International Journal of Computer Applications (0975 - 8887)
Volume 123 - No.15, August 2015

Author verification using a Graph-based Representation

Esteban Castillo
Universidad de las Américas Puebla
Department of Computer Science, Mexico

Ofelia Cervantes
Universidad de las Américas Puebla
Department of Computer Science, Mexico

Darnes Vilariño
Benemérita Universidad Autónoma de Puebla
Faculty of Computer Science, Mexico

David Báez
Universidad de las Américas Puebla
Department of Computer Science, Mexico

ABSTRACT

This paper presents a methodology for tackling the authorship verification problem. The approach is based on comparing the similarity between a given unknown document against the known documents using a graph representation that captures the syntactic sequence of texts and a graph similarity measure. An unknown document can be classified as having been written by the same author if the majority of the comparisons surpass a predefined threshold. The best results were obtained on the Clef PAN 2014 dataset: 79% for the Spanish and 68% for English, showing that the proposed methodology could be a way for determining a document authorship.

General Terms

Authorship problem, graph representation, classification

Keywords

Authorship Verification, Syntactic Sequence Graph, Graph Similarity

1. INTRODUCTION

Discovering the writing style patterns of a text in order to unambiguously attribute the authorship of a given anonymous document is a very hard problem [1, 2] that has become of high interest in areas like Information Retrieval (IR), Natural Language Processing (NLP) and computational linguistics. The most common framework for mapping this kind of problem is the **authorship attribution**, where given texts of uncertain authorship and sample documents from a small, finite set of candidate authors, the problem consists of mapping the uncertain texts onto their true authors among the candidates. While the authorship attribution is considered as an unreasonably easy task, a more demanding problem is the **authorship verification**, where a set of documents written by a single author (usually very few) and a questioned document is given, the problem is to determine whether the questioned document was written by that particular author or not. In this sense, the importance of finding the correct features through the use of a proper representation is fundamental for solving this kind of problems.

In this paper a classification method that does not use classical classification algorithms in the context of authorship verification is evaluated. It is proposed to compare documents associated to an author and an unknown document using a graph representation that keeps the syntactic sequence of texts and a graph similarity measure in order to detect if a document belongs to an author. The use of a graph representation allows to map in a simple data structure the relationship between one word over another one in texts where there is a syntactic order.

The rest of this paper is organized as follows. Section 2 briefly discusses current work in the area of graph representation and authorship verification. Section 3 provides details on the design and implementation of the methodology used to verify the real author of a set of unknown documents. In Section 4 the experimental results are presented and discussed. Finally, implications and conclusions derived from this work thus far are presented in Section 5.

2. RELATED WORK

Recent research has shown that different NLP tasks may be accurately modeled by means of structures based on graph theory. Even though these two disciplines may be perceived as different topics, they share a common trait which may lead to obtain efficient solutions. There exist research works that employ graph representations and algorithms in order to deal with many NLP problems [3]. However, it can be seen that they only exploit a minimum part of their great potential. Actually, major works in the literature focus their efforts on representing a reduced number of different levels of natural language descriptions such as lexical, morphological, syntactic, sentence-level, semantic, etc.

The most relevant survey of existing work on the use of graphs in the IR field is presented in [4]. This work makes special emphasis in the creation of different graph representations that captures the syntactic information of texts. One of the main contributions of this survey is the use of different graph representations based on the co-occurrence of words, which is a simple but effective way

