Extracting Relations in Texts with Concepts of Neighbours

Hugo Ayats, Peggy Cellier, and Sébastien Ferré

Univ Rennes, INSA, CNRS, IRISA
Campus de Beaulieu, 35042 Rennes, France
{hugo.ayats,peggy.cellier,sebastien.ferre}@irisa.fr

Abstract. During the last decade, the need for reliable and massive Knowledge Graphs (KG) increased. KGs can be created in several ways: manually with forms or automatically with Information Extraction (IE), a natural language processing task for extracting knowledge from text. Relation Extraction is the part of IE that focuses on identifying relations between named entities in texts, which amounts to find new edges in a KG. Most recent approaches rely on deep learning, achieving state-of-the-art performances. However, those performances are still too low to fully automatize the construction of reliable KGs, and human interaction remains necessary. This is made difficult by the statistical nature of deep learning methods that makes their predictions hardly interpretable. In this paper, we present a new symbolic and interpretable approach for Relation Extraction in texts. It is based on a modeling of the lexical and syntactic structure of text as a knowledge graph, and it exploits *Concepts of Neighbours*, a method based on Graph-FCA for computing similarities in knowledge graphs. An evaluation has been performed on a subset of TACRED (a relation extraction benchmark), showing promising results.

1 Introduction

During the last decade, the need for reliable and massive knowledge bases, represented as Knowledge Graphs (KG), increased. KGs allow to structure, organize and share knowledge. A challenge is to build KGs that are at the same time reliable and large. There exist several ways to create those KGs: manually with forms (e.g., wikidata[1]), providing reliability (assuming the producer is reliable), or automatically by using Information Extraction (IE) techniques [8], allowing to easily build very large KG by using the large amount of existing textual data (e.g., scientific papers, books, official websites). IE is a natural language processing task for extracting knowledge from text. Relation Extraction is a sub-task of IE that focuses on identifying relations between named entities in texts, which amounts to find new edges between KG entities. It is a classification task, where given two named entities in a sentence, the goal is to predict the relation type between the two.

[1] https://www.wikidata.org/

Among existing approaches for the relation extraction task, the deep learning methods are the most efficient. First, those method were based on convolutional neural networks [15]. Other deep learning approaches, such as [17], work on pruned dependency trees. Today, methods based on language models such as BERT [18] provide the best results. However, those deep learning methods have two main limitations. First, they are not sufficiently reliable for a full automation. Second, they suffer, as most of numerical approaches, of a lack of explanations for predictions, which hinders human interaction.

Formal Concept Analysis (FCA) has already been shown effective for classification tasks [9], and has the advantage to provide symbolic representations of predictions that can be used as explanations. However, it faces a problem of tractability when all concepts need to be computed. An interesting approach is to adopt lazy learning, where this computation is delayed until there is an instance to be classified, and only the concepts that are relevant to that instance are computed. This lazy approach was applied early to the incremental building of a logical context [4], then later advocated in [10] with Pattern Structures [6], and applied, for instance, to relation extraction in biomedical texts [11]. More recently, this has been formalized as *Concepts of Neighbours* [2] based on Graph-FCA [1]. The particularity of the Concepts of Neighbours method, such as this contribution, is the use of an efficient anytime algorithm based on the partitioning of the set of entities into concepts. This has for consequence that there is no need for a sampling strategy: all entities appear in the extensions of the returned concepts. The use of Concepts of Neighbours has been applied to the completion of knowledge graphs [3], and has shown competitive results compared to deep learning approaches, and state-of-the-art results compared to rule-based approaches such as AnyBURL [13] or AMIE+ [5]. Rule-based approaches are also interpretable but they are not lazy because they compute a set of rules before seeing any test instance. Therefore, they have to strongly restrict the considered graph patterns for combinatorial reasons, unlike in Concepts of Neighbours where all connected graph patterns with constants are supported. AnyBURL only mines path rules and AMIE+ focuses on small connected and closed rules.

In this paper, we introduce a symbolic and lazy approach for relation extraction based on Concepts of Neighbours. A first contribution is the representation of sentences by graphs, covering both lexical and syntactic information. Other contributions, compared to previous application of Concepts of Neighbours, are that the instances to be classified are couples of graph nodes, instead of single nodes, and that node labels are organized into a taxonomy (e.g., word hypernyms). We validate our approach with experiments on TACRED [20], a dataset for evaluating relation extraction methods.

In the sequel, Section 2 gives preliminaries about Graph-FCA and Concepts of Neighbours. Then, Section 3 introduces the modeling of sentences as graphs. Section 4 details how Concepts of Neighbours can be used for relation extraction. Finally, Section 5 presents the experiments conducted on dataset TACRED.

$O = \{Charles, Diana, William, Harry, Kate, George, Charlotte, Louis, male, female\}$
$A = \{parent, spouse, female, male\}$
$I = \{parent(\{William, Harry\}, \{Charles, Diana\}),$
$parent(\{George, Charlotte, Louis\}, \{William, Kate\}),$
$spouse(Charles, Diana), spouse(William, Kate),$
$male(\{Charles, William, Harry, George, Louis\}),$
$female(\{Diana, Kate, Charlotte\})\}$

Fig. 1: Example Graph-FCA context $K = (O, A, I)$ describing part of the British royal family. Notation $p(\{a, b\}, \{c, d\})$ stands for $p(a, c), p(a, d), p(b, c), p(b, d)$.

