of the approach as compared to several baselines for downstream applications.[2]

2 FINE-GRAINED RELATION SEMANTICS

Relation polysemy is quite common in knowledge graphs due to 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. However, due to the diversity of the kinds of associations between the entities, the abstract relations may not be sufficiently representative of the underlying semantics that they are supposed to capture. 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, however 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 use cases as discussed further in this section.

[1] The terms *class* and *entity type* will be used interchangeably in the rest of the text.

[2] The code and data is available at https://github.com/nitishajain/FineGReS.

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 [5, 11, 30]. 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) [6] 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.

It is to be noted that the discovery of fine-grained relation semantics is important in the context of *KG refinement*, not being merely limited to already existing datasets, but also 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 or can 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. Finally, it is also important to note that the approach of determining semantic sub-relations in existing KGs and their ontology can be applied to the very important open challenge of constructing *domain-specific KGs* from new corpora [16]. By way of adding novel relations to already existing ontologies, this work shows the potential and promise of aiding *ontology matching* [25] for supporting new domains.

3 RELATED WORK

Relation Semantics. While the idea of learning embeddings for *words* by considering their multiple contextual semantics is not new [32], the contextual semantics of existing *relations* in knowledge graphs have not been studied as much. This is due to the fact that most KGs are populated on the basis of a pre-defined ontology where the relations and their semantics have already been fixed [20, 22]. Yet, issues with the relations in such KGs still persist. Kalo et al. [15] have previously presented a detailed analysis on finding and unifying synonymous relations that are found in most large KGs to reduce the number of relations for the sake of better semantics. Similar in spirit, we bring attention to the complementary problem statement of identifying the relations in KGs that exhibit more than one meaning based on different contexts and claim that they should be represented by multiple sub-relations with more precise semantics.

In other related work, [14] 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.

KG Embeddings. In the context of relational learning models, few works have looked into KG relations for the goal of learning better embeddings. For instance, in [19] 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 [39], 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 [13] 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 [35] the authors discussed the generation of multiple translation components of relations based on their semantics with the help of a bayesian non-parametric 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, previous works have only discussed the semantics of KG relations in the context of KG embeddings with the primary goal of training better models that can show improvement on the link prediction (or knowledge graph completion) task. However, this work explicitly pays attention to the identification of polysemous relations in the KGs and discovery of the latent relation semantics with the overall goal of knowledge graph refinement and improvement of the quality of the relations in underlying ontology. Relational models have been leveraged as effective and promising enablers for this task instead of being the focal topic of this work. 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. We present a systematic and data-driven method for this task.

Relation Extraction and Open IE. While this work is concerned with relations between entities, it is important to distinguish it from the task of relation extraction from texts. There are many previous approaches that identify relationships between entities in texts and perform clustering on phrases to derive the relations [2, 23, 28, 33], such approaches aim to identify relation patterns that exactly conform to a singular semantic intent. In stark contrast, we aim to find the different semantic intents that may be already present in a single KG relation. Moreover, relation extraction techniques heavily rely on the contextual cues available in the text, whereas the only context available with regard to the relations in KGs is the associated entities and their types. As such, these approaches are indeed not comparable to our work.

Research pertaining to the processing of entity and relation phrases in the context of Open Information Extraction is more relatable to our goals. Previous approaches on the canonicalization of relation phrases (that are present in Open IE triples) have attempted to establish the semantics of the relations by performing clustering over the phrases [9, 31]. Among such approaches, the closest to ours is the work by Min et al. [21] that discusses the ambiguity in the meanings of relation phrases present in Open IE triples such as $\langle Euro, be\ the\ currency\ of, Germany\rangle$ and $\langle authorship, be\ the\ currency\ of, science\rangle$. While this approach concerns with disambiguation of relation phrases in texts rather than relations in KGs, we still consider this work as a baseline approach that does not employ embeddings for deriving the semantics and compare our embeddings-based approach against it.

4 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 $\mathcal{E}$. A 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 $\mathcal{P}_r$. Thus we have, $\mathcal{P}_r = \{\langle t_h, t_t\rangle | \langle e_h, typeOf, t_h\rangle, \langle e_t, typeOf, t_t\rangle, \langle e_h, r, e_t\rangle \in \mathcal{G}_r\}$. The total number of such unique type pairs for relation r 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. E.g. in *TransE* [3], for a triple $\langle h, r, t\rangle$ the vectors $\mathbf{h}$, $\mathbf{r}$ and $\mathbf{t}$ satisfy the relation $\mathbf{h} + \mathbf{r} = \mathbf{t}$ or $\mathbf{r} = \mathbf{t} - \mathbf{h}$. 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 $\boldsymbol{e_h}$, $\mathbf{r}$ and $\boldsymbol{e_t}$ as obtained from an embedding corresponding to a KG triple $\langle e_h, r, e_t\rangle$, we define a vector $\boldsymbol{\Delta_r}$ which is a function of $\boldsymbol{e_h}$ and $\boldsymbol{e_t}$. Further, every $\boldsymbol{\Delta_r}$ vector is mapped to a type pair $\langle t_h, t_t\rangle$ corresponding to the entities e_h, e_t.

