Graph-Based Representations for Computational Linguistics Tasks

Nahun Loya[1], Iván Olmos[1], David Pinto[1], and Jesús González[2]

[1] Benemérita Universidad Autonóma de Puebla
Faculty of Computer Science
Av. San Claudio y 14 Sur, C.P. 72570, Puebla Pue., Mexico
{nahun.loya,iolmos,dpinto}@cs.buap.mx
[2] National Institute of Astrophysics, Optics and electronics
Department of Computer Science
Luis Enrique Erro No. 1, C.P. 72840, Tonantzintla, Puebla, Mexico
jagonzalez@inaoep.mx

Abstract. In this work is introduced a methodology for discover hidden patterns in texts based on a graph-based representation. This methodology is divided on three main steps: first, a pre-processing phase, where texts are cleaning of unuseful data; second, we propose three different graph-based representations, where texts can be mapped to labeled graphs; third, the data mining phase, which can be performed with the SUBDUE system. Also, we show how could be interpreted the outputs patterns. The advantage of our methodology in opposite with others techniques is that allow extract structured patterns from texts.

Key words: Graph-based representations, PLN, authorship attribution, author profiling identification.

1 Introduction

Nowadays, there is a huge amount of data based on texts. Therefore users need modern tools capable to analyze and extract useful information over data, because is not possible a manual analysis since the huge volume of data. Text mining is a new burgeoning research field that attempts to glean meaningful information from natural language texts, with the aim to extract useful information (knowledge) [5].

A key factor to have success in the extraction knowledge process is the text representation. In the Natural Language Processing (NLP) has been proposed different text representations, such as bags of words [3], analysis with "n-grams" (detection of contiguous sequeces of words) [4]. More sofisticated representations include the use of ontologies, which represents knowledge as a set of concepts within a domain, and the relationships between those concepts [2], or techniques based on the Latent Semantic Indexing, where queries and documents are represented into a space with latent semantic dimensions [1].

Based on the above mentioned, there are different ways to represent texts, each of one with advantages or disadvantages (in terms of space complexity, time

complexity, or capability of preserve patterns that exist in the original texts). In this sense, this work introduce the use of labeled graphs as text representation, because is possible to include many linguistics levels of the original text such as, lexical, morphological, syntactical, etc. Furthermore, graphs let us express concepts and relations using their nodes and their edges.

The main contribution of this work is the introduction of a methodology that define how input texts can be represented with labeled graphs, with the aim to discover common patterns (common graphs) in a data mining task, taking advantage of the flexibility of the graphs to represent complex structures.

The rest of the paper is structured as follows: Section 2 introduced some notation useful to represent labeled graphs. Section 3 presents the proposed methodology for discovering patterns in text based on a graph representation. Section 4 introduce general concepts with the aim to understand the graph-based data mining phase. Finally, in section 5 is presented the conclusions.

2 Graph definitons and basic concepts

Several researchers have been proposed different graph notations. The most widely used graph notation for graphs is $G = (V, E)$, where V is a set (not empty) of vertices, and E is a set of edges, $E \subseteq V \times V$. However this notation is not suitable for text mining, because data information is represented through labels that are attached to the vertices and edges. Based on the above mentioned, is introduced a graph notation for labeled/unlabeled graphs.

Definition 1 *A labeled graph G is a 6-tuple $G = (V, E, L_V, L_E, \alpha, \beta)$, where:*

- $V = \{v_i | i = 1, ..., n\}$ is a finite set of nodes, $V \neq \emptyset$, and $n = \#vertices$ in G.
- $E \subseteq V \times V$ is a finite set of edges, $E = \{e = \{v_i, v_j\} | v_i, v_j \in V, 1 \leq i, j \leq n\}$.
- L_V, is a set of nodes labels.
- L_E, is a set of edges labels.
- $\alpha : V \rightarrow L_V$, is a function that assigning labels to the nodes.
- $\beta : E \rightarrow L_E$, is a function that assigning labels to the edges.

For unlabeled graphs and without loss of generality, it is possible to assign a same label for all vertices and all edges. Based on this notation, it is possible to introduce different graph topologies useful for our work.

Definition 2 *A chain topology is defined as the set of all graphs $G = \{V, E, L_V, L_E, \alpha, \beta\}$, where: $V = \{v_j | j = 1, \ldots, n\}$; $E = \{e = \{v_j, v_{j+1}\}: 1 \leq j \leq n-1\}$; L_V = Set of label of vertices; L_E = Set of label of edges; $\alpha : V \rightarrow L_V$.; $\beta : E \rightarrow L_E$..*

Definition 3 *The ring topology is the set of all graphs $G = \{V, E, L_V, L_E, \alpha, \beta\}$, where: $V = \{v_j | j = 1, \ldots, n\}$; $E = \{e = \{v_j, v_{j+1}\}: j = 1 \ldots n-1\} \bigcup \{e = \{v_n, v_1\}\}$; L_V = Set of label of vertices; L_E = Set of label of edges; $\alpha : V \rightarrow L_V$,; β: $E \rightarrow L_E$.*

Definition 4 *The star topology is the set of all graphs* $G = \{V, E, L_V, L_E, \alpha, \beta\}$, *where:* $V_1 = \{v_0\}$; $V_2 = \{v_j | j = 1, \ldots, n\}$; $V = V_1 \bigcup V_2$; $E = \{e = \{v_0, v_j\}: 1 \leq j \leq n\}$; L_V = *Set of label of vertices;* L_E = *Set of label of edges;* $\alpha = \{\alpha(v) = etq, etq \in L_V, v \in V_2\} \bigcup \{\alpha(v_0) = inicial, v_0 \in V_1\}$; $\beta = \{\beta(e) = etq, etq \in L_E \ y \ e \in E\}$

For example, the sentence *"Abraham Lincoln fue presidente de EU"* can be mapped to a graph with a star topology as is shown in figure 1, which is defined as follows: $G = \{V, E, L_V, L_E, \alpha, \beta\}$, where: $V = \{v_{ini}, v_j | \ j = 1, \ldots, 6\}$; $E = \{e = \{v_{ini}, v_j\}: 1 \leq j \leq 6\}$; $L_V = \{Abraham, Lincoln, fue, presidente, de, EU\}$; $L_E = \{next\}$; $\alpha = \{\alpha(v_1) = Abraham, \alpha(v_2) = Lincoln, \alpha(v_3) = fue, \alpha(v_4) = presidente, \alpha(v_5) = de, \alpha(v_6) = EU\}$; $\beta = \{\beta(e_1) = next, \ldots, \beta(e_6) = next\}$

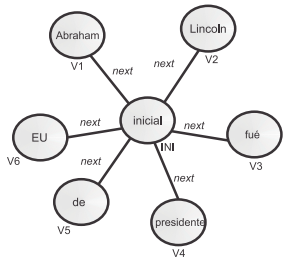

Fig. 1. Example of a sentence mapped to a graph with a star topology.

Important graph topologies were introduced in this section: chain, ring, and star. However, there exist more topologies (see figure 2), some of one based on the above described, such as: trees (derived from stars), backbone, backbone-tree, star ring, etc. Our work is based on trees and stars topologies, but it is possible to propose new representations with others topologies.

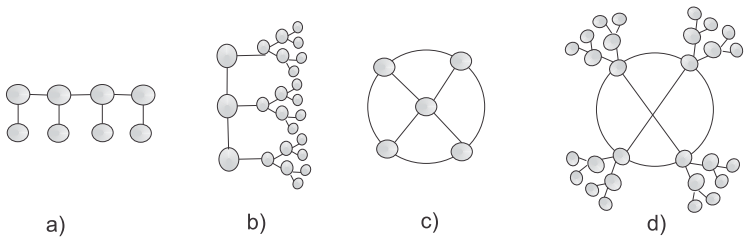

Fig. 2. Different types of topologies: a)Backbone, b)Backbone-tree, c)Star ring, d)Star ring with trees.