2 Preliminaries

In this section, we recall the main definitions and results of *Concepts of Neighbours* [3]. We start by defining *graph contexts* and *graph concepts*, introduced in Graph-FCA [1], a generalization of Formal Concept Analysis (FCA) [7] to graphs. A *graph context* $K = (O, A, I)$ is a labeled and directed multi-hypergraph, where objects are nodes, attributes are edge labels, and incidence elements are edges (noted like atoms in predicate logic $a(o_1, \ldots, o_k)$). As a running example, Figure 1 defines a small context describing (part of) the British royal family.

Definition 1. *A* graph concept *is defined as a pair $C = (R, Q)$, where R is a set of k-tuples of objects and Q is a conjunctive query such that $R = res(Q)$ is the set of results of Q, and $Q = msq(R)$ is the most specific query that verifies $R = res(Q)$. R is called the* extension *$ext(C)$, and Q is called the* intension *$int(C)$.*

The most specific query $Q = msq(R)$ is the conjunctive query representing what the tuples of objects in R have all in common. For the sake of simplicity, we restrict the following examples to 1-tuples, aka. *singletons*. In the running example, the singletons $(William)$ and $(Charlotte)$ have in common the following query, Q_{WC}, that says that both have married parents:

$$\begin{aligned} Q_{WC} &= msq(\{(William),\ (Charlotte)\}) \\ &= (x) \leftarrow parent(x, y),\ female(y),\ parent(x, z),\ male(z),\ spouse(y, z). \end{aligned}$$

We have $R_{WC} = res(Q_{WC}) = \{(William), (Harry), (George), (Charlotte), (Louis)\}$ so that $C_{WC} = (R_{WC}, Q_{WC})$ is a graph concept.

A concept $C_1 = (R_1, Q_1)$ is more specific than a concept $C_2 = (R_2, Q_2)$, in notation $C_1 \leq C_2$, if $R_1 \subseteq R_2$. For example, a concept more specific than C_{WC} is the concept of *the children of Kate and William*, whose extension is $\{(George), (Charlotte), (Louis)\}$, and whose intension is:

$$(x) \leftarrow parent(x, y), (y = Kate), parent(x, z), (z = William).$$

The total number of graph concepts in a knowledge graph is finite but in the worst case, it is exponential in the number of objects, and in arity k. It is therefore

not feasible in general to compute the set of all concepts. Instead of considering concepts generated by subsets of tuples, we consider concepts generated by pairs of tuples, and use them as as a symbolic form of distance between objects.

Definition 2. *Let $t_1, t_2 \in O^k$ be two k-tuples of objects. The* conceptual distance $\delta(t_1, t_2)$ *between t_1 and t_2 is the most specific graph concept whose extension contains both tuples, i.e. $\delta(t_1, t_2) = (R, Q)$ with $Q = msq(\{t_1, t_2\})$, and $R = res(Q)$.*

For example, the above concept C_{WC} is the conceptual distance between $(William)$ and $(Charlotte)$. The "distance values" have therefore a symbolic representation through the concept intension Q that represents what the two tuples have in common. The concept extension R contains in addition to the two tuples all tuples t_3 that match the common query ($t_3 \in res(Q)$). Such a tuple t_3 can be seen as "between" t_1 and t_2: in formulas, for all $t_3 \in ext(\delta(t_1, t_2))$, $\delta(t_1, t_3) \leq \delta(t_1, t_2)$ and $\delta(t_3, t_2) \leq \delta(t_1, t_2)$. Note that order $\leq$ on conceptual distances is a partial ordering, unlike classical distance measures.

A numerical distance $dist(t_1, t_2) = |ext(\delta(t_1, t_2))|$ can be derived from the size of the concept extension, because the closer t_1 and t_2 are, the more specific their conceptual distance is, and the smaller the extension is. For example, the numerical distance is 5 between $(William)$ and $(Charlotte)$ (see C_{WC}), and 3 between $(George)$ and $(Charlotte)$.

The number of conceptual distances $\delta(t_1, t_2)$ is no more exponential but quadratic in the number of objects $|O|$. In the context of lazy learning, tuple t_1 is fixed, and the number of concepts become linear. For k-tuples, that number is bounded by $|O|^k$. Those concepts are called *Concepts of Neighbours.*

Definition 3. *Let $t \in O^k$ be a k-tuple of objects. The* Concepts of Neighbours *of t are all the conceptual distances between t and every tuple $t' \in O^k$.*

$$C\text{-}N(t, K) = \{\delta(t, t') \mid t' \in O^k\}$$

Figure 2 shows the 6 Concepts of Neighbours of the singleton $(Charlotte)$, and their partial ordering as a Venn diagram. For instance, the concept containing $(Charlotte)$ only is included in the concept that also contains $(Louis)$ and $(George)$, which is included in the concept that also contains $(William)$ and $(Harry)$. This implies that George is semantically closer to Charlotte than William is. Although $(Louis)$ and $(Diana)$ are both nearest neighbours of $(Charlotte)$, and at the same extensional distance 3, they are so for different reasons as they belong to different Concepts of Neighbours with different intensions. Louis has the same parents as Charlotte, while Diana has the same gender.