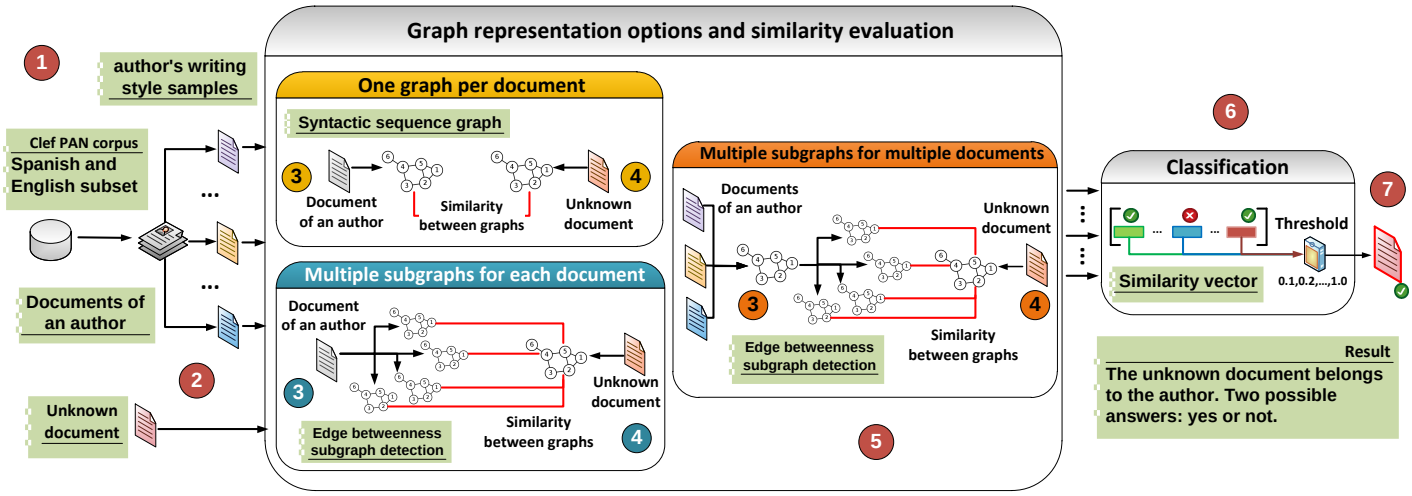

Fig. 1. Authorship verification methodology.

to represent the relationship of one term over another one in texts where there is no syntactic order.

On the other hand, in [5, 6] different methods are described for extracting semantic graphs through the text, where each vertex corresponds to a word and the edges represent the semantic dependency between two words. In this representation, these papers propose the generation of knowledge through relationships established between the words and the semantic dependencies (or categories). Such discovered knowledge is represented as a new graph showing the essential information of an author. Moreover, in [7] the process of information discovery over large volumes of information is explained through the use of knowledge graphs, also called graphs of semantic domain, where information is presented as the union of the key concepts that represent a topic in particular. Besides showing the key elements for representing documents using graphs (taking into account the use of syntactic features like n-grams on a level of the graph), special emphasis is placed on finding significant patterns in a supervised and unsupervised manner, making a comparison of them and describing their advantages and disadvantages related to other graph representations.

While many papers focus on the extraction of syntactic and semantic features using a graph representation, others [8, 9] explore the idea of features based on the interaction of terms like in a social network, where it is important to find the most relevant words (trending ones) based on the relations they have. Other papers [10, 11] use the idea of the interaction of terms to find communities (subgraphs) which could be used in the comparison of text documents that are related. Research works that use graph representations for texts in the context of authorship problem barely appear in the literature [12, 13]. the concept of n-grams with a frequency of occurrence vector has usually has been proposed to solve it [14]. However, there is still an enormous gap between this approach and the use of more detailed graph structures that represent in a natural way the text associated to the documents.

3. AUTHORSHIP VERIFICATION PROCESS USING A GRAPH REPRESENTATION

For tackling the authorship verification problem a methodology is proposed (see Figure 1) in which it is implemented a classification method based on the use of a graph representation and a graph similarity measure. The methodology consists of seven steps:

(1) Select a dataset (corpus) for the authorship verification problem (see section 4.1).

(2) Preprocess all documents in the dataset. This task includes elimination of punctuation symbols and all the elements that are not part of the ASCII encoding. Then, all the remaining words are changed to lowercase.

(3) Map all the documents associated with an author to a graph representation (see subsection 3.1 and 3.3) taking into account one of the following options:
- **One graph per document**: for each document of an author, create a graph representation.
- **Multiple subgraphs for each document** : for each document of an author, create a graph representation and then, obtain the four most important subgraphs using the edge betweenness algorithm [11].
- **Multiple subgraphs for multiple documents**: Map all documents associated to an author to a graph representation. Then, obtain the four most important subgraphs using the edge betweenness algorithm.

(4) Map each unknown document to a graph representation (see section 3.1).

(5) Obtain the similarity (see subsection 3.2) between each graph associated to an author and the unknown graphs.

(6) The values obtained in the previous step are tested against a proposed threshold (Δ) to determine if the author graphs are similar to the unknown graphs.

(7) If each unknown graph is similar to most of the graphs associated to an author, then the unknown document belongs to the author, otherwise it does not belong (a binary classification).

3.1 Syntactic sequence graph

Among different proposals for mapping texts to graphs, the use of the sequence in which the terms appear in a sentence is an effective way to represent the relationship and importance of one term over another one in texts where there is a strict syntactic order [3, 4] (usually literary, essays and news texts). Formally, a syntactic sequence graph is represented by $G = (V, E, L, \alpha)$, where:

- G is a directed graph.
- $V = \{v_i | i = 1, ..., n\}$ is a finite set of vertices that consists of the words contained in either one or several texts.
- $E \subseteq V \times V$ is the finite set of edges which represents that two vertices are connected by means of the sequence of the text if their corresponding lexical units appear together in a sentence.
- L is the edges tag set which consists of the number of times that two vertices appear together in the text.
- $\alpha : E \rightarrow L$ is a function that assigns a tag to a pair of associated vertices.

As an example, consider the following sentence ζ extracted from a text T in the English dataset[1]: *"The violence on the TV. The article discussed the idea of the amount of violence on the news."*, which after the preprocessing stage (see Section 3) would be as follows: *the violence on the tv the article discussed the idea of the amount of violence on the news*. Based on the proposed representation, preprocessed sentence ζ can be mapped to the syntactic sequence graph shown in Figure 2.

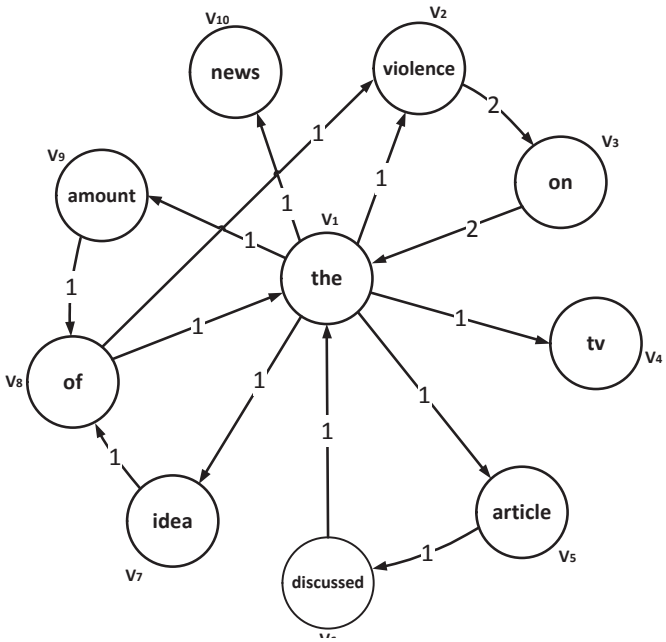

Fig. 2. Syntactic sequence graph example.

The Syntactic sequence graph shown in figure 2 has the following features:

- The set of vertices consists of the preprocessed words in sentence ζ considering that if there are multiple occurrences of a word, only one vertex is created.
- An edge between two vertices represents those words that appear together (next to another) in a sentence, at least once.
- The edge direction represents the order in which words appear in the sentence.
- The label edge between two vertices represents the number of times that two words appear together in a sentence ζ.

3.2 Graph similarity

Figure 3 shows the proposed steps to calculate the similarity of two graphs, considering the use of a vertex similarity measure. The process consists of four steps:

(1) Obtain all vertices (words) that share the authors graph and an unknown graph.

(2) Apply the dice similarity measure [15] for each graph, taking as input the shared vertices of the previous step and the graph to analyzed. The result is a matrix that represents the similarity scores for each pair of input vertices, based on their connections patterns. Formally, dice similarity calculates the similarity of two vertices as twice the number of common neighbors divided by the sum of the degrees[2] of the vertices.

(3) Obtain the upper triangular values for each matrix and use them to build a vector representation [16]. The rest of the matrix values are not useful, because the main diagonal represents the similarity of a input vertex with itself and the lower triangular is the same as the upper one.

(4) Apply the cosine similarity or the Euclidean distance [17] between the two vectors. The result is a value in the range of 0 to 1 that indicates how similar the two graphs are.

The idea behind this measure is to compare the topological patterns of the vertices that share both graphs[3] based on how the author uses the words in the syntactic sequence of texts. The main advantage of this graph similarity evaluation is that we can compare graphs with different number of vertices and edges which differs from the common algorithms for matching graphs [18] and from the use of classical similarity measures to compare vectors, like cosine or Euclidean.

3.3 Subgraph detection

The edge betweenness is an **algorithm for discovering subgraphs** based on the use of the betweenness centrality measure[4] [19]. The idea of this algorithm is that the **betweenness of the edges** connecting two subgraphs is typically high, as many of the shortest paths between vertices in separate subgraphs go through them. So if we gradually remove the edge with highest betweenness from the network, and recalculate edge betweenness after every removal, sooner or later the network falls off to two components. Then after a while one of these components falls off to two smaller components, etc. until all edges are removed. This is a divisive hierarchical approach, the result is a set of subgraphs that are densely connected themselves.

[1] It is the same process for the Spanish dataset.

[2] The degree of a vertex is the total number of vertices adjacent to the vertex.

[3] Considering that always share a set of stop words like the articles, etc.

[4] Indicator of how often a vertex/edge is located on the shortest path between other vertices/edges in the graph.

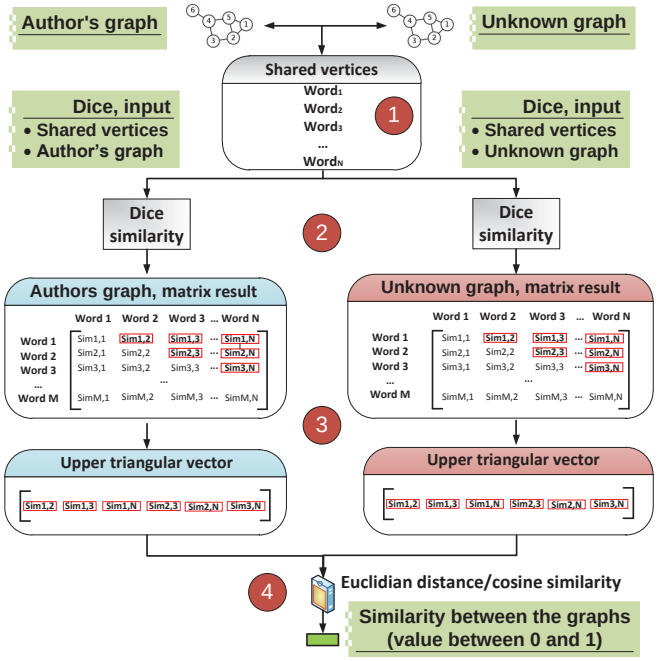

Fig. 3. Graph similarity evaluation process.

In this paper the edge betweenness is used in step four of the proposed methodology, choosing one of the following interpretations:

- **One graph per document**: Taking into consideration that many documents associated with an author have different topics and are not related to each other, the documents of the dataset without the subgraph detection process are used.
- **Multiple subgraphs for each document**: In order to obtain more positive samples associated with an author (one of the main problems of the authorship verification), the most representative subgraphs of each document are extracted.
- **Multiple subgraphs for multiple documents**: Considering that many documents have the same topic, a graph of all the documents associated with an author is created and then the positive subgraph samples are obtained by means of the algorithm. The resulting subgraphs incorporate the syntactic sequence of all documents rather than the isolated ones.

As an example, consider figure 4, which shows the graph representation[5] of the text document that includes sentence ζ (see subsection 3.1). The graph contains 629 vertices (different words) and 9813 edges, showing the type of graphs that are obtained from a text document in the dataset. In order to obtain the four most representative subgraphs the edge betweenness algorithm is applied:

(1) Figure 5 shows the graph after having applied the algorithm, where each color represents the vertices of a subgraph. For this example the algorithm gets 12 different subgraphs based on the betweenness of the edges.

(2) Figure 6 illustrates representative subgraphs after removed those with less than 20 vertices. Note that the the number of vertices is based on the average sentence length of the documents in the dataset.

(3) Finally, figure 7, shows the four most representative subgraphs, considering the subgraphs with a greater number of vertices and edges.

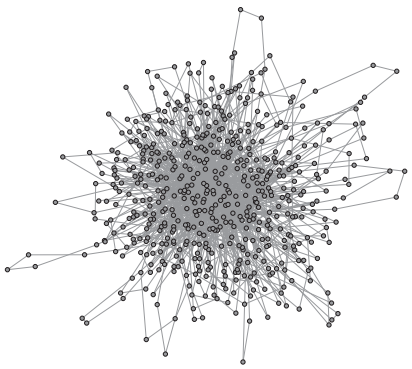

Fig. 4. Original graph representation.

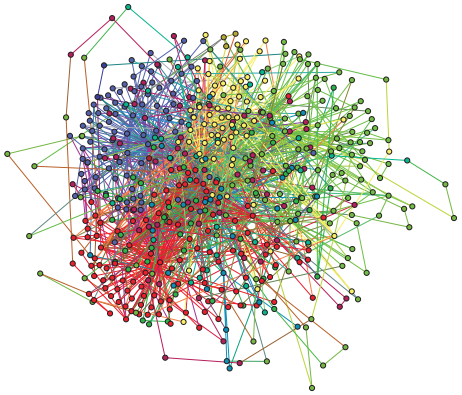

Fig. 5. Subgraphs obtained using edge betweenness.

4. EXPERIMENTAL RESULTS

Results obtained with the proposed methodology are discussed in this section. First, the dataset used is described, then the experiments performed, and finally the results of each experiment.

4.1 Dataset

The document collection used in the experiments is a subset of the **Clef PAN 2014 corpus**[6][20], which includes, several text documents in Spanish and English on different topics and genres. The dataset is divided in two groups:

- **Training documents:** