Learning Subgraph Patterns from text for Extracting Disease–Symptom Relationships

Mohsen Hassan, Adrien Coulet, Yannick Toussaint

Learning Subgraph Patterns from text for Extracting *Disease–Symptom* Relationships

Mohsen Hassan, Adrien Coulet, and Yannick Toussaint

LORIA (CNRS, Inria NGE, Université de Lorraine),
Campus scientifique, Vandoeuvre-lès-Nancy, F-54506, France
{mohsen.sayed,adrien.coulet,yannick.toussaint}@loria.fr

Abstract. To some extent, texts can be represented in the form of graphs, such as *dependency graphs* in which nodes represent words and edges represent grammatical dependencies between words. Graph representation of texts is an interesting alternative to string representation because it provides an additional level of abstraction over the syntax that is sometime easier to compute. In this paper, we study the use of graph mining methods on texts represented as dependency graphs, for extracting relationships between pairs of annotated entities. We propose a three step approach that includes *(1)* the transformation of texts in a collection of dependency graphs; *(2)* the selection of frequent subgraphs, named hereafter *patterns*, on the basis of positive sentences; and *(3)* the extraction of relationships by searching for occurrences of patterns in novel sentences. Our method has been experimented by extracting *disease–symptom* relationships from a corpus of 51,292 PubMed abstracts (428,491 sentences) related to 50 rare diseases. The extraction of correct *disease–symptom* relationships has been evaluated on 565 sentences, showing a precision of 0.91 and a recall of 0.49 (F-Meaure is 0.63). These preliminary experiments show the feasibility of extracting good quality relationships using frequent subgraph mining.

1 Introduction

In many domains such as biomedical research, text is a major source of information; unfortunately text corpora are frequently too large to be fully considered manually [1]. We focus here on the task of Relation Extraction (RE), which consists in identifying and qualifying valid relationships between entities already recognized in the text. Figure 1 illustrates the process of RE with an example of relation between a disease and a symptom. First, Named Entity Recognition (NER) identifies the interesting entities in the text and annotate them with the corrected category. Second step identifies if named entities are involved in a relationship (and may qualify the type of the relationship) or not.

Texts may be represented at different levels: words, bag of words, sequences of words, syntactic trees, graphs (dependency graphs); and they may be enriched by some linguistic features: part of speech, syntactic or semantic features. In this paper we study how text, represented in the form of graphs, can be processed with simple graph mining methods, to perform RE.

In: P. Cellier, T. Charnois, A. Hotho, S. Matwin, M.-F. Moens, Y. Toussaint (Eds.): Proceedings of DMNLP, Workshop at ECML/PKDD, Nancy, France, 2014.

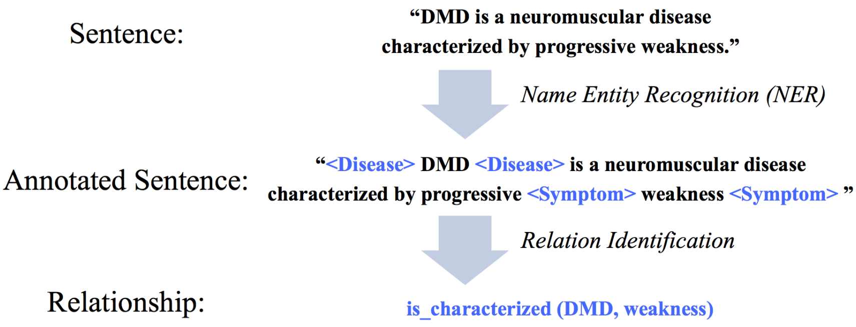

Fig. 1: The process of Relation Extraction (RE)

Frequent Subgraph Mining (FSM) is a graph mining method that extracts frequently occurring subgraphs either from a single graph or a set of graphs [2]. We propose in this paper to extract relationships from text through a three step method, based on FSM. The first step concerns data preparation and consists in transforming texts into graphs and recognizing name entities. The second step relies on the identification of labeled and oriented subgraphs, named hereafter *patterns*, that are connecting frequently two imposed typed entities, *e.g.*, subgraphs connecting one disease to one of its symptom. The third step uses generated patterns for extracting relationships between these entities.

The paper is organized as follows: Section 2 presents background elements regarding graph mining. Section 3 introduces our three step method. Section 4 reports experiments of our method on the extraction of *disease–symptom* relationships. Section 5 presents related works and Section 6 discusses the interest of using graph mining for RE.

2 Graph Mining

A graph is defined as a pair $G = (V, E)$ where V is a set of vertices (or nodes) and E is a set of edges connecting vertices such as $E \subseteq V \times V$. A graph is a *directed graph* when edges are oriented pairs of vertices. A graph is a *labeled graph* when vertices and edges are associated with labels.

2.1 Frequent Subgraph Mining

$S = (SV, SE)$ is a subgraph of G if $SV \subseteq V$ and $SE \subseteq E$. Given a graph collection $\mathcal{G} = \{G_1, G_2, ..., G_k\}$, with $G_i = (V_i, G_i)$, and a minimum support min_sup, the Frequent Subgraph Mining task (denoted FSM) extracts the collection of subgraphs $\mathcal{S} = \{S_1, ..., S_n\}$, with $S_i = (SV_i, SG_i)$ that occur in $\mathcal{G}$ with a support greater than min_sup. The support of a subgraph S_i is the number of its occurrences in $\mathcal{G}$[1].

