Relation Extraction for Knowledge Base Completion: A supervised approach

Héctor Cerezo-Costas[1] and Manuela Martín-Vicente[1]

Gradiant, AtlantTIC, Edificio CITEXVI, local 14, Universidade de Vigo, Spain
http://www.gradiant.org

Abstract. This paper outlines our approach for the extraction on predefined relations from unstructured data (OKE Challenge 2018: Task 3). We train a deep learning classifier on data aligned between DBPedia relations and Wikipedia pages.

Keywords: Relation Extraction · Classification · Deep Learning

1 Introduction

In relation extraction from unstructured text the objective is to identify semantic connections between two entities in a short piece of text. This sub-problem considers that the entities were previously detected, for example, with a named entity recognizer. In theory, this relation must be inferred from the context surrounding the entities although extra information might be available (e.g. entity links with external open data services such as DBPedia [1]). This problem is interesting because automatic relation extraction could help to add new information to linked data services or create new ones. It is also de basis of many question-answering solutions [2].

Previous approaches to this problem in the state of the art used bootstrapping from a seed of relations from DBPedia to obtain new patterns that carry semantic information of a relation (e.g. $(Person, marry, Person)$ implies $(Person, spouse, Person)$) [4]. A two step approach was followed in [3]. Taking the relations from Freebase as basis, the authors first search the patterns that typically connect two entities for a relation. Afterwards, those patterns are used to find new entities that match those patterns in raw text in order to augment the database. Other methods employ weak supervision [7] or distant supervision [6] to overcome the problem of the absence of manually labelled ground truth.

2 Approach

2.1 Architecture

The simplified architecture of the system is depicted in Figure 2. Although the system consumes and produces NIF-RDF documents, internally the information is preprocessed to be used by the classification engine. The text with the query

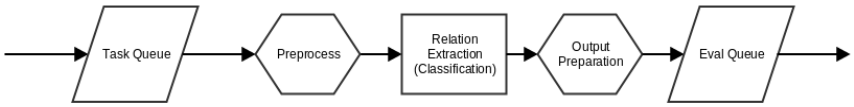

Fig. 1. Architecture of the system.

is tokenized. As the engine only obtains relations for every pair of entities, we take as input every possible combination of two entities.

Entity names in the query are substituted by a token containing the role and type of the entity, if known (e.g. *"David Beckham married Victoria Adams in 1999"* is substituted to *"Subject_Person married Object_Person in 1999"*). As we cannot know the role of an entity in a relation beforehand we try the two possible combinations for each pair of entities.

2.2 Classification

The relation identification is modelled as a classification problem. The core of the engine is a deep learning classifier that uses self attention [5]. We use an implementation in *pytorch* [1] that we adapted to add extra capabilities that were not initially present in the system (such as the activation function selection or treating the problem as a query-text problem [2]).

After the model assigns a probability for each possible relation, a heuristic determines the relations that will be finally assigned to a pair of entities. For example the heuristic knows which relations are allowed for each pair of entities and which are not allowed. Since we substitute real names of entities with a wildcard with the role and type of the entity in the preprocessing stage, some relations are difficult to distinguish only from the context (e.g. *Country* vs *Location*). We add rules for adding/modifying some relations when a given entity is in a list of countries or when a more generic relation could apply (e.g. adding *location* if *birthPlace* relation is detected). The objective of these heuristics are to improve the output of the system or to overcome some limitations of our approach (e.g. only one relation could be assigned to a given pair of entities).

3 Model Training

3.1 Data preparation

One of the biggest challenges we have to confront is that we need a big corpus to train our deep learning supervised model. Manually creating such corpus is extremely costly in time and effort, and therefore it is unfeasible in most cases.

We follow a process in several steps with little human intervention to obtain a corpus of considerable size:

[1] Pytorch http://pytorch.org/

[2] This last capability was discarded due to its bad performance.

- **Gathering triples from DBPedia**. We use public DBPedia dumps for obtaining the initial corpus for training. We extract triples from the file *mappingbased_objects_en.ttl*. Only those triples with relations in the benchmark are filtered for training. From this subset, we extract the list of unique entities that take part in at least one relation of the corpus.
- **Obtaining data types of entities**. We use the file *instance_types_en.ttl* to obtain the data type of an entity. Data types are further processed to take the type of highest order following the DBPedia ontology (*Person*, *Organisation* and *Place*). We use an extra wildcard type to tag those entities whose type is unknown.
- **Extracting sentences from Wikipedia pages**. Finally, for each pair of entities linked by a relation, we extract their respective Wikipedia pages and select the sentences in which both entities appear. We allow partial matching of entities with certain restrictions (e.g. we consider *Beckham* for *David Beckham* as a positive match but we do not allow the match *North* for *North America*). Entities in the sentences are substituted by tags representatives of the role of the entity in the relationship (*subject* or *object*).

After a careful inspection of the corpus, we detected that some sentences did not carry the meaning of the relationship. That was especially true with relationships indicative of location (e.g. *birthPlace* or *deathPlace*). For example, the subject could study or work in a city that was her birthplace but this sentence did not contain, explicitly or implicitly, any useful information about this topic. Hence, an extra filtering step is executed in the corpus. We set a seed of words representative of each relationship and try to find iteratively other frequent words (adjectives, verbs and nouns) co-occurring with the seeds that might act as triggers. At the end of each iteration we manually filter out the words that are erroneously marked as triggers. Triggers might be shared among different relationships (e.g. *bandMember* and *formerBandMember* have many trigger words in common).

Those trigger words are finally used to refine the original corpus to obtain the final training data. Figure 2 shows the distribution of records of the training data (1261165 records). As it can be clearly seen, relations relative to places (*Country*, *birthPlace* or *location*) dominate the corpus.

3.2 Training

Given two entities with their types and a short text in which both entities appear, we model the problem as a multiclass classification one in which the objective is to extract the best relationship among a possible set. We consider 32 classes (31 relationships and one extra label that means no relationship - *norel*).

For each training register we exchange the roles of the entities and mark it as "norel". However there are two special cases. Two of the relationships are bidirectional: *spouse* and *relative*. Furthermore, there are two pairs of relationships comprising opposing relationships (*child-parent* and *doctoralStudent-doctoralAdvisor*). We take that into account when transforming the data for training. Table 1 shows one example of each type.

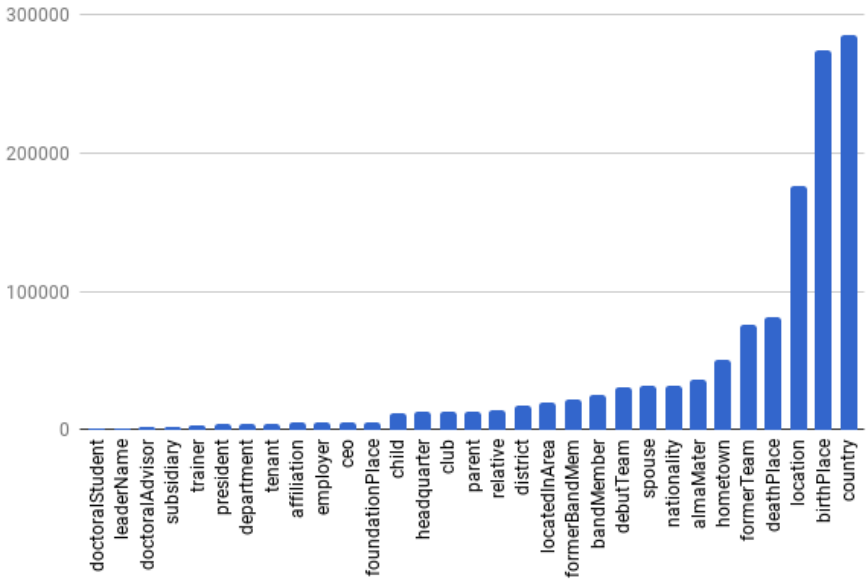

Fig. 2. Distribution of relations in the training corpus.

The most relevant parameters of the model are summarized in Table 2. Due to the highly unbalanced training data, we apply weights to the different classes during training. The algorithm converged after only six epochs.

Table 1. Training samples

Description	Label
Subject_Person died in Object_Place in August 2001 at the age of 86	deathPlace
Object_Person died in Subject_Place in August 2001 at the age of 86	norel
Subject_Person was Object_Person 's third wife	spouse
Object_Person was Subject_Person 's third wife	spouse
In 1943, Subject_Person disowned his daughter Object_Person for marrying the English actor director and producer Charlie Chaplin.	child
In 1943, Object_Person disowned his daughter Subject_Person for marrying the English actor director and producer Charlie Chaplin.	parent

4 Conclusion

In this paper we present our approach to the relation extraction Task 3 of the OKE Challenge 2018. The core of the system is a deep learning engine that clas-

Table 2. Classification Parameters

Parameter	Value	Parameter	Value
Words in dictionary	300.000	Embedding size	300
Hidden Units (GRU)	300	Attention Units	150
Attention Hops	30	Dropout	0.5
Class Number	32	Batch Size	32
Activation Function	Relu	Input size (classif. layer)	1500
Learning Rate (Adam)	0.001	Number of Layers (GRU)	2

sifies the relation between a pair of detected entities. Finding data for training that system is very complicated and much of our effort went in that direction. Although we employed more than 1 million records for training, they were not completely revised by a human and they are far from being ideal. Training records are still very unbalanced and some of them are very biased (e.g. existing records of the relation *trainer* in DBPedia were about professional fighting whereas the figure of trainers exists in nearly all sports).

Another improvement of the system will be the inclusion of a correference system in the analytical loop. On one hand, the identification of multiple references to the same entity in a document will help with the extraction of more training records. On the other hand it will avoid some mistakes of the system when it assigns a relation between two instances of the same entity in a sentence.

References

1. Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., Ives, Z.: Dbpedia: A nucleus for a web of open data. In: The semantic web, pp. 722–735. Springer (2007)
2. Bordes, A., Usunier, N., Chopra, S., Weston, J.: Large-scale simple question answering with memory networks. arXiv preprint arXiv:1506.02075 (2015)
3. Cannaviccio, M., Barbosa, D., Merialdo, P.: Accurate fact harvesting from natural language text in wikipedia with lector. In: Proceedings of the 19th International Workshop on Web and Databases. p. 9. ACM (2016)
4. Exner, P., Nugues, P.: Entity extraction: From unstructured text to dbpedia rdf triples. In: The Web of Linked Entities Workshop (WoLE 2012). pp. 58–69. CEUR (2012)
5. Lin, Z., Feng, M., Santos, C.N.d., Yu, M., Xiang, B., Zhou, B., Bengio, Y.: A structured self-attentive sentence embedding. In: Proceedings of the 5th International Conference on Learning Representations (ICLR) (2017)
6. Mintz, M., Bills, S., Snow, R., Jurafsky, D.: Distant supervision for relation extraction without labeled data. In: Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2. pp. 1003–1011. Association for Computational Linguistics (2009)
7. Weston, J., Bordes, A., Yakhnenko, O., Usunier, N.: Connecting language and knowledge bases with embedding models for relation extraction. pp. 1366–1371 (2013)