Problem Definition. Given a relation $r \in \mathcal{R}$ in $\mathcal{G}$, the set of vectors $\{\boldsymbol{\Delta_{r_1}}\boldsymbol{\Delta_{r_2}}...\boldsymbol{\Delta_{r_{\mathcal{G}_r}}}\}$ for the graph $\mathcal{G}_r$ and the set of type pairs for this relation as denoted by $\mathcal{P}_r$, the goal is to find an optimal configuration of clusters $C_{opt} = \{C_1, C_2...C_N\}$, where the $\boldsymbol{\Delta_{r_i}}$ vectors are uniquely distributed among the clusters i.e. each $\boldsymbol{\Delta_{r_i}} \in C_j$, $i = 1...|\mathcal{G}_r|$, $j = 1...N$, s.t. an objective function $\mathcal{F}(C_{opt})$ is maximized. Further, each cluster C_j represents the semantic union of a *subset of type pairs* from $\mathcal{P}_r$ such that $\exists\ \boldsymbol{\Delta_{r_i}} \in C_j$ where $\boldsymbol{\Delta_{r_i}}$ is mapped to one of the type pairs in this subset. 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.

5 *FineGReS*

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

5.1 Semantic Mapping for Facts

For every unique relation r in $\mathcal{G}$, we firstly find the subset of triples $\mathcal{G}_r$ 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 $\mathcal{P}$[3]. At this stage, every triple in $\mathcal{G}_r$ is associated with a type pair $\langle t_h, t_t\rangle \in \mathcal{P}$ that represents the semantics of this triple. 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.

5.2 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 [14], we derive a representation for the relation from $\boldsymbol{e_h}$ and $\boldsymbol{e_t}$ vectors corresponding to every triple in $\mathcal{G}_r$. In this way, for every relation r, a set of vectors $\boldsymbol{\Delta_r}$ is obtained from the KG embeddings, in addition to the actual $\mathbf{r}$ vector that the embedding already provides. These $\boldsymbol{\Delta_r}$ vectors are then mapped to the corresponding type pairs (according to the types of the underlying entities). With this, each unique type pair is, in turn, mapped to and represented by a subset of $\boldsymbol{\Delta_r}$ vectors. The $\boldsymbol{\Delta_r}$ vectors encode the combined information conveyed by the head and tail entity types and represent the relationship between the entities, thus encapsulating the latent semantics of the relations in different triples. The $\boldsymbol{\Delta_r}$ vectors serve as the data points for the clustering (with the associated type pairs being their labels).

[3] We denote $\mathcal{P}_r$ as $\mathcal{P}$ when the relation r is clear from the context.

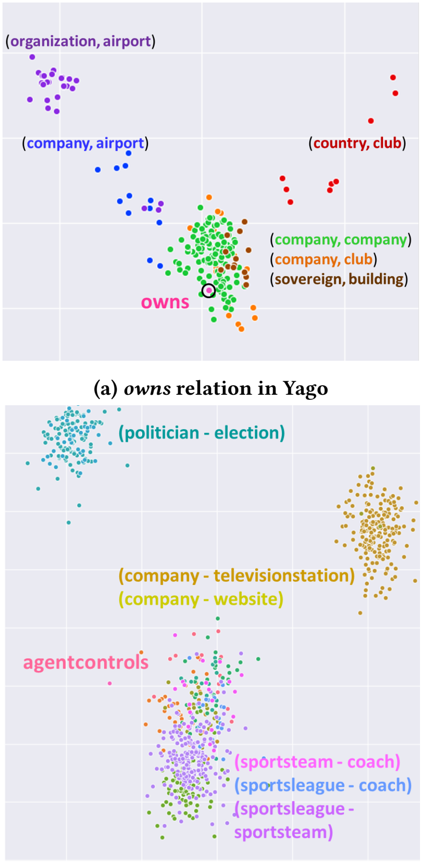
Relation Semantics. While it is believed that KG embeddings are able to capture relation similarity in the embedding space, i.e., relations having similar semantics occur close together in the vector space [7, 15], we found that relations having multiple semantics (based on the context of their entities) are, in fact, not represented well in the vector space. 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 $\mathbf{r}$ 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. Figure 2 shows examples from the NELL and Yago datasets 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 $\boldsymbol{\Delta_r}$ vectors that represent the distinct latent semantics exhibited by different entity type pairs connected by the same relation, as described next.