[1] The relative support of S_i is $\frac{|S_i|}{|\mathcal{G}|}$

FSM algorithms are mainly based on two distinct approaches: *Apriori*-based and *pattern growth*-based approaches. *Apriori*-based graph mining algorithms share similarities with Apriori-based frequent itemset mining algorithms [3]. In their case, the search for frequent subgraphs starts with graphs with no edge. At each iteration, the size of the newly discovered frequent substructures is increased by one by joining two subgraphs from the previous iteration. AGM, FSG and FFSM are examples of Apriori-based algorithms [2,4,5]. The *pattern-growth* mining algorithms extend a frequent graph by trying to add successively a new edge to every possible position. If the new graph is frequent, a new frequent graph can be expended; if it is not frequent a new edge is tried to be added. gSpan [6], CloseGraph [7]and Gaston [8] are examples of pattern-growth algorithms.

2.2 gSpan

gSpan is a FSM algorithm that processes undirected labeled graphs. Given a collection of such graphs, gSpan returns the set of frequent subgraphs and their support. To generate this result, gSpan generates a Tree Search Space (TSS) that is composed of all trees and subtrees that rely in the collection of graphs. gSpan represents each tree of the TSS using a specific encoding, named *minimum Depth-First Search (DFS) Code*. This code is unique for each tree because it is constructed following the unique DFS traversal that follows the lexicographic order of vertex labels.

gSpan follows a pattern-growth mining approach, *i.e.,* expends at each iteration a frequent graph with a new edge, trying every potential position. An issue with this approach is that the same graph can be discovered several times from different frequent graphs. gSpan avoids this problem by introducing a *right-most extension technique*, where edge extensions only takes place on a specific position determined by DFS Codes.

3 Relationship Extraction using Frequent Subgraph Mining

We propose an original method based on FSM to extract relationships from text. Figure 2 depicts an overview of this three step method. Each step is detailed in next subsections.

3.1 Data Preparation

This step aims at transforming a collection of texts into a collection of Dependency Graphs (DG). To achieve this, texts are submitted to the following tasks: Sentence Splitting, NER and Named Entity (NE) Substitution, Sentence Filtering, Dependency Parsing and lemmatization. First, texts are split into sentences. Then, NEs are recognized. We focused on relation between diseases and symptoms. Thus, we replaced each occurrence of these entities by the corresponding generic word “DISEASE” or “SYMPTOM”. Sentences are filtered to keep those

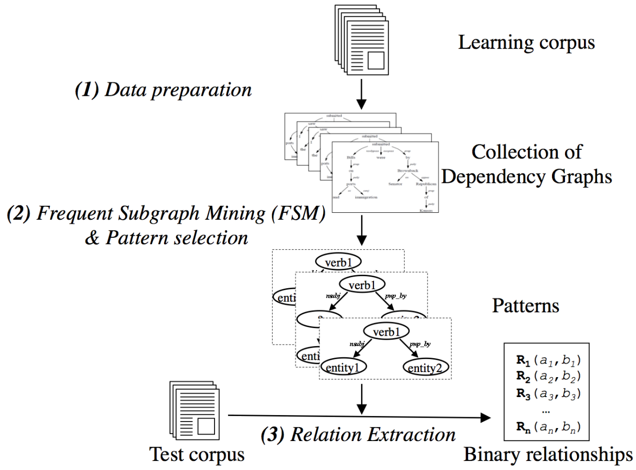

Fig. 2: Overview of our Relation Extraction (RE) method

involving at least two entities of interest. Dependency parsing produces for each sentence one labeled directed graph, named DG. Such DG is made of vertices that represent words and edges that are grammatical dependencies between words. Figure 3 shows the dependency graph of the sentence "*DMD is a neuromuscular disease characterized by progressive weakness.*".

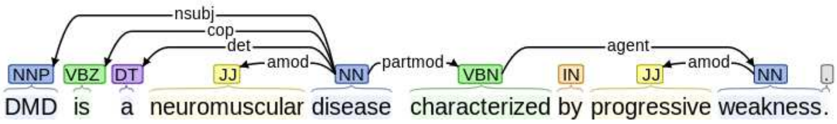

Fig. 3: Example of Dependency Graph (DG) processed by the Stanford Parser and drawn with Brat

Finally, words in DG are replaced by their lemmas, by a more general form that is more likely to appear in other graphs. Figure 4 shows an example of DG resulting from the data preparation step.

The collection of formatted DG is the input to FSM for mining the most frequent subgraph patterns that preserve the relations between two named entities.

3.2 Pattern Extraction

Frequent Subgraph Mining (FSM) aims at extracting useful patterns for the relation extraction process. Given a set of DGs and the support threshold, gSpan

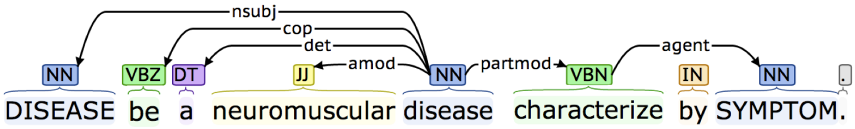

Fig. 4: Example of Dependency Graph after replacement of recognized entities (diseases and symptoms) by generic words (DISEASE and SYMPTOM) and lemmatization

extracts an undirected subgraph patterns. These patterns give the relationships between interesting annotated entities. Figure 5 shows an example of such pattern, extracted from graph in Figure 4. This subgraph pattern gives the relation based on grammatical dependencies between the disease "DMD" and the symptom "weakness".

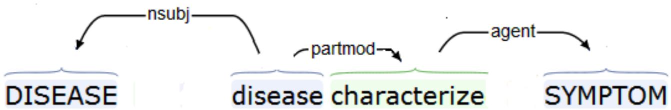

Fig. 5: Example of extracted pattern

Then, the patterns that contain the following are excluded: *(1)* conj_and or conj_or dependency relation between any two nodes; *(2)* The dependency path for DISEASE is equal to the dependency path for SYMPTOM, this means that DISEASE and SYMPTOM have the same semantic role in the sentence and this might be an error from NER; *(3)* no node for DISEASE or SYMPTOM (at least one disease and one symptom must be exist). Figure 6 shows an example of such excluded patterns. These patterns can be discovered from a sentence like "this disease is characterized by DISEASE and SYMPTOM"[2].

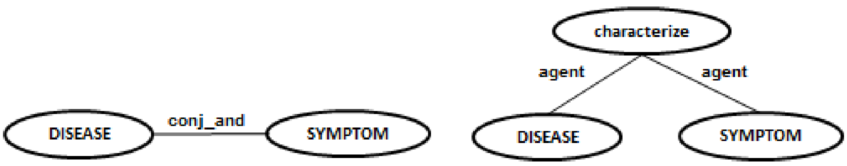

Fig. 6: Two examples of excluded patterns

[2] The uppercase words are the generic words for NEs

Bunescu and Mooney proposed a kernel method that used the shortest path between the two entities in the undirected version of the dependency graph [9]. We proposed similarly to compute the shortest path, but from directed dependency graph, which is useful for expressing the direction of relation between entities and consequently gives more precise relations. Two paths with the same sequence of syntactic dependency labels are similar if the direction of the syntactic dependencies are the same. Hence, the shortest path method (SPM) for extracting a smaller set of patterns than gSpan patterns has been used [9]. It consists in extracting the shortest path between two entities (*e.g.,* disease and symptom) in a dependency graph. Bunescu and Mooney used words and POS for expressing their pattern, but in SPM we consider the whole subgraph.

Figure 7 shows the shortest path between the two entities DISEASE and SYMPTOM in the dependency graph.

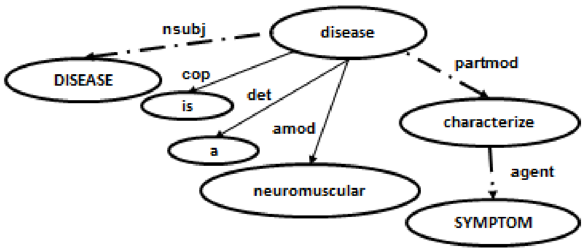

Fig. 7: The shortest path between DISEASE and SYMPTOM

Given the following two annotated sentences "DISEASE is a disease characterized by SYMPTOM" and "DISEASE is anomaly accompanied by SYMPTOM". First, SPM get the graph of each sentence as shown in figure 8. Then, SPM compute the common shortest path from the graphs of the two sentences. If the values of the nodes in the pattern are different, their values are replaced by "*" and keeping a list of all possible values for each node. Hence, two graphs patterns can be merged and represented in one generalized pattern [3]. The support of the new generalized pattern is less than or equal the sum of the supports of the two patterns. The support of the generalized pattern is automatically computed when executing the generalization operation which makes the process run faster.

Figure 9 shows two examples of pattern, one resulting from SPM patterns and the other resulting from gSpan. SPM method checks every node in the subgraph pattern to contain all possible values. This makes the pattern more general than gSpan pattern and increases the frequency value of the pattern. This has two advantages: first it produces a smaller set of patterns than gSpan patterns which is easier for analysis and evaluation purposes; second it leads to

[3] There is no redundancy because all redundant patterns are merged into one pattern