The *proper extension* of a concept of neighbours δ_l is the part of its extension that does not appear in sub-concepts. The proper extensions define a partition over the set of objects O, where two objects are in the same proper extension if and only if they are at the same conceptual distance. For instance, Diana and Kate are in the same proper extension.

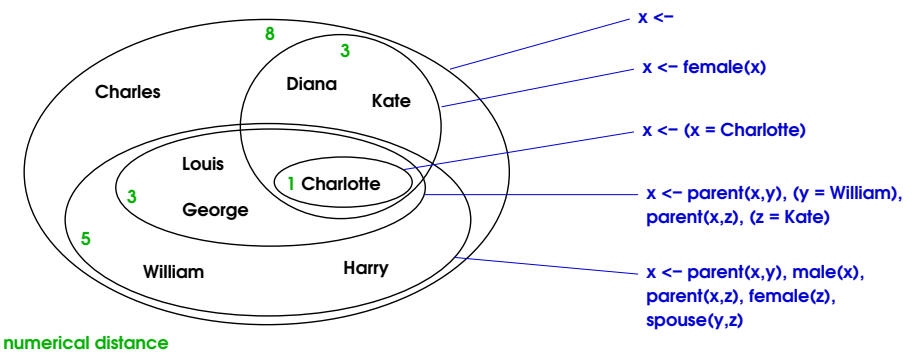

Fig. 2: Venn diagram of the extensions of the 6 Concepts of Neighbours of (*Charlotte*), labelled by their intension (right) and numerical distance (inside).

3 Modeling Sentences as Graphs

The input data of relation extraction is a set of annotated sentences. Each sentence is annotated by two entities, the subject and object of the relation, and by the type of those entities. The input sentences are split in two parts: training sentences for which the relation is known, and test sentences for which the relation is to be predicted. In this section we explain and discuss the modeling of annotated sentences as graphs because this is the required input of Concepts of Neighbours, and more precisely as RDF graphs because this is the expected input of the existing implementation of Concepts of Neighbours.

3.1 NLP Treatments

Before building the RDF graph, several NLP treatments are applied on each sentence[2]. First, the sentence is split into tokens. Second, *part-of-speech* (POS) tags[3] and lemmas are computed for each token. Third, the syntactic structure of the sentence is extracted as a dependency tree[4]. Finally, named entities are identified, and a named entity type is associated to each of them.

Table 1 shows the result of the processing of the sentence *"The University of Rennes is French"*. For example, it shows that the 4th token is *Rennes*, has for lemma *Rennes* and for POS tag *NNP* (a proper noun). It is part of a named entity of type *ORGANIZATION*. It has for parent in the dependency tree the 2nd token and it is linked to this token via relation *nmod* (linking an nominal modifier to its parent noun). We then apply a few post-treatments in order to simplify and improve the sentence representation.

[2] We use the *CoreNLP* tool [12] but other tools could be used.
[3] We use the 58 POS tags of English Penn Treebank [16].
[4] We use the dependency grammar proposed by Treebank Universal Dependencies.

Table 1: Example of a processed sentence.

ID	token	lemma	POS	NER	head	deprel
1	The	the	DT	-	2	det
2	University	University	NNP	ORGANIZATION	6	nsubj
3	of	of	IN	ORGANIZATION	4	case
4	Rennes	Rennes	NNP	ORGANIZATION	2	nmod
5	is	be	VBZ	-	6	cop
6	French	french	JJ	NATIONALITY	-	ROOT
7	.	.	.	-	6	punct

Removing punctuation. Punctuation tokens (e.g., token 7 in Table 1) and their links are removed from the dependency tree. This is easy as they only occur as leaves. Note that the parser takes into account punctuation when extracting the dependency tree.

Compound named entities. A named entity can overlap several contiguous tokens, for instance *University of Rennes* overlaps tokens 2-4 in the example. However, a named entity is a semantic unit: it holds its own meaning, which can be very different from the meaning of its individual tokens. Therefore, manipulating a named entity as a succession of tokens can cause an important loss of semantics. Except when there is a parse error, the tree structure of an entity is a subtree of the dependency tree. We call it a *factor* by analogy with the definition of a string factor. The proposed solution is to collapse the subtree into its root. Then the sentence retains a valid syntactic and semantic structure (no dangling link for instance). For example, in the sentence presented in Table 1, the named entity *"University of Rennes"* is collapsed into token 2, and tokens 3 and 4 disappear. The expression *"University of Rennes"* can indeed be seen as a proper noun (POS tag *NNP*). In case of a parse error, the named entity is collapsed to the last token as a fallback.

3.2 Sentences as an RDF Graph

In order to model a set of processed sentences as one RDF graph, each token is represented by an RDF node (e.g., `id:1_2` for the 2nd token of the 1st sentence), and each dependency link is represented by an RDF edge. The lemmas, POS tags, and named entity types of a token are represented by RDF types on the corresponding node (see discussion below). Figure 3 gives the RDF representation for the example in Table 1. The 2nd token is modeled by node *id:1_2*, which has as types lemma *University of Rennes*, POS tag *NNP*, and named entity type *organization*, and is linked to node *id:1_6* by relation *nsubj*.

Representation of lemmas and POS tags as RDF types. As specified above, we use relation *rdf:type* instead of defining specific relations for linking a node

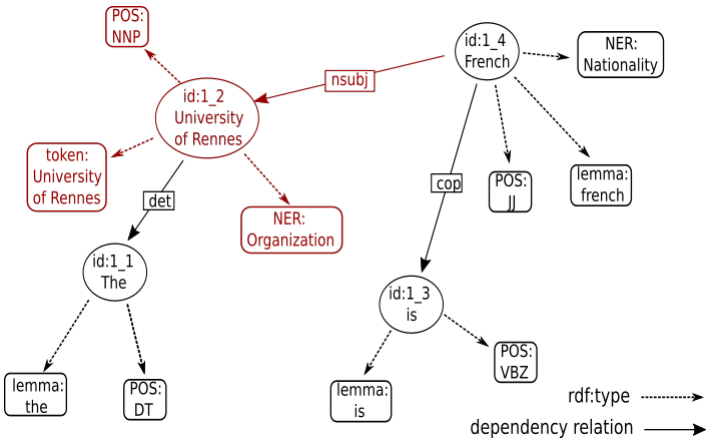

Fig. 3: Example of a sentence modeled as an RDF graph.

to its lemma or POS tag. This choice has been made for three reasons. First, by having lemmas and POS tags represented by RDF types rather than RDF nodes, we avoid to have two sentences get connected as soon as they share a lemma or a POS tag. Second, for the computation of Concepts of Neighbours, it prevents from having intensions including dummy patterns such as *"has an unspecified POS tag"*. Third, as presented in Section 3.3, it allows us to create a type hierarchy for lemmas and POS tags.

Lemmatisation of named entities. Note that, if in the general case the lemma of a token is a good representation, it does not stand in the case of named entities: e.g., *unite state of America* vs *United States of America*. Therefore, for named entities, we use the original words instead of the lemmas in the RDF graph.

Optimization of the modeling. Although the algorithm computing Concepts of Neighbours is anytime, the size of the RDF graph has an impact on the number of computed concepts, and hence on the quality of predictions. We can prune the RDF graph according to the position of the subject and object, in a way that reduces its size without loosing too much information. Indeed, Zhang et al. [19] states that not all dependencies are of same interest for extracting relations. Only those close to the path between the subject and the object carry useful information. However, it can be easily seen that reducing the dependency tree to a path would remove essential information for relation extraction, e.g. in the case of a negation attached to a verb that is on the path. Our solution is to prune the dependency tree to keep only the path between the subject and the object, plus the tokens up to maximal distance K from this path. Several values of K were tested, and the value $K = 1$ appears to be a good trade-off between size reduction and performance.

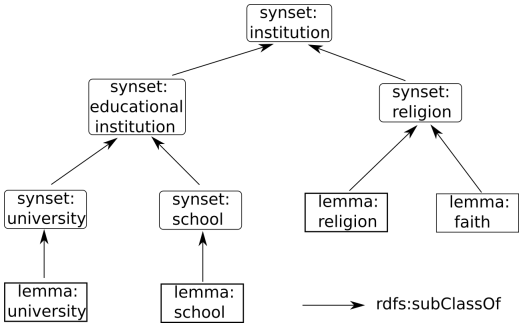

Fig. 4: Fragment of a type hierarchy obtained from WordNet

3.3 Type Hierarchies

An RDF graph can be enriched with inferred types and edges by declaring domain knowledge. The most common form of domain knowledge is hierarchies of types, based on the RDFS property `rdfs:subClassOf`. The inference rule is that: if node X has type A and A is a subclass of B, then X also has type B. We use several type hierarchies to increase the generalization power of Concepts of Neighbours. First, the set of POS-tags [16] is fine-grained enough to create a hierarchy on a few POS-tags: for example, the gerund of a verb (POS-tag *VBG*) is a subclass of verb (POS-tag *VB*). In total, we have 11 `rdfs:subClassOf` declarations[5].

Second, in order to add semantic knowledge to the modeling, the lexical database WordNet [14] is used for creating a lemma hierarchy for nouns and verbs. Each synset of a lemma is considered as a superclass of the lemma, and each hypernym of a given synset is considered as a superclass of this synset. Figure 4 shows a fragment of this lemma hierarchy. For instance, synset *educational institution* allows to generalize over lemmas *university* and *school*. The lemma hierarchy thus increases the chance to find similarities between sentences using words that have different lemmas but close meanings.

4 Relation Extraction with Concepts of Neighbours

Once the modeling is done, for each test example we want to compute which examples of the training corpus are the more similar, and do a prediction from their annotation. In order to do so, an RDF graph regrouping the modeling of all the sentences of the dataset is made, the Concepts of Neighbours method is used to group the examples by similarities, and then a decision method is used to do a prediction from those Concepts of Neighbours. In the following, we first

[5] Full hierarchy at `https://gitlab.inria.fr/hayats/jena-conceptsofneighbours/-/blob/master/src/conceptualKNN/utils/postag.ttl`

present how the Concepts of Neighbours have been adapted to this specific task. Then, we describe two decision methods for making prediction from Concepts of Neighbours.

4.1 Concepts of Neighbours for Relation Extraction in Texts

In order to use Concepts of Neighbours on the modeling presented in Section 3, a few aspects needs to be addressed. First, we need to clearly identify the RDF nodes representing the subject and the object in each sentence. Then, we need to compute Concepts of Neighbours on a $(subject, object)$ couple, compared to a single node in previous applications. This strongly increases the number of potential neighbours. Finally, we show how to reduce this number of in the specific case of relation extraction.

Identification of the Subjects and Objects There is a need to unambiguously identify the nodes of the RDF graph forming the $(subject, object)$ couples. The issue is that subjects or objects can overlap several tokens. The collapse of the named entities presented in Section 3.1 solves this problem in the vast majority of cases as subjects and objects are most of the time either named entities or one-token expressions. However, ambiguous cases still exist in a small proportion of the sentences. This problem can appear in three cases: first, when the subject or object is expressed as a nominal group (e..g, *"the man"*, *"the university"*); second, when the subject or object includes a named entity but is longer than it (e.g., *"the President of the United States of America"* is a subject, whereas only *"United States of America"* is tagged as a named entity); third, when the subject or object is a named entity that has not been recognized as a named entity.

The solution proposed to solve these cases is similar to the solution presented in Section 3.1 to collapse named entities: as a subject or an object is necessarily a group of contiguous tokens with a particular meaning, it must form a tree factor in the dependency tree. Therefore, it can be considered that the root of this factor carries the semantic and syntactic information, and then can be pointed out as the subject or the object. Like for named entities, if the tokens do not form a tree factor because of a wrong annotation or a parse error, the last token is used instead of the root. It can be pointed out that, unlike for named entities, the choice has been made not to merge the tokens constituting the subject or the object as their syntactic structure (if any) is generally informative, like in *"the President of X"*.

Concepts of Neighbours for Couples of Nodes The Concepts of Neighbours of the identified couples $(subject, object)$ are then computed. The method was originally designed to generate Concepts of Neighbours for tuples of arbitrary size. However, until now, it was only applied for unary concepts. The switch from unary concepts to binary ones (and by extension n-ary) has two main consequences discussed in the following.

Intension. First of all, unlike unary concepts, binary concepts can have an intension that is not connected. For example, in the sentence presented in Figure 3, both the connected intension (relating the object to the subject via dependency *nsubj*):

$$(s, o) \leftarrow nsubj(o, s), UnivRennes(s), french(o), cop(o, x), is(x)$$

and the disjoint one (no path between subject and object):

$$(s, o) \leftarrow UnivRennes(s), french(o), cop(o, x), is(x)$$

can appear during the computation of the Concepts of Neighbours. In both cases, the pruning strategy presented in Section 3.2 ensures that the intension focuses on and around the path between the subject and the object.

Reduction of the Set of Couples. Another issue is the large number of potential neighbours: as detailed in [3], to compute the Concepts of Neighbours for a tuple of k objects in a graph involving n objects, the algorithm has to generate and partition n^k tuples. Therefore, for a large graph, the computation of concepts of arity greater than 1 is rapidly intractable. In the present case, if we consider a dataset composed of ten thousands of sentences, there are tens of billions of potential neighbours. However, the use of the Concepts of Neighbours method is in this context for extracting relations by comparing a $(subject, object)$ couple from a test sentence to annotated couples from the training dataset, and it appears that the number of annotated couples is far smaller: only one per example in the training dataset. Therefore, in the following, we use this set of couples for the computation of Concepts of Neighbours, as it permits to simultaneously reduce drastically the computation cost and remove noise from the computed concepts while keeping all the knowledge of interest.

In addition, as evoked in Section 3, in a relation extraction dataset each subject and object has a type, and these types can be used to reduce the set of potential neighbours further. For example, if an example has for subject a person and for object a location, it can be seen that the relation expressed by this example could be *place_of_birth* or *place_of_living*, but can not be *age* or *parent*. Therefore, for a given couple $(subject\_type, object\_type)$, a set of compatible relations can be deduced from the training dataset. If there is only one compatible relation, this relation can be predicted without computing Concepts of Neighbours for this example, and if there are several possible relations, the set of $(subject, object)$ couples from the training dataset that are annotated with compatible relation types can be used as the set of possible neighbours in the algorithm.

4.2 Scoring Methods

The computation of Concepts of Neighbours of a *(subject, object)* pair from the test dataset returns a set of concepts, each concept is associated to a set of neighbour couples, and to an extensional distance. In addition, the specialization

presented in the previous section ensures that each neighbour couple is annotated with a compatible relation type. From this result, in order to be able to predict a relation, we need to associate a score to each relation type. In the following we present two scoring methods: one based on a weighted votation, and another based on the confidence measure.

Exponential-weighted Vote (EV). Each neighbour (s, o) of each concept "votes" for its annotated relation type $r(s, o)$. However, if not weighted this method will only reflect the proportion of each relation among the training examples annotated with a compatible relation. To avoid this problem, the extensional distance $dist(c)$ of concept c can be used to weight each vote. The extensional distance of a concept of neighbours measures the degree of similarity between the couple from which the concept has been computed and the neighbours that the concept contains: the lower the distance, the higher the similarity. Therefore, each vote is weighted by a decreasing function of the extensional distance. We use the following formula to score a relation type r based on a set of Concepts of Neighbours C.

$$score(r, C) := \sum_{c \in C} \sum_{(s,o) \in proper(c)} w(c)\ \mathbf{1}_{r(s,o)=r} \quad \text{where } w(c) = e^{-dist(c)}$$

We have chosen the inverse exponential function to define each weight $w(c)$ because of its rapid decrease, which privileges nearest neighbours. This way, the relation of one very similar example is preferred to the relation of a large number of vaguely similar examples.

Maximum Confidence (MC). The second method is similar to the method used by AnyBURL [13], and has been successfully used with Concepts of Neighbours for link prediction [3]. The idea is to consider the intension $int(c) = (s, o) \leftarrow P_c$ of each concept c, to use pattern P_c as the body of a rule, and for each relation type r to compute the confidence of the rule $\mathbf{R}_{c,r} : P_c \rightarrow r(s, o)$, defined as usual as:

$$conf(\mathbf{R}_{c,r}) = \frac{|\{(s, o) \mid r(s, o)\} \cap ext(c)|}{|ext(c)|}$$

For each relation type r, the score is the list of the confidences of all rules $\mathbf{R}_{c,r}$ predicting that relation, in descending order.

$$score(r, C) := (conf(\mathbf{R}_{c,r}))_{c \in C} \text{ in descending order}$$

Such scores are ranked according to inverse lexicographic ordering. That is, the predicted relation type is the relation type with the higher maximal confidence. If several relation types have the same maximal confidence, the relation type with the higher second maximal confidence is predicted, and so on.

5 Experiments

In this section we present the experiments conducted on TACRED [20], a standard relation extraction benchmark, to evaluate the proposed method.

5.1 Dataset and Baseline

TACRED is a dataset made of 106,264 annotated examples, split into a training corpus (68,124 examples), a development corpus (22,631 examples) and a test dataset (15,509 examples). Each example is a sentence with two identified entity mentions (a subject and an object), typed among 23 possible types (the types used by the Stanford NER system [12]), and annotated with a relation type among 41 effective classes and a *no_relation* class denoting an absence of relation between the two mentions. In order to reflect what can be found in real-world texts, 79.5% of the examples are in the *no_relation* class.

Several remarks can be made about this dataset. First, as the classification with Concepts of Neighbours is a lazy learning method, there is no validation step, then the development dataset can be merged with the training dataset to form a bigger training dataset. Second, the negative examples (those in the *no_relation* class) have no reasons to look like each other but can look like examples in other classes, because they express random situations. Therefore, as our method looks for similarities between a test example and training examples, those negative examples cannot easily be handled in our method. That is why negative examples are removed from the dataset in the experiments, and we focus on discriminating between the 41 relation types rather than discriminating between the presence and absence of a relation. This shrinks the training corpus to 18,446 examples and the test corpus to 3,325 examples.

The evaluation of an approach on TACRED is usually made using the micro-averaged F1 score. However, as the negative examples have been removed from the dataset, this score is equivalent to accuracy. Therefore, accuracy is the measure we use in this evaluation.

The fact that experiments are made on a subset of TACRED causes that direct comparison with existing approaches is no longer possible. In order to evaluate whether our modeling and use of concepts of neighbours are beneficial, we introduce a simple baseline based on named entity types. It predicts the most frequent relation type among the training sentences that have the same subject type and the same object type as the test sentence.

5.2 Experimental Settings

As the algorithm to compute the Concepts of Neighbours is anytime, a timeout has to be chosen. In order to see the influence of the computation time on the classification task, eight experiments have been run with respective timeouts: 10, 20, 30, 60, 120, 300, 600 and 1200 seconds. For each timeout we compare four configurations combining unpruned/pruned modelings (Section 3.2) and the two scoring methods (Section 4.2).

We have implemented our approach in Java[6], and we use library *ConceptualKNN*[7] for the computation of Concepts of Neighbours, which is based on

[6] Code available at `https://gitlab.inria.fr/hayats/conceptualknn-relex`

[7] `https://gitlab.inria.fr/hayats/jena-conceptsofneighbours`

Table 2: Accuracy of our approach depending on timeout, pruning, and scoring method, compared to the baseline.

	Timeout (seconds)							
Approach	**10**	**20**	**30**	**60**	**120**	**300**	**600**	**1200**
Baseline	80.4							
Unpruned, EV	78.6	79.1	79.2	79.4	79.2	79.2	79.6	79.6
Unpruned, MC	78.0	78.2	78.3	78.9	79.9	79.8	80.2	80.4
Pruned, EV	79.5	79.7	79.6	80.1	80.4	80.3	80.4	80.4
Pruned, MC	79.1	80.1	80.3	**81.3**	**82.1**	**82.5**	**82.6**	**82.5**

Apache Jena[8], a Java library for semantic web applications. Experiments have been run on Grid5000[9] to exploit parallel computation.