5.3 Clustering for Fine-grained Semantics

For each relation r, the total number of unique type pairs $\mathcal{L} = |\mathcal{P}|$ is theoretically the maximum number of possible semantic sub-relations or clusters that could be obtained for r. This will create 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 triples that it appears in, at the same time, a naive assignment of sub-relations pertaining to all 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 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. For this, clustering is performed in an iterative manner with a predefined number of clusters and combinations of type pairs within each cluster for the iterations. Since it is not feasible or practical to consider an exhaustive number of possible clusters, *FineGReS* leverages the *semantic similarity of type pairs* to narrow down the search space for obtaining the optimal clusters. First, the vector representations for the types are derived. Subsequently, the similarity scores between all combinations of the unique type pairs (t_{h_i}, t_{t_i}), (t_{h_j}, t_{t_j}) are obtained by calculating the similarity scores between the vectors corresponding to the head entity types t_{h_i} and t_{h_j} as well as the tail entity types t_{t_i} and t_{t_j} and then taking their mean value.

Iterative Clustering. The iterative clustering begins with $\mathcal{L}$ clusters, with each cluster corresponding to one type pair for the relation in the first iteration. At this point, the cluster labels for the data points ($\boldsymbol{\Delta_r}$ vectors) are denoted by individual type pairs directly and serve as the 'ground truth' for evaluation. Next, the similarity scores of all the type pair combinations are calculated, and the two type pairs that are most similar are considered as candidate pairs to be merged together and placed in a single cluster for the second iteration. To generate the cluster labels, the data points corresponding to the candidate type pairs are assigned the same distinct label (that could be generated e.g. by combining the individual label names). The number of clusters is given as $\mathcal{L}$ - 1 during the second iteration of clustering, and the cluster labels consist of $\mathcal{L}$ - 2 original type pairs and the 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 type pairs as candidates for merging in the next iteration to reduce the number of clusters is repeated until all type pairs have been gradually merged back together in a single cluster. In some iterations, the most similar type pairs could be already in the same cluster, so the next most similar pairs are considered until candidates to merge are found. This ensures that the number of clusters always shrinks in subsequent iterations until eventually all clusters are merged back and the algorithm converges. At each iteration, the quality of the clusters is calculated (as detailed in 6.2) and this is regarded as the function $\mathcal{F}(C_{opt})$. The results from the iteration



(a) *owns* relation in Yago

(b) *agentcontrols* relation in NELL

Figure 2: Visualization (after PCA reduction) of relation vectors with associated type pairs

having the maximum value of this function is chosen as the optimal configuration of clusters C_{opt}. The complexity of this algorithm for a relation is proportional to the number of unique type pairs in the dataset and in practice, the run time of iterative clustering process ranges between a few seconds to a few minutes per relation. It is to be noted that while this approach discovers the sub-relations, their labeling is a separate task on its own. In this work, we simply use the type pairs to derive representative labels, e.g. a sub-relation of *created* that connects *company* with *computer_game* could be named as *created-company-computer_game* and so on. However, a proper naming scheme for these relations is concerned with the task of ontology design and is out of the scope of the current work.

6 EVALUATION

We evaluate the effectiveness of our proposed method by performing a series of experiments with several feature-based baselines and a non-embedding baseline approach, as well as variations of the *FineGReS* technique with different embedding models and clustering techniques. In particular, the experiments will help to answer the following research questions —
Q1. Does *FineGReS* approach find meaningful and useful fine-grained relation semantics as compared to baselines? (Section 6.2)
Q2. Do the sub-relations really reflect what the users need in terms of semantics? (Section 6.3)
Q3. Do fine-grained relation semantics benefit a KG-based application such as entity classification? (Section 6.4)
The experiments are also supported by a qualitative analysis and detailed discussion of the results.

6.1 Experimental Setup

Datasets. We prepared datasets derived from Yago3 and NELL-995 knowledge graphs for the experiments. For the Yago3 dataset, the entity types (concepts) of all entities were extracted from the accompanying ontology and ranked in terms of frequency. Yago ontology is composed of concepts that are derived from Wordnet as well as from Wikipedia categories (Wikicat). Since the Wikicat concepts are often fine-grained sub-classes of Wordnet concepts, we only consider Wordnet concepts for obtaining non-overlapping clean set of concepts. We considered the top 53 frequent concepts for creating our dataset as the frequencies dropped considerably thereafter. The accounted concepts each had atleast 10,000 entities associated with them. Thereafter, we extracted the facts triples from Yago3 that were comprised of subject(head) and object(tail) entities associated with the chosen concepts. This resulted in a set of 1,492,078 triples, which were augmented with the corresponding types of entities. The final dataset consists of 31 relations and 917,325 unique entities. Note that only the data points from the relations having multiple type pairs (after filtering out the ones having too few triples to avoid errors from incorrect entity type mapping) associated with them were considered for clustering.