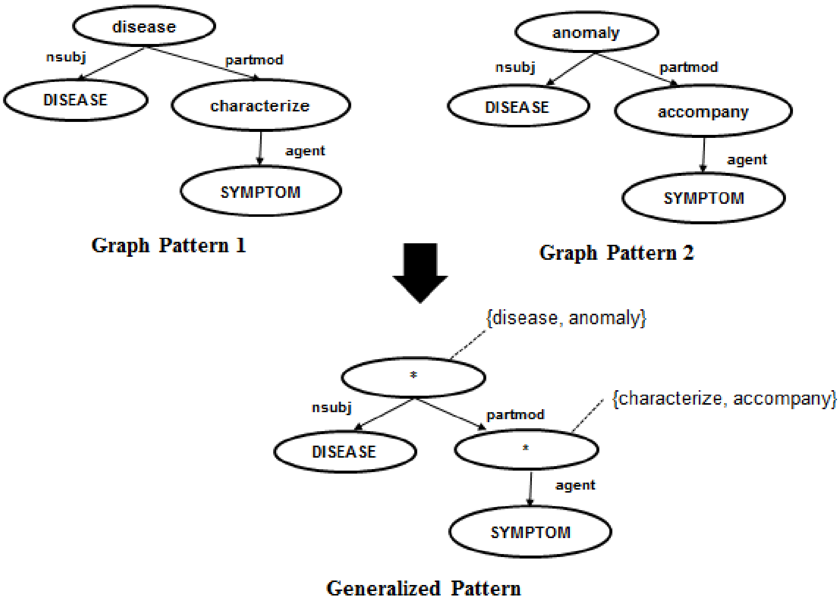

Fig. 8: Example of our method pattern

a higher coverage than gSpan when the pattern may not be extracted by gSpan because of its low frequency. On the other side, SPM did not use POS tags as a single feature as proposed in [9], what makes the pattern more generic and increases the coverage of patterns but induces a lower precision.

When SPM extends every node with all possible values, some values don't represent a correct relation between the annotated entities. Figure 10 shows rejected patterns that should be removed by the generalization operation to increase the quality of patterns.

The extracted patterns are classified into two classes: positive and negative patterns. The classification is based on pattern support and quality. The quality Q of a pattern is computed by the following formula

$$Q = \frac{T}{S} \tag{1}$$

where T is the number of all correct sentences in the pattern extension and S is the support of the pattern. A sentence is correct if it contains the pattern and the relation identified by the pattern is correct. For example, the pattern in Figure 8 has support 23. This means that the number of sentences that contain this pattern is 23. All disease-symptom relationships provided in these 23 sentences are correct. Then, T=23 and Q=23/23. Hence, the quality of this pattern is 1.

The pattern is a positive pattern if its support is higher than a minimum support (min_sup) threshold and its quality is higher than a minimum quality

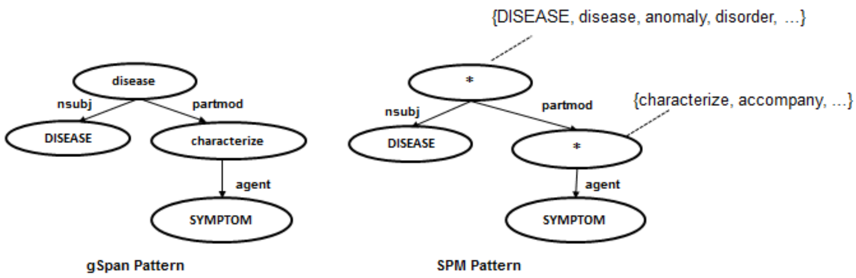

Fig. 9: Examples of patterns generated from gSpan and SPM

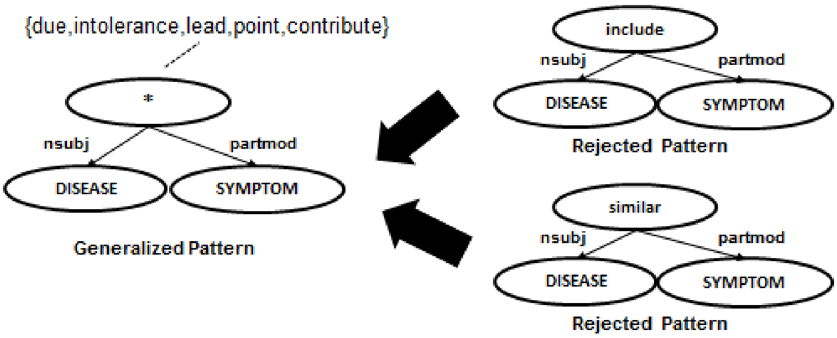

Fig. 10: Example of rejected patterns

threshold. Only positive patterns are considered for extracting new relationships from a new corpus.

3.3 Relationship Extraction using Patterns

Positive patterns previously selected are used to discover new relationships between entities mentioned in a new corpus. Similarly to the learning process, a set of dependency graphs are generated from the new corpus exactly as the data preparation step (sentences splitting and NER are also required before dependency parsing). Then, a pattern matching for the selected patterns with the dependency graphs is done to extract the binary relationships between the interesting entities. A value that expresses the quality of each new extracted relation is also returned (accordingly to the quality of the pattern used in the extraction process).

4 Experiment

We build-up experiments on the basis of a medical corpus related to rare diseases. This corpus is explored to extract relationships between diseases and symptoms. Figure 11 presents the process of our experiments and its evaluation. Details are provided in the following subsections.

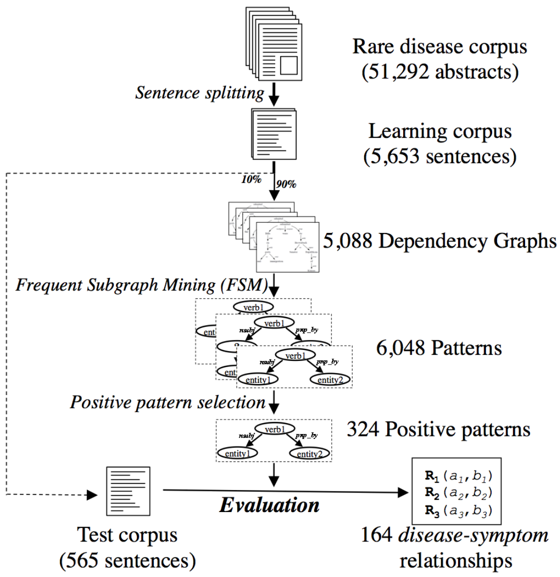

Fig. 11: Overview of our experiment for the extraction of disease-symptom relationships

4.1 Rare Disease Corpus

Our rare disease corpus is composed of 51,292 PubMed abstracts related to 50 Orphanet[4] rare diseases[5]. Abstracts are obtained by querying manually PubMed, using its web user interface. The query submitted to PubMed has the following form: "($disease_{1,pref\_name}$ `or` $disease_{1,syn_1}$ `or`...`or` $disease_{1,syn_n}$) `or`... `or` ($disease_{k,pref\_name}$ `or` $disease_{k,syn_1}$ `or`...`or` $disease_{k,syn_m}$)"

where $disease_{i,pref\_name}$ and $disease_{i,syn_j}$ are respectively referring to the preferred name and the j^{th} synonym of disease i according to the Orphanet Rare Disease Ontology[6].

[4] `http://www.orpha.net`
[5] The 50 diseases are listed at: `http://www.loria.fr/~msayed/50RareDiseases`
[6] `http://www.bioportal.bioontology.org/ontologies/ORDO`

4.2 Building a Dependency Graph Dataset

51,292 abstracts are split in 428,941 sentences using LingPipe[7], and subsequently submitted to disease and symptom NER. We use MetaMap to annotate each sentence of the corpus using UMLS semantic types "Disease or Syndrome" and "Sign or Symptom" [10]. MetaMap annotations of some very general words like "disorder" or "symptoms" have been removed to avoid noise in the rest of the process. Annotated sentences are divided into a *learning corpus*, made of 90% of sentences randomly selected, and a *test corpus*, made of the 10% left. In the learning corpus, each recognized disease or symptom is replaced by the generic string DISEASE or SYMPTOM (which are indexed when several disease or symptom are recognized in one sentence). Sentences that does not contain at least one disease and one symptom are filtered out, what reduces their number to 5,653.

The Stanford Parser is used to build the Dependency Graph (DG) of each sentence [11]. It is set to collapsed dependencies with propagation of conjunct dependencies option. As a result, conjunctions are propagating the dependencies that involves the conjuncts. For example, in the sentence "DISEASE is characterized by SYMPTOM1 and SYMPTOM2" this option guarantees that the same pattern will be observed between DISEASE and SYMPTOM1, and between DISEASE and SYMPTOM2. Finally, lemmatization is achieved using the Stanford CoreNLP suite.

4.3 Frequent Subgraph Mining

From prepared dependency graphs, gSpan extracts all frequent subgraphs. Those are filtered to keep only subgraphs that contain one disease and one symptom. This guarantees that only patterns that describe the dependency relation between disease and symptom are kept. We applied our program to gSpan subgraphs to identify in each case the shortest path between the nodes DISEASE and SYMPTOM. When several diseases or symptoms are in a unique sentence, one shortest path is computed for each pair (DISEASEx–SYMTOMy). This resulted in 6,048 subgraph patterns, a smaller set compared to gSpan result, consequently easier to evaluate. Because we think that shortest paths represent the most significance part of subgraph, we focused on these reduced graphs.

4.4 Selection of Positive Patterns

First, patterns with a $frequency \geq 2$ are selected from the 6,048. Accordingly, 615 patterns are frequent, covering 2,535 sentences from all 5,653 (44.84%). Second, patterns with a *pattern quality* ≥ 0.5, our quality threshold, are selected (see formula 1). It results 324 patterns (that cover 1,329 sentences or 23.51%), which are considered as *positive patterns* and are aiming at extracting new relationships from text.

[7] `http://alias-i.com/lingpipe`

4.5 Evaluation

Finally, we evaluate the ability of positive patterns to identify disease-symptom relationships in the test corpus. The evaluation process can be divided in three tasks. *(i)* For each sentence in the test corpus, two lists of disease-symptom pairs are composed: the list *i-a* of all possible disease-symptom pairs found in the sentence; the list *i-b* of pairs extracted by our method, *i.e.,* when a positive pattern matches the DG of a test sentence. Obviously, list *(i-b)* is a subset of *(i-a)*. *(ii)* Each pair of list *i-a* is marked manually as Correct if it corresponds to a relation actually mentioned in the sentence, or Incorrect, if it is not. *(iii)* Pairs that are marked as Correct and are extracted by our method (*i.e.,* in list *i-b*) are True Positive (TP); pairs that are marked as Incorrect and are not extracted by our method are True Negative (TN); pairs that are marked as Incorrect and are extracted by our method (*i.e.,* in list *i-b*) are False Positive (FP); pairs that are marked as Correct and are not extracted by our method are False Negative (FN).

Table 1: Confusion matrix corresponding to the evaluation of RE method on a corpus of 565 sentences. Because several relationships can be extracted from one sentence, TP+TN+FP+FN is higher than the number of sentences.

		Pattern-based extraction	
		Extracted	Not extracted
Manual extraction	Correct	TP=149	FN=172
	Incorrect	FP=15	TN=441

Table 1 shows the confusion matrix resulting form the evaluation process. It enables to compute the precision (P)[8] , recall (R)[9], F-measure [10], accuracy (Acc) [11] and specificity (S)[12]. Evaluation shows high precision (0.90) and specificity (0.96); reasonable F-measure (0.76) and accuracy (0.75); and a low recall (0.46).

5 Related Works

5.1 Mining Text as Set of words

Co-occurrence is the simplest method to identify relationships between two entities that co-occur in the same sentence. This approach is based on the hypothesis that if two entities are mentioned frequently together, it is likely that these

[8] P=TP/(TP+FP)
[9] R=TP/(TP+FN
[10] F-measure=2*P*R/(P+R)
[11] Acc=(TP+TN)/(P+N)
[12] S=TN/(FP+TN)

two entities are related. Co-occurrence methods have been successfully applied to the automated construction of networks of biomolecules such as protein-protein or gene-disease networks [12,13]. Co-occurrence approach tends to achieve a good recall but low precision. This can be balanced when one is mining very large corpus. Another issue with such approaches is that the type of relationships and their direction are unknown.

Bags of words are artificial constructs where one textual document is represented as an unordered set of the words it contains, *i.e.*, the *bag*. In this set, each word is usually associated with its frequency of occurrence in the document, then enabling to weight words within the bag. This is used to classify documents with similar words and words frequency profiles. Indeed, when associated with a proper dictionary, a document represented as a bag can be encoded as a simple vector of integers. This is a compact representation that enables to work with large corpora of documents. It suffers from low precision.

Sequence of words It consists of a partial order (*i.e.,* the sequence) of words, POS tags, general POS tags, entity or chunk type, etc. These features are used to build patterns or rules that assert a relationships between entities. Blohm *et al.* presented method based on a taxonomic sequential pattern for RE which extends a sequential mining algorithm to take into account a taxonomy of morpho-syntactic and lexico-semantic features [14]. It allows generalization or specialization among patterns, which affects the precision and the recall of the patterns. Quiniou *et al.* studied how to use the sequence mining to identify more generic linguistic patterns and show that sequence mining is more powerful than n-grams to express the linguistic patterns [15]. Béchet *et al.* provided a sequential pattern mining algorithm which discover the relations between genes and rare diseases in biomedical corpus [16]. The proposed algorithm extracts expressive linguistics patterns more efficient than patterns extracted with itemsets. Sequence mining tends to generate a very high number of patterns what makes difficult the analysis and evaluation tasks. Consequently filter are usually applied to reduce the number of extracted patterns.

5.2 Mining Trees

Parse Tree is an ordered, rooted tree that represents the syntactic structure of a sentence. Some works have been proposed to use such syntactic structure for extracting relations between entities. Galitsky introduced the operation of the syntactic generalization which take a pair of syntactic parse trees and find the maximal common subtrees [17]. Galitsky employed the nearest neighbour learning method to find the maximal common subtrees. Zhang *et al.* proposed a kernel approach that uses the syntactic tree representation of sentences for RE. They studied how to capture the syntactic structure by using a convolution tree kernel and support vector machines [18]. Zelenko *et al.* also proposed a tree kernel method, but using shallow parse tree representations [19]. The same tree kernel approach has been used by Culotta and Sorensen, but allowed feature weighting and used additional features such as Wordnet, POS, entity types [20]. In both approaches, a relation instance is defined by the smallest subtree in the parse or dependency tree that includes interesting entities. The tree kernel approaches achieve good results but they are hard to implement and computationally complex. Note that trees are specific type of graphs and mining trees can be easily adapted to graphs.

5.3 Mining Graphs

Many RE methods based on DG have been proposed [21,22]. Chowdhury et Lavelli proposed a hybrid kernel approach, which is based on different features: dependency patterns, regex patterns, path enclosed and shallow linguistic kernels [23]. In this case, dependency patterns are reduced graphs, which are subgraphs from dependency graphs. The reduced graph extends the shortest path (smallest common subgraph) of the dependency by adding *(a)* dependent nodes (when exist) of nodes in the shortest path; *(b)* the immediate governor(s) (when exist) of the least common governor. For sake of simplicity, we choose in this paper to consider only the shortest path with no dependents. Bunescu *et al.* proposed a RE approach similarly based on the shortest path between two entities in undirected dependency graphs [9].

Chowdhury et Lavelli also proposed to use a reduced graph pattern that is a set of syntactic dependences of the corresponding reduced graph. For example, the reduced graph pattern of the graph represented Figure 7 is $\langle nsuj, cop, det, jj, partmod, agent\rangle$. Note that, in this case, reduced graph patterns are undirected.

Adolphs *et al.* developed a rule learning algorithm to learn graph rules which identify subgraphs in arbitrary graphs [24]. First, subgraphs are extracted. Then, subgraph generalization is done to form rules by underspecifying the nodes and introducing place-holders labeled with the role for the argument nodes.

6 Discussion and Conclusion

6.1 Discussion

The proposed hybrid kernel approach of Chowdhury et Lavelli [23] is evaluated on 5 different corpora for the extraction of the protein-protein relationship and the results varied from corpus to another. Considering pattern features and corpora used, our method shows a good precision (0.91) and low recall (0.49). This illustrates that graph mining can produce precise patterns for RE but additional work is required, such as adding features (*e.g.*, similar to those proposed in Chowdhury's work). Béchet *et al.* [16] use sequential mining patterns for extracting gene-disease relationships. The method gives the best precision 0.68 (recall 0.36) when using min_sup = 50 while the best recall is 0.65 (precision is 0.66) when using min_sup = 5. While in our method we achieve the best precision

0.94 (recall 0.33) and the best recall 0.67 (precision 0.41). In addition, the huge number of patterns produced by sequence mining; makes the interpretation task hard.

In our experiments, we fixed min_sup=2. When min_sup=2, the number of extracted patterns is 615. When increasing the min_sup threshold, the number of extracted patterns and recall decrease. For example, if min_sup=3, then the number of extracted patterns is 268. When decreasing the min_sup threshold, the number of extracted patterns and recall increase. For example, If min_sup=1, then the number of extracted patterns is 6048. This number of patterns is large for analysis and patterns with support=1 may be not important because they are rare patterns.

Figure 12 shows the relation between the precision and pattern quality threshold and between recall and pattern quality threshold. Precision increases and recall decreases when the pattern quality threshold increases. The best precision value is 0.94 when the quality threshold is 100 and the best recall value is 0.67 when the quality threshold is 0. The trade-off between the precision and recall is required according to the purpose of the application.

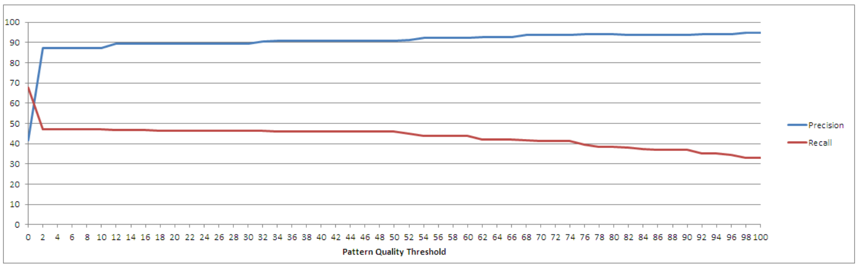

Fig. 12: The relation between precision and recall with the quality threshold

The study of FN and FP relations is necessary for improving the recall and precision respectively. In the following sentence “In areas in which transmission is occurring, WNV infection should be considered in patients with acute flaccid paralysis.”, the relation between disease “WNV” and symptom “flaccid paralysis” is marked as FN relation (because we didn’t generate a positive pattern that describes this relation). A possible solution for this problem is to consider patterns with low frequency (rare patterns), another solution is to enlarge the learning corpus. These produce a larger patterns set which reduces FN relations and increases the recall.

On the other side, to increase the precision, the number of FP relations needs to be reduced. The following sentence “Muscular dystrophy is a nosology for a group of hereditary muscle disorders characterized by progressive wasting and weakness of skeletal muscle, where degeneration of muscle fibers is detected by pathological examination” generates a FP relation between “hereditary muscle” and “weakness”. One solution is to consider only patterns with high quality by increasing the quality threshold to ensure that the extracted patterns are precise enough.

Finally, Unlike gSpan and Chowdhury’s work, SPM doesn’t able to keep other features such as negation relation. issues like this must be token in consideration for further improvements and extensions to SPM.

6.2 Conclusion

This paper illustrates how graph mining can be used for RE. We propose a simple method based on FSM to extract relationship from a corpus of text represented as DGs. FSM enables to identify subgraph patterns that are filtered based on their frequency and quality. Selected patterns are in turn used to extract relationships form novel sentences. Our evaluation on a corpus related to rare diseases showed a very high precision of 0.91.

In the future, the recall of the FSM-based method may be enhanced by improving its ability to identify FN relations. Also, thanks of the readability of the extracted patterns, studying and adding new features or constraints to improve the quality of these patterns is possible and may increase the recall and precision values. Combining features of sequences, syntax trees and dependency graphs may introduce more precise patterns with higher recall.

References

1. Larsen, P.O., von Ins, M.: The rate of growth in scientific publication and the decline in coverage provided by science citation index. Scientometrics (2010) 575–603
2. Kuramochi, M., Karypis, G.: Frequent subgraph discovery. In: Proceedings of the 2001 IEEE International Conference on Data Mining. ICDM ’01, Washington, DC, USA, IEEE Computer Society (2001) 313–320
3. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules in large databases. In: Proceedings of the 20th International Conference on Very Large Data Bases. VLDB ’94, San Francisco, CA, USA, Morgan Kaufmann Publishers Inc. (1994) 487–499
4. Inokuchi, A., Washio, T., Motoda, H.: An apriori-based algorithm for mining frequent substructures from graph data. In: Proceedings of the 4th European Conference on Principles of Data Mining and Knowledge Discovery. PKDD ’00, London, UK, UK, Springer-Verlag (2000) 13–23
5. Huan, J., Wang, W., Prins, J.: Efficient mining of frequent subgraphs in the presence of isomorphism. In: Proceedings of the Third IEEE International Conference on Data Mining. ICDM ’03, Washington, DC, USA, IEEE Computer Society (2003) 549–
6. Yan, X., Han, J.: gspan: Graph-based substructure pattern mining. In: Proceedings of the 2002 IEEE International Conference on Data Mining. ICDM ’02, Washington, DC, USA, IEEE Computer Society (2002) 721–
7. Yan, X., Han, J.: Closegraph: Mining closed frequent graph patterns. In: Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. KDD ’03, New York, NY, USA, ACM (2003) 286–295

8. Nijssen, S., Kok, J.N.: The gaston tool for frequent subgraph mining. Electr. Notes Theor. Comput. Sci. **127**(1) (2005) 77–87
9. Bunescu, R.C., Mooney, R.J.: A shortest path dependency kernel for relation extraction. In: Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing. HLT '05, Stroudsburg, PA, USA, Association for Computational Linguistics (2005) 724–731
10. Aronson, A.R.: Effective mapping of biomedical text to the umls metathesaurus: the metamap program. Proc AMIA Symp (2001) 17–21
11. de Marneffe, M.C., Manning, C.D.: The stanford typed dependencies representation. In: Coling 2008: Proceedings of the Workshop on Cross-Framework and Cross-Domain Parser Evaluation. CrossParser '08, Stroudsburg, PA, USA, Association for Computational Linguistics (2008) 1–8
12. Šarić, J., Jensen, L.J., Ouzounova, R., Rojas, I., Bork, P.: Extraction of regulatory gene/protein networks from medline. Bioinformatics **22**(6) (March 2006) 645–650
13. Friedman, C., Kra, P., Yu, H., Krauthammer, M., Rzhetsky, A.: Genies: a natural-language processing system for the extraction of molecular pathways from journal articles. Comput. Appl. Biosci. **17**(suppl_1) (June 2001) S74–82
14. Blohm, S., Buza, K., Cimiano, P., Schmidt-Thieme, L. Applied Semantic Web Technologies. In: Relation Extraction for the Semantic Web with Taxonomic Sequential Patterns. Taylor and Francis Group (2011) 185–209
15. Quiniou, S., Cellier, P., Charnois, T., Legallois, D.: What about sequential data mining techniques to identify linguistic patterns for stylistics? In Gelbukh, A.F., ed.: CICLing (1). Volume 7181 of Lecture Notes in Computer Science., Springer (2012) 166–177
16. Béchet, N., Cellier, P., Charnois, T., Crémilleux, B., Jaulent, M.C.: Sequential pattern mining to discover relations between genes and rare diseases. In Soda, P., Tortorella, F., Antani, S., Pechenizkiy, M., Cannataro, M., Tsymbai, A., eds.: CBMS, IEEE (2012) 1–6
17. Galitsky, B.: Machine learning of syntactic parse trees for search and classification of text. Engineering Applications of Artificial Intelligence **26**(3) (2013) 1072 – 1091
18. Zhang, M., Zhou, G., Aw, A.: Exploring syntactic structured features over parse trees for relation extraction using kernel methods. Inf. Process. Manage. **44**(2) (March 2008) 687–701
19. Zelenko, D., Aone, C., Richardella, A.: Kernel methods for relation extraction. J. Mach. Learn. Res. **3** (March 2003) 1083–1106
20. Culotta, A., Sorensen, J.: Dependency tree kernels for relation extraction. In: Proceedings of the 42Nd Annual Meeting on Association for Computational Linguistics. ACL '04, Stroudsburg, PA, USA, Association for Computational Linguistics (2004)
21. Fundel, K., Küffner, R., Zimmer, R.: Relex—relation extraction using dependency parse trees. Bioinformatics **23**(3) (January 2007) 365–371
22. Coulet, A., Shah, N.H., Garten, Y., Musen, M.A., Altman, R.B.: Using text to build semantic networks for pharmacogenomics. Journal of Biomedical Informatics **43**(6) (2010) 1009–1019
23. Chowdhury, M.F.M., Lavelli, A.: Combining tree structures, flat features and patterns for biomedical relation extraction. In: EACL. (2012) 420–429
24. Adolphs, P., Xu, F., Li, H., Uszkoreit, H.: Dependency graphs as a generic interface between parsers and relation extraction rule learning. In: Proceedings of the 34th Annual German Conference on Advances in Artificial Intelligence. KI'11, Berlin, Heidelberg, Springer-Verlag (2011) 50–62