5.3 Quantitative Results

Accuracy of the baseline and of the four versions of the proposed method are presented in Table 2. For a timeout of at least 60s, the proposed approach with the Maximum Confidence scoring and the pruned modeling, has a better accuracy than the baseline, surpassing it by 1.7 point with a timeout of 120s and by 2.2 points with a timeout of over 600s. It can be observed that the pruning of the dependency trees in the modeling is necessary in order to beat the baseline. We assume that, without pruning, the search space for concepts of neighbours gets much larger, and so their computation cannot focus on the useful parts of the dependency trees in the allocated timeout. The EV scoring method shows negative results compared to Maximum Confidence and the baseline. Two conclusions can be made. First, the exponential decrease of vote weights seems to neglect too much distant concepts as Maximum Confidence does not penalize them. Second, it seems better to select a few high-confidence concepts than trying to aggregate predictions from all concepts.

Considering the last line of Table 2 (pruned and MC method), we observe a saturation phenomenon. Indeed, there is an important gain when timeout gets from 10s to 120s, but a far smaller gain from 120s to 1200s. Most of the concepts of neighbours are computed in less than 120 seconds, and over 120 seconds, only a few concepts are added. The same phenomenon is seen in Figure 5. The first chart clearly shows that most of the concepts are obtained in less than 120s. In addition, the second chart shows that, in the case of a model with pruned dependency trees, over 99% of the test sentences have a fully computed set of Concepts of Neighbours – and therefore the predictions are made on the Concepts of Neighbours themselves and not an approximation – for a timeout over 600s, while for a model with full dependency trees, only about 70% of the examples have fully computed set of Concepts of Neighbours.

[8] `https://jena.apache.org/`
[9] `https://www.grid5000.fr/w/Grid5000:Home`

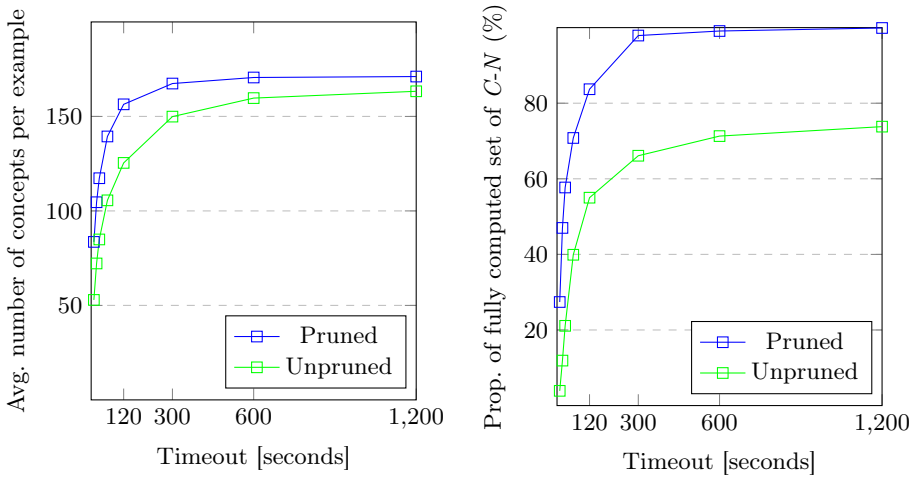

Fig. 5: Average number of concepts per example and proportion of fully computed sets of Concepts of Neighbours

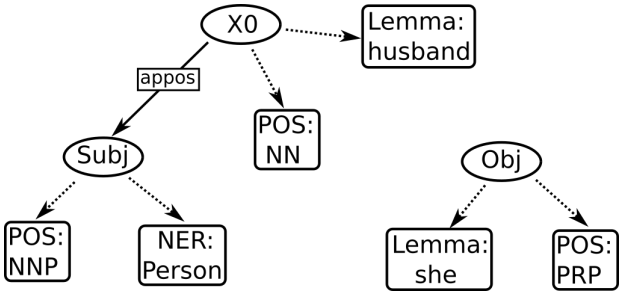

Fig. 6: Example of intension of a concept

5.4 Qualitative Results

An important asset of this relation extraction method is its interpretability: from a prediction, it can be retrieved which Concepts of Neighbours were used to do this prediction, and for each concept what graph pattern is expressed by the intension and which sentences of the training dataset match this concept.

For illustration, we examine the following example: *"**Her** husband, **Brad Hagemo**, is an optometrist and Scientologist"*. First, as the subject and the object (in bold) designate persons, there are six possible relations: *per:spouse*, *per:siblings*, *per:parents*, *per:children*, *per:other_family* and *per:alternate_names*. After computation of the Concepts of Neighbours, the relation *per:spouse* is predicted with Maximum Confidence. This prediction is based on the fact that three Concepts of Neighbours predict this relations with a confidence of 1, while at most two concepts predict any other relation with such a confidence. The graph

pattern used as intension for these concepts can be made explicit. For example, the intension of one of these concepts is shown in Figure 6. It expresses that the subject is a person designated as a husband, and the object is a personal pronoun of lemma *"she"*. This intension can effectively predict with quite a high confidence a marital relation. The training examples matching this intension are the sentences *"Kissel had [...] accused* ***her*** *husband, Merrill Lynch investment banker* ***Robert Kissel*** *of [...] domestic violence."* and *"Jane Callahan Gude, 84, [...] a tireless campaigner for* ***her*** *husband, former U.S. Rep.* ***Gilbert Gude****, died March 24 [...]."*

This example shows that, with this method, an interpretation can be extracted from a prediction. In addition, this shows that a prediction can be made with good confidence from concepts with a disjoint intension. In practice, we observed that concept intensions are rarely connected. A reason is that they may form patterns that are too specific to match any sentence of the training corpus.