A similar process was followed for the NELL-995 dataset. In this dataset, the type information is embedded with the entities and thus could be directly extracted from the data triples. Similar to the above heuristics, the types of the entities were restricted to the most frequent types (top 41) found in the dataset (with the less frequent types being replaced by their more frequent supertypes when found in the NELL ontology[4]). The numerical entities were removed from the dataset since they did not have an associated type. The final dataset consists of a total of 200 relations and 75,492 entities along with their corresponding types, and 154,213 triples in total.

Finding Type Similarity. The process of iterative clustering is guided by the semantic similarity of the different type pairs for a given relation and therefore, obtaining the representations for the entity types is an important step in the *FineGReS* method. Here, we describe two different strategies to derive these representations — *concept-based embeddings* and *entity-based embeddings.*

Concept-based Type Representations. In order to directly obtain vector representations of the entity types, we use the pre-trained ConVec embeddings [29] that are publicly available.[5] These 300-dimensional embeddings were obtained by training over a dataset of 1.5 million words including the Wikipedia *concepts* and thus represent the semantics for the entity types quite well. While the ConVec embeddings work well in most cases, sometimes the entity types are multi-word phrases, especially in the case of NELL dataset. In addition, the NELL ontology is quite large with a much wider vocabulary due to the continuous learning paradigm of NELL. As such, in order to obtain the vector representations that are not found in ConVec and to calculate the similarity scores between the entity types (including both words and phrases), we leveraged the pre-trained *Sentence-BERT* [27] models from the HuggingFace library [34].

Entity-based Type Representations. The entities associated with the types can also provide meaningful representations for the entity types. For each type, we first obtain the vectors for the corresponding entities from Wikipedia2Vec tool [36]. Since the Wikipedia2Vec embeddings were derived from the mentions of the entities on the entire Wikipedia corpus, they effectively encapsulate the textual semantics of the entities. The vector representation for the entity type was then obtained by taking the average of the entity embedding vectors. Once the type vectors were obtained from either of the above strategies, the *cosine* similarity measure was used for calculating the similarity matrix between the entity types pairs.[6]

Knowledge Graph Embeddings. We perform our experiments on the following widely used KG embedding models — *TransE* [3] and *DistMult* [37]. 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 [4] for Yago3-10 dataset and from the OpenKE library [10] for NELL-995 dataset.

Clustering Techniques. Several different clustering algorithms were employed to obtain the clusters in the vector space — KMeans clustering (KMC), Optics (OPC) and Hierarchical Agglomerative clustering (HAC).

[4] Available at - http://rtw.ml.cmu.edu/rtw/resources
[5] https://github.com/ehsansherkat/ConVec
[6] We also tried the *euclidean* similarity measure and it shows very similar results. For the rest of the paper, we only refer to results from the *cosine* similarity scores.

Table 2: Performance of *FineGReS* clusters for *TransE* and *DistMult* embedding models with different clustering techniques in comparison with feature-based baselines (*best F1 values in bold, best for a model on each dataset underlined*).

		TransE						*DistMult*					
		KMC		HAC		OPC		KMC		HAC		OPC	
		Micro	Macro	Micro	Macro	Micro	Macro	Micro	Macro	Micro	Macro	Micro	Macro
Yago	*subject*	.359	.258	.358	.242	.0013	.0001	.426	.260	.434	.258	.0007	.00003
	object	.213	.167	.187	.112	.0009	0	.248	.136	.339	.167	.0007	.00007
	pair	.153	.095	.162	.064	.0012	**.0002**	.140	.266	.058	.095	.0007	.00006
	$FineGreS_{entity}$	.472	.337	**.527**	**.339**	**.0014**	**.0002**	.519	.321	.532	.337	**.0008**	**.00008**
	$FineGreS_{concept}$	**$\underline{.537}$**	**$\underline{.357}$**	.525	.329	.0011	**.0002**	**$\underline{.597}$**	**$\underline{.376}$**	**.582**	**.347**	**.0008**	**.00008**
NELL	*subject*	.217	.125	.256	.159	.006	.0026	.281	.155	.255	.151	.004	.0009
	object	.302	.209	.286	.176	.006	.0017	.291	.204	.323	.204	.005	.0006
	pair	.128	.083	.132	.070	.005	.0033	.089	.051	.137	.079	.005	.0013
	$FineGreS_{entity}$	.345	.178	.467	.210	.007	.0034	**.686**	.194	.454	.207	**.006**	**.0014**
	$FineGreS_{concept}$	**.576**	**.379**	**$\underline{.711}$**	**$\underline{.434}$**	**.008**	**.0039**	.376	**.387**	**$\underline{.719}$**	**$\underline{.431}$**	**.006**	**.0014**

Table 3: Performance of *FineGReS* clusters in comparison with the non-embedding baseline (*best values in bold*)

	Yago		NELL	
	Micro	Macro	Micro	Macro
Baseline	.439	.275	.442	.261
TransE	.537	.357	.711	**.434**
DistMult	**.597**	**.376**	**.719**	.431

Baselines. To the best of our knowledge, there is no existing research that has leveraged knowledge graph embeddings to discover fine-grained semantics of relations in large KGs. As such, we establish several baselines in this work for analysis and comparison of our proposed approach as well as future works.

Feature-based Baselines. To derive sub-relations from a polysemous relation in the KG, several simplistic configurations were explored. The semantics can be driven by the entity types of solely subject or object entities. The different type pairs can also be a criteria for new sub-relations. Hence, we define the baselines as —
pair - Sub-relations obtained on the basis of every unique type pair that is associated with a relation, this setting corresponds to the maximum number of sub-relations.
subject - Sub-relations created by grouping the type pairs by subject entity types i.e. each sub-relation represents all type pairs associated with a common subject type and different object types.
object - Similar to *subject*, but grouping instead by the object entity types.

Non-embedding Baseline. Few previous works have discussed the ambiguity in the meanings of relation phrases present in Open IE triples [9, 21]. Particularly, in [21] the authors propose *'Type A'* relations where the same relation phrase is associated with different types of subject and object entities, hence denoting different semantics. Such polysemous relation phrases are indeed identified as distinct relations through a variant of the Hierarchical Agglomerative Clustering(HAC) technique. Note that this approach heavily relies on the textual context of the entities and relations which is missing in KG triples. Nevertheless, as this is the closest related approach to our work, we consider it as a non-embedding baseline that is purely text-driven. To best implement this baseline approach from [21], the entity similarity was derived from text-driven entity embeddings [36] instead of the KG embedding models (as done in our approach). These text-driven embeddings encapsulate the textual context as well as sentence-level lexical patterns available in Wikipedia texts via word-based skip gram and anchor context models. An entity similarity matrix (corresponding to the entity similarity graph in [36]) was thus constructed from these entity embeddings and the clustering was performed based on the pairwise similarity values from this matrix to obtain relations with distinct semantics.

6.2 Evaluation of *FineGReS* Relation Semantics

Following previous works related to relation phrase clustering [9, 31], we employ micro and macro metrics to evaluate the quality of the clustering in terms of precision, recall and F1. Table 2 reports the weighted (as per the number of data points) F1 metrics for the datasets obtained by the feature-based baselines and the *FineGReS* method in the different settings of KG embeddings and clustering techniques. Note that $FineGreS_{concept}$ and $FineGreS_{entity}$ correspond to the different variations of the *FineGReS* approach in terms of obtaining type representations (refer to Section 6.1). Additionally, we also compared the best performing setting of the *FineGReS* method in case of *DistMult* and *TransE* models to the non-embedding baseline and present the results in Table 3.

Observations. It can be seen that in all settings the clusters obtained by the proposed *FineGReS* method outperform the baselines in terms of both micro and macro metrics on Yago and NELL datasets. Clustering with *kmeans* and *hierarchical agglomerative* techniques show better results in comparison with *optics* which is a density-based clustering technique.[7] Overall, the results provide strong evidence in support of the efficacy of our method for finding optimal configurations of clusters for the relations, from

[7] This was also observed by previous work in the context of KG embeddings [12].