3 Methodology for text representation using labeled graphs

The main contribution of our work is the introduction of a methodology where it is possible to reduce texts to labeled graphs (transformation process). The metodology is presented in this section, which consist of three main steps, as is shown in figure 3:

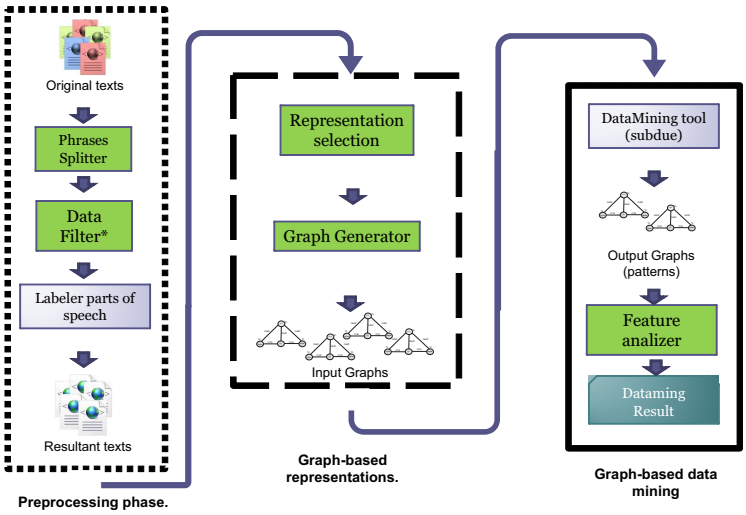

Fig. 3. Steps of the Proposed methodology.

3.1 Preprocessing phase

The preprocessing phase implements three steps, where the input texts are splitting, cleaning, and tagged. Consider that the input text has a plain format. Then, the text is divided in sentences, delimited by the puntuation mark ".". Clearly, there are some cases where this rule should not be applied. For example, if appear suspension points ("Dudé..", in Spanish), and abbreviations ("Dr. Juan Peréz", in Spanish). Consequently, it is necessary to detect " tokens", where words with a single mark "." at the end are considered as tokens. Next, stopwords are removed, because they not apport relevant information about the semantic of the text. Then, a morphological tagger is used[3] with the aim to detect features of the words, such as lemmas, dependency relations between words, and others. Thus, in this step every word in the text is labeled, depending of the roll that each

[3] In this work is used the Freeling tagger, www.freeling.org

word take into the sentence. As result of this, in the next step (process where texts are mapped to graphs) is possible to use different type of information of the input text, such as the original words of the text, lemmas, parts of the speach (grammatical categories of the words), sequence relations between words (the "next" relation), and type dependency relation between words.

3.2 Graph representations for texts

In this section, three graph-based representations are proposed, which are useful to represent different levels of information that the original texts have. As we mentioned before, the Freeling tool is used to generate the gramatical categories of each word. For the sake of simplicity, the set of all grammatical categories included in the input text is represented with POS, the set of all lemmas is represented with PALLEM, and the words of the input text is denoted with WORDS.

Sequential three representation (STR). This representation is focused on modeling the structure of a sentence. This representation is called "Sequential three representation (STR), based on a hybrid star topology. In STR the words are mapped to vertices, where vertices are labeled with the lemma of each word. Also, two types of relations are established between vertices: first, all vertices that represent neighborhood words (in sequence) are linked with an edge labeled with the word "next". Also, these vertices are linked to a vertex called "enun" with an edge labeled with the word "POS", as is shown in figure 4. This topology is focused to detect common structures (common graphs) between two or more sentences, including secuencial or not secuencial words.

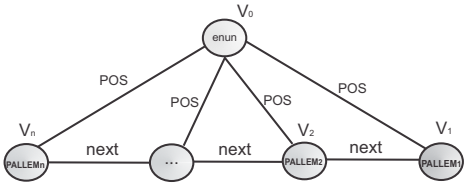

Fig. 4. Representation (STR).

Formally the representation is expressed as follows. Consider a sentence $S =< word_1, word_2, \ldots, word_n >$, where $word_i$ represents a word in the input sentence (that previously has been preprocessed). Consider that $POS(word_i)$ represents the grammatical class of the word $word_i$. A graph $G = \{V, E, L_V, L_E, \alpha, \beta\}$ is a STR representation of the sentence $S =< word_1, word_2, \ldots, word_n >$, if:

- $V = \{v_0\} \cup V_{PALLEM}$, where $V_{PALLEM} = \{v_1, \ldots, v_n\}$
- $E = \{\{v_0, v_i\}$, where $v_i \in V_{PALLEM}$, $i = 1, \ldots, n\} \cup \{\{v_i, v_{i+1}\}$, where $v_i \in V_{PALLEM}, i = 1, \ldots, n-1\}$