6 Conclusion

We have presented a lazy-learning method for relation extraction, based on the modeling of linguistic data as a graph and on the computation of Concepts of Neighbours on this graph. This approach has been evaluated and validated against a baseline on the subset of positive examples of the TACRED benchmark. In addition, this comes with the advantage that this approach is interpretable as, for each prediction, detailed information about how this prediction has been made is given in the form of graph patterns over the linguistic structure.

Several aspects of this contribution lead to tracks for future work. First, this method could be coupled with another method – potentially Deep Learning – in order to be able to distinguish the positive examples from the negative ones. The Concepts of Neighbours method can also be improved in order to provide more flexible and expressive patterns, useful in the case of natural language processing. Finally, our approach could be adapted to other NLP tasks that require search for linguistic similarity.

References

1. Ferré, S., Cellier, P.: Graph-FCA: An extension of formal concept analysis to knowledge graphs. Discrete Applied Mathematics 273, 81–102 (2019)
2. Ferré, S.: Answers Partitioning and Lazy Joins for Efficient Query Relaxation and Application to Similarity Search. In: The Semantic Web. pp. 209–224. Cham (2018)
3. Ferré, S.: Application of concepts of neighbours to knowledge graph completion. Data Science (2020), `https://content.iospress.com/articles/data-science/ds200030`, to appear
4. Ferré, S., Ridoux, O.: The use of associative concepts in the incremental building of a logical context. In: U. Priss, D. Corbett, G.A. (ed.) Int. Conf. Conceptual Structures. pp. 299–313. LNCS 2393, Springer (2002)
5. Galárraga, L., Teflioudi, C., Hose, K., Suchanek, F.M.: Fast rule mining in ontological knowledge bases with AMIE+. The VLDB Journal 24(6), 707–730 (2015)

6. Ganter, B., Kuznetsov, S.: Pattern structures and their projections. In: Delugach, H.S., Stumme, G. (eds.) Int. Conf. Conceptual Structures. pp. 129–142. LNCS 2120, Springer (2001)
7. Ganter, B., Wille, R.: Formal Concept Analysis: Mathematical Foundations. Springer-Verlag (1999)
8. Grishman, R.: Twenty-five years of information extraction. Natural Language Engineering pp. 677–692 (2019)
9. Kuznetsov, S.O.: Machine learning and formal concept analysis. In: International Conference on Formal Concept Analysis. pp. 287–312. Springer (2004)
10. Kuznetsov, S.: Fitting pattern structures to knowledge discovery in big data. In: Cellier, P., Distel, F., Ganter, B. (eds.) Int. Conf. Formal Concept Analysis, pp. 254–266. LNAI 7880, Springer (2013)
11. Leeuwenberg, A., Buzmakov, A., Toussaint, Y., Napoli, A.: Exploring Pattern Structures of Syntactic Trees for Relation Extraction. In: Formal Concept Analysis. vol. 9113, pp. 153–168. Cham (2015), series Title: LNCS
12. Manning, C.D., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S.J., McClosky, D.: The Stanford CoreNLP Natural Language Processing Toolkit. Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations pp. 55–60 (2014)
13. Meilicke, C., Chekol, M.W., Ruffinelli, D., Stuckenschmidt, H.: Anytime Bottom-Up Rule Learning for Knowledge Graph Completion. In: Proc. of the Twenty-Eighth Int. Joint Conf. on Artificial Intelligence. pp. 3137–3143 (2019)
14. Miller, G.A.: WordNet: An Electronic Lexical Database. MIT Press, Cambridge, MA (1998)
15. Nguyen, T.H., Grishman, R.: Relation Extraction: Perspective from Convolutional Neural Networks. In: Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing. pp. 39–48 (2015)
16. Toutanova, K., Klein, D., Manning, C.D., Singer, Y.: Feature-rich part-of-speech tagging with a cyclic dependency network. In: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - NAACL '03. vol. 1, pp. 173–180 (2003)
17. Wu, F., Zhang, T.: Simplifying Graph Convolutional Networks. Proceedings of the 36th International Conference on Machine Learning p. 11 (2019)
18. Yamada, I., Asai, A., Shindo, H., Takeda, H., Matsumoto, Y.: LUKE: Deep Contextualized Entity Representations with Entity-aware Self-attention. In: Proc. of the Conf. on Empirical Methods in Natural Language Processing (EMNLP). pp. 6442–6454. ACL (2020)
19. Zhang, Y., Qi, P., Manning, C.D.: Graph Convolution over Pruned Dependency Trees Improves Relation Extraction. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing. pp. 2205–2215 (2018)
20. Zhang, Y., Zhong, V., Chen, D., Angeli, G., Manning, C.D.: Position-aware Attention and Supervised Data Improve Slot Filling. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing. pp. 35–45 (2017)