Table 4: Examples of Fine-grained Sub-Relations

Dataset - Relation (Setting)	Count	*FineGReS* Sub-Relations
Yago - *owns* (*TransE*-HAC)	3	{⟨company, airport⟩ ⟨organization, airport⟩}, {⟨sovereign, building⟩}, {⟨company, club⟩ ⟨company, company⟩, ⟨country, club⟩}
Yago - *created* (*TransE*-OPC)	4	{⟨artist, medium⟩ ⟨officeholder, movie⟩}, {⟨writer, fictional_character⟩}, {⟨writer, movie⟩ ⟨writer, television⟩ ⟨writer, fictional_character⟩ ⟨artist, movie⟩ ⟨artist, computer_game⟩ ⟨player, movie⟩}, {⟨company, computer_game⟩}
NELL-*agentCompetesWith* (*TransE*-KMC)	5	{⟨company, person⟩ ⟨website, person⟩ ⟨person, person⟩, ⟨sportsteam, sportsteam⟩}, {⟨person, company⟩, ⟨person, website⟩} {⟨animal, animal⟩, ⟨bird, animal⟩}, {⟨bank, bank⟩}, {⟨mammal, politicsissue⟩}
NELL-*subpartOfOrganization* (*DistMult*-KMC)	8	{⟨sportsteam, sportsteam⟩ ⟨stateorprovince, sportsteam⟩ ⟨university, sportsteam⟩ ⟨city, sportsteam⟩ }, {⟨organization, organization⟩}, {⟨televisionstation, city⟩}, {⟨company, company⟩ ⟨televisionstation, company⟩}, {⟨sportsteam, sportsleague⟩}, {⟨bank, bank⟩} {⟨televisionstation, televisionnetwork⟩}, {⟨televisionstation, website⟩}

which sub-relations with well-defined semantics can be derived. Furthermore, it is observed that $FineGreS_{concept}$ performs better than $FineGreS_{entity}$ in majority of the cases for both Yago and NELL datasets. We conjecture this is due to the fact that the semantics of the entity types directly obtained from ConVec vectors (see Section 6.1) are more precise, whereas, the semantics derived from the vectors of the entities associated with the types are prone to noise and errors. Therefore, the type similarities would be more semantically aligned in the case of $FineGreS_{concept}$, thereby leading to superior performance of the method. Another important insight from the results is that while it is indeed favorable to replace a polysemous relation with multiple sub-relations, it is certainly not a trivial task to obtain these sub-relations by simply defining their semantics in terms of unique type pairs. The *pair* baseline that corresponds to such sub-relations can be seen to score consistently lower in all settings. The *subject* and *object* baselines fair better in this regard, though the proposed *FineGReS* approach is clearly the most optimal. From Table 3, it can be seen that the non-embedding baseline was outperformed by *FineGReS* with the exception of *TransE* giving better result for NELL dataset. As mentioned, this baseline benefits from textual context which is lacking for our approach and therefore, a fare comparison is hard to perform. Still, the results indicate that KG embeddings are able to represent the semantics of the relations and identify fine-grained relation semantics in large KGs, even in the absence of additional cues or background knowledge.

Qualitative Results. Table 4 shows a few representative examples of the sub-relations, along with their count, 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 ⟨*sovereign*, *building*⟩ pair for *owns* relation where *sovereign* is semantically distant from other types or *agentCompetesWith* where ⟨*bank*, *bank*⟩ 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 for a relation could indeed correspond to the *pair* or *subject*/*object* baseline depending on the associated type pairs. The *FineGReS* method is able to automatically determine this optimal configuration of the sub-relations for each relation relying solely on the triples in the KG dataset and the associated entity type information.

6.3 Manual Evaluation with Yago

In order to estimate the usefulness of the fine-grained sub-relations obtained from *FineGReS*, we performed a limited manual evaluation and analysis on the Yago dataset. Three annotators were given the different type pairs associated with 15 candidate relations in Yago having more than two distinct type pairs) and asked to independently identify any potential sub-relation clusters by assigning labels to the type pairs. The relations for which atleast two annotators agreed on the label assignments were taken into consideration as the true values. These were then compared with the labels obtained from the top k best performing *FineGReS* settings from Table 2 for each relation and the *Hits@k* metric was calculated. Essentially, we measure how often the sub-relations identified by human annotators for each relation were also found by the proposed technique among the top k performers. The values of *Hits@1* and *Hits@3* were found to be *0.33* and *0.66* respectively, indicating that the sub-relations discovered by *FineGReS* indeed resembled the semantics that the human annotators had identified to be useful for many of the relations. The manual evaluation proved to be challenging due to the subjective nature of this task, where humans could not always identify the precise semantics of potential sub-relations in the absence of additional context. Embeddings derived from relational learning models are superior in this regard as they are able to encapsulate the latent semantics of the KG relations, hence they are well-suited to the task of fine-grained relation discovery.