- $L_V = \{enun\} \cup L_{PALLEM}$, where $L_{PALLEM} = \{PALLEM(word_1) \ldots, PALLEM(word_n)\}$
- $L_E = \{next\} \cup L_{POS}$, where $L_{POS} = \{POS(word_1) \ldots, POS(word_n)\}$
- $\alpha : V \rightarrow L_V$ where $\alpha(v_0) = enun$, $\alpha(v_i) = PALLEM(word_i)$ if $v_i \in V_{PALLEM}$
- $\beta :\rightarrow L_E$, where $\beta(\{v_x, v_{X+1}\}) = next$ if $v_x \in V_{PALLEM}$ and $x = 1, \ldots, n-1$, either $\beta(\{v_x, v_y\}) = POS(word_i)$ if $v_x = v_0$ and $v_y \in V_{PALLEM}$

For example, consider the following two sentences: S1="Abraham Lincoln fue presidente de Estados Unidos"; S2="Estados Unidos tuvo un presidente llamado Abraham Lincoln". After the preprocessing phase, the original sentences are lemmatized: S1'="Abraham_Lincoln ser presidente Estados_Unidos"; S2'="Estados_Unidos tener presidente llamar Abraham_Lincoln". Using Freeling, it is possible to calculate the set of grammatical categories (POS). The results of this phase are: S1"="NP VS NC NP"; S2"="NP VM NC VM NP". Finally, the STR representation of S1" and S2" is shown in figure 5.

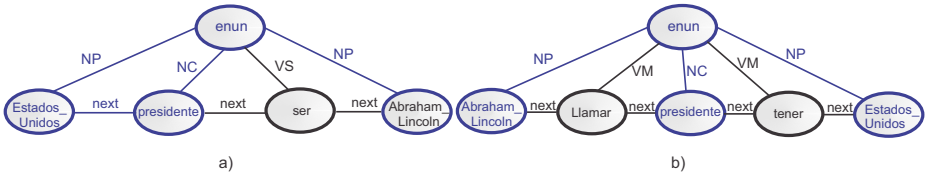

Fig. 5. Example of STR representation.

Based on these graph representations of the original sentences, with a GBDM tool such as SUBDUE, it is possible to discover common structures between these graphs (represented with no-black lines). Note that the common patterns includes words that are not in a strict order. Due to the structure of STR, the graph-based data mining process is capable to discover such patterns. As consequence, this representation can be used to discover common structures between sentences, including: common lemmas (of the words), common lemmas that appear in sequence, and common structure of the sentences that are similar in the input sentences.

Representation based on word sequence (RBWS). The next representation is called "Representation based on word sequence (RBWS)". The aim of this representation is to detect the relationship between words in each sentence. For each different word in the input sentence, is created a node in the graph, labeled with its lemma. Then, and edge between two nodes of the graph is created with the label "next", if and only if the words associated with these vertices appear in sequence in the original sentence. Therefore, this representation is not based on a specific graph topology, instead it is defined by the structure of the sentence. As consequence, it is possible to discover n-grams, but unlike the original n-gram

model, in RBWS is not defined a particular value for n. This representation is shown in figure 6.

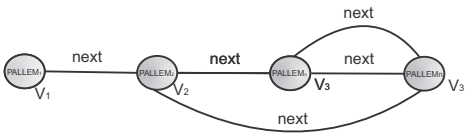

Fig. 6. Representation (RBWS).

For example, consider the following two sentences: S1="Abraham Lincoln fue presidente de Estados Unidos. El presidente dio los fundamentos de la libertad de Estados Unidos"; S2="Los fundamentos de la libertad fueron dados por el presidente de los Estados Unidos". After the lemmatization process of the preprocessing phase, the result sentences are: S1'="Abraham_Lincoln ser presidente Estados_Unidos presidente dar fundamento libertad Estados_Unidos"; S2'="Fundamento libertad ser dar presidente Estados_Unidos". The RBWS representation for these two sentences is shown in figure 7.

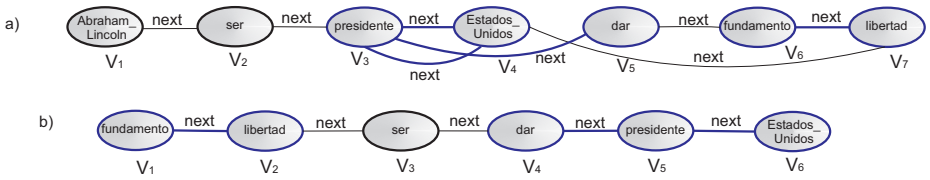

Fig. 7. Example of RBWS representation.

In this example, it is possible to see that the common graph represents words that are not only in sequence, but also in not sequence. Moreover, since this repesentation have not a specific topology, then it is possible to discover complex patterns, which represents words that have common neighborhood words.

Representation based on dependence type (RDT). In the same way of the previous representation, RDT is a representation without a specific topology. The aim of this representation consist of the construction of a graph based on the analysis of type dependencies. The graph is defined with vertices, where their labels include the lemma of the word (PALLEM), and the grammatical category, denoted by POS. For example, consider the spanish word "comió", its corresponding vertex is labeled with "comer_vb", where "comer" is the lemma and "vb" represents the grammatical category. Finally, the dependency types define the edges of the graph. Clearly, the labels of the vertices are more restrictive at the moment of the mapping process, but is an interesting differentiation

criterion for words with different grammatical categories. This representation is illustred in figure 8.

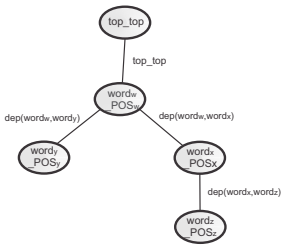

Fig. 8. Representation based on dependence type(RDT).

As an example, consider the sentences: S1="Abraham Lincoln fue presidente de los americanos"; S2="Michel Obama será presidenta americana". With the reprocessing phase, the original sentences are transformed to: S1'="Abraham_Lincoln fue presidente americanos", and S2'="Michel_Obama será presidenta americana". After that, the lemmas of each word are calculated: S1"="Abraham_Lincoln ser presidente americano", and S2"="Michel_Obama ser presidente americano". Additionally, all typed-dependency relations are calculated, as in shown in table 1.

Relation	Token 1	Token 2
top_top (Root Node)	top_top	ser_VS
subj (Topic of the sentence)	Abraham_Lincoln	ser_VS
att (Verb Attribute)	presidente_NC	ser_VS
adj_mod (Modifier adjective)	americano_AQ	presidente_NC

Table 1. Dependence typed relations of the example RDT.

Finally, based on the RDT representation, the graphs shown in figure 9 are generated. It is evident that this graph representation reduce the possibilities of matching in the data mining phase, and if we analyze a set of sentences with few examples, then the probability to discover interesting patterns is reduced. However, this limitation can be useful in sets with a high number of instances.

4 Graph-based data mining phase

Based on the transformation process described above, it is possible to generate a set of graphs from a set of input sentences (texts). Now, it is necessary to explain how it is possible to discover hidden patterns in graphs, called the Graph-Based

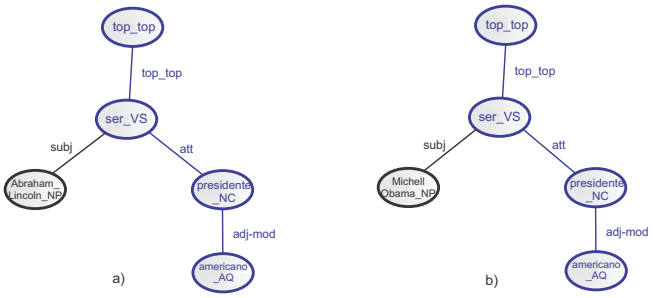

Fig. 9. Example of type dependencies (RDT).

Data Mining phase. One of the most important tools in the GBDM area is SUBDUE. In this section is introduced a brief description of this tool[4].

In a general way, this tool admit as input a set of labeled graphs, and as output report common substructures (subgraphs) of the input graphs. For the case of text mining, the output represents common patterns of the sentences, such as common words, characteristics and relations. In this sense, the output can be analyzed in two ways:

- The characteristics reported into the graphs. In this sense, it is possible to extract all words, lemmas, etc. that graphs reports. As consequence of this, interesting patterns can be found, such as: words or lemmas with a high number of instances, n-gramms (words that appear together), words or lemmas with a high probability to appear into the text, but not one beside the other, etc. This information can be used in the PLN area such as characteristics of the texts.
- The structure reported into the graphs. This part is one of the most important results of this approach, because not only is possible to discover common words, lemmas, etc., but also interesting relations between these elements, which represent structured patterns of the text. This part is an important contribution, because many tools in PLN are not capable to report this type of results.

As an example, in this work was performed a set of experiments in a set of documents written in Spanish by different authors: CS Lewis, Darren Shan, J. K. Rowling, Justin Somper, Jules Verne and Rick Riordan (translated into Spanish)[5]. The objective of the task was to identify particular patterns of each author. These results can be used in tasks called authorship attribution. In this experiment, the RDT representation was select, because is capable to represent structure of the input text, which could represent writing styles. One example of interesting patterns discovered in this process is shown in figure 10.

[4] For more info consult http://ailab.wsu.edu/subdue/
[5] These books can be downloaded from http://espanol.free-ebooks.net/

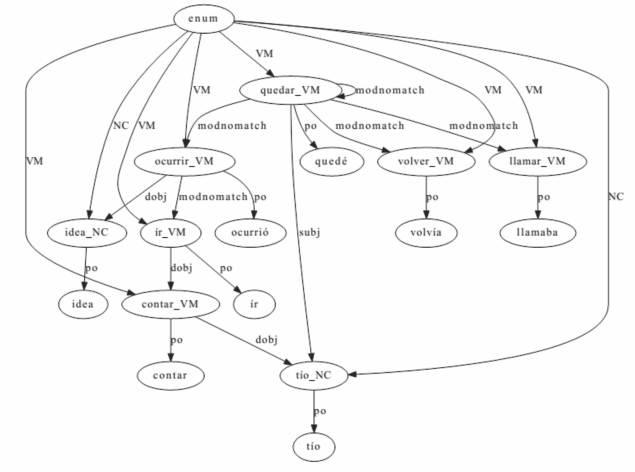

Fig. 10. Example of a pattern discovered with the RDP Representation for the authorship attribution task.

5 Conclusions

In this paper was proposed a methodology useful to represent texts with labeled graphs, with the aim to discover hidden patterns in text with a graph-based data mining process. The methodology is divided in three main steps: a preprocessing phase, the mapping process (where text are transformed to graphs), and the data mining process.

The preprocessing phase is focused to clean, remove unuseful words, splitte the text in sentences, and assign a set of labels that allow to determine the role of each word in the text. Based on this information, in this work is introduced three graph-based representation useful to map texts to graphs: STR, RBWS, and RDT. Each of them is capable to represent different information of the input text, such as word frequency, relations between words and lemmas, structure of the sentence, etc. Also, is presented a brief discussion of what kind of patterns can be discovered with each representation. Finally, is presented an example where was used this methodology to discover styles of writing in a set of text of different authors.

This paper only expose the idea of the authors of how it is possible to discover knowledge from text with a new alternative method with respect to the techniques used in the PLN area. Then, as future work, we need to prove that this proposal is useful to analyze and discover hidden patterns in real applications. Also, it is necessary to perform a theoretical analysis of these representations, with the aim to prove interesting properties such as time complexity and space complexity costs.

References

1. Deerwester, S., Dumais, S.T., Furnas, G.W., Landauer, T.K., Harshman, R.: Indexing by latent semantic analysis. Journal of the American Society for Information Science 41(6), 391–407 (1990)
2. Harish, B.S., Guru, D.S., Manjunath, S.: Representation and classification of text documents: A brief review. IJCA,Special Issue on RTIPPR (2), 110–119 (2010), published By Foundation of Computer Science
3. Harris, Z.: Distributional structure. Word 10(23), 146–162 (1954)
4. Manning, C.D., Schütze, H.: Foundations of statistical natural language processing. MIT Press, Cambridge, MA, USA (1999)
5. Witten, I.H., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques, Second Edition (Morgan Kaufmann Series in Data Management Systems). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (2005)