Discussion. A closer inspection of the sub-relations obtained from *FineGReS* revealed further interesting insights. First of all, due to the data-driven nature of the proposed approach, where only KG triples serve as data points, the results are worse for relations with a smaller representation in terms of the number of triples in the dataset, as compared to the relations with a larger number of triples. This is quite expected as the clustering algorithms fail to

Table 5: Performance Comparison for Entity Classification Task for Yago and NELL (*R refers to original relations, Base refers to the non-embedding baseline*)

							FineGReS	
		R	*pair*	*subject*	*object*	*Base*	*TransE*	*DistMult*
Yago	P	.893	.916	.906	.918	**.926**	.923	**.928**
	R	.908	.925	.921	.935	.938	**.941**	**.942**
	F1	.894	.914	.909	.924	.926	**.931**	**.931**
NELL	P	.643	.692	.696	.665	.567	**.705**	**.713**
	R	.689	.727	.729	.703	.645	**.736**	**.747**
	F1	.650	.701	.713	.683	.584	**.715**	**.726**

identify good clusters in the vector space when there are very few data points available. Along the same lines, it is important to point out that if a type pair has few data points but it is semantically distinct from the others, it is still identified as a separate cluster, e.g. in the case of $\langle bank, bank\rangle$ type pair for the *agentCompetesWith* relation in NELL (Table 4). This way, the semantics of the type pairs for a relation play a decisive role in the clustering, rather than the number of data points (i.e. frequency with which the relation connects the different type pairs). Furthermore, it was seen that the proposed approach is rather too aggressive for some relations, where there might be different entity types associated with the relation but they still represent the same semantic. Especially in Yago, while relations such as *lives_In* and *married_To* convey a clear meaning, due to the hierarchical ontology structure, the entities associated with these relations form different type pairs such as *(officeholder, country)* and *(scientist, site)* in the case of *lives_In* relation. Therefore, the *FineGReS* approach discovers separate sub-relations for these relations despite the same semantic. In the same dataset there is another relation *participatedIn* where the types *officeholder* and *scientist* play distinctly different roles and indeed belong to separate sub-relations. As the mapping of the entities to their types is performed consistently for all the triples in the dataset, and not on a per-relation basis, the proposed method cannot distinguish the cases where the entities such as *officeholder* and *scientist* should be abstracted to represent the *person* type, as a human annotator would understand. Related to this discussion, it is noteworthy that the assignment of the types to the entities can be differently performed in the underlying dataset depending on the required level of granularity. The proposed method can, in principle, work at different levels of fine-grained semantics as dictated by richness of type assignment of the entities in the hierarchy of the ontology or as desired by a downstream application.

6.4 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 [24]. 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 [26], textual information [17] as well as embeddings [1]. Taking cue from the same, we design a simple architecture with a CNN classifier [38] for the multi-label classification task which can jointly classify both the entities in a given triple to their respective types.[8] The model consists of a convolutional layer with feature detector, and ReLu activation, this is followed by a max pooling layer and dropout layer to reduce over-fitting. The output is passed through a fully connected layer with softmax activation to obtain the probability of the different classes for being the predicted type for the entities. The Adam optimizer was used with the learning rate set to 0.0001. The experiments were run on a server with Intel X86 CPU and using a single NVIDIA GTX1080 GPU with 11GB RAM. 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 relevant triples, obtained from the best performing setting of the *FineGReS* method as well as from the baseline techniques described in Section 6.1. The performance of entity classification measured in terms of weighted precision, recall and F1 scores (averaged over 10 runs) is shown in Table 5 for Yago and NELL. The main objective is to measure the improvement in performance when the relations in the triples of the KG are dictated by well-defined, fine-grained semantics as opposed to ambiguous semantics. The results confirm that entity classification task indeed sees an improvement when the underlying dataset is comprised of relations with fine-grained semantics obtained from *FineGReS* method, in comparison to the original polysemous relations (denoted as *R* in the tables), as well as the relations obtained from other feature-based and non-embedding baselines. In particular, the gains seen over the *pair* 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 the relations.

7 CONCLUSION

In this paper, we have presented the task of fine-grained relation discovery in knowledge graphs, which is an important problem that has not been fully explored. We have proposed a scalable and data-driven method *FineGReS* that automatically determines an optimal configuration for deriving fine-grained sub-relations by taking advantage of the latent relation semantics represented by KG embedding models. This technique does not rely on additional background knowledge and thus it can be employed for arbitrarily large and heterogeneous KGs. We established several baselines and conducted extensive empirical evaluation that demonstrated the difficulty of this task and the efficacy of the proposed method for learning fine-grained relation semantics. The improved performance for the task 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 approach to derive relation 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 tasks such as relation extraction and question answering over KGs.

[8] The setup is intentionally simple in these experiments so as to draw attention to the effect on performance from different configurations of relations and pseudo sub-relations in the KG dataset. It could arguably be replaced by any state-of-the-art technique.

REFERENCES

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

[2] Danushka Tarupathi Bollegala, Yutaka Matsuo, and Mitsuru Ishizuka. 2010. Relational duality: Unsupervised extraction of semantic relations between entities on the web. In *Proceedings of the 19th International Conference on World Wide Web*. 151–160.

[3] 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*. 2787–2795.

[4] Samuel Broscheit, Daniel Ruffinelli, Adrian Kochsiek, Patrick Betz, and Rainer Gemulla. 2020. LibKGE - A Knowledge Graph Embedding Library for Reproducible Research. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. 165–174.

[5] Jinxiu Chen, Donghong Ji, Chew Lim Tan, and Zhengyu Niu. 2005. Unsupervised Feature Selection for Relation Extraction. In *Companion Volume to the Proceedings of IJCNLP Conference including Posters/Demos and tutorial abstracts*. https://aclanthology.org/I05-2045

[6] Wanyun Cuix, Yanghua Xiao, et al. 2017. KBQA: Learning question answering over QA corpora and knowledge bases. In *Proceedings of the VLDB Endowment*, Vol. 10. 656–676.

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

[8] John R Firth. 1957. A synopsis of linguistic theory, 1930-1955. *Studies in linguistic analysis* (1957).

[9] Luis Galárraga, Geremy Heitz, Kevin Murphy, and Fabian M Suchanek. 2014. Canonicalizing open knowledge bases. In *Proceedings of the 23rd ACM International Conference on Information and Knowledge Management*. 1679–1688.

[10] 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*.

[11] 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)*. 415–422.

[12] Nitisha Jain, Jan-Christoph Kalo, Wolf-Tilo Balke, and Ralf Krestel. 2021. Do Embeddings Actually Capture Knowledge Graph Semantics?. In *Proceedings of the 2021 Extended Semantic Web Conference*. Springer, 143–159.

[13] 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*. 687–696.

[14] 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 *Proceedings of the 2020 Conference on Automated Knowledge Base Construction*.

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

[16] Mayank Kejriwal. 2019. *Domain-specific knowledge graph construction*. Springer.

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

[18] 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 (2015), 167–195.

[19] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence*.

[20] Farzaneh Mahdisoltani, Joanna Biega, and Fabian Suchanek. 2015. YAGO3: A Knowledge Base from Multilingual Wikipedias. In *Proceedings of the 7th Biennial Conference on Innovative Data Systems Research (CIDR 2015)*.

[21] 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*. 1027–1037.

[22] 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. *Commun. ACM* 61, 5 (2018), 103–115.

[23] Thahir Mohamed, Estevam Hruschka, and Tom Mitchell. 2011. Discovering relations between noun categories. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*. 1447–1455.

[24] 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*. 515–525.

[25] Lorena Otero-Cerdeira, Francisco J Rodríguez-Martínez, and Alma Gómez-Rodríguez. 2015. Ontology matching: A literature review. *Expert Systems with Applications* 42, 2 (2015), 949–971.

[26] Heiko Paulheim and Christian Bizer. 2013. Type inference on noisy RDF data. In *Proceedings of the International Semantic Web Conference*. 510–525.

[27] 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.

[28] Sebastian Riedel, Limin Yao, Andrew McCallum, and Benjamin M Marlin. 2013. Relation extraction with matrix factorization and universal schemas. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 74–84.

[29] Ehsan Sherkat and Evangelos E Milios. 2017. Vector embedding of wikipedia concepts and entities. In *Proceedings of the International Conference on Applications of Natural Language to Information Systems*. Springer, 418–428.

[30] 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*. 304–311.

[31] Shikhar Vashishth, Prince Jain, and Partha Talukdar. 2018. Cesi: Canonicalizing open knowledge bases using embeddings and side information. In *Proceedings of the 2018 World Wide Web Conference*. 1317–1327.

[32] 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*. 1262–1267.

[33] Koki Washio and Tsuneaki Kato. 2018. Neural Latent Relational Analysis to Capture Lexical Semantic Relations in a Vector Space. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. 594–600.

[34] 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* (2019).

[35] 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)*. 2316–2325.

[36] Ikuya Yamada, Akari Asai, Jin Sakuma, Hiroyuki Shindo, Hideaki Takeda, Yoshiyasu Takefuji, and Yuji Matsumoto. 2020. Wikipedia2Vec: An Efficient Toolkit for Learning and Visualizing the Embeddings of Words and Entities from Wikipedia. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. 23–30.

[37] Bishan Yang, Scott Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding Entities and Relations for Learning and Inference in Knowledge Bases. In *Proceedings of the International Conference on Learning Representations (ICLR) 2015*.

[38] 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 8th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. 253–263.

[39] 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*. 3198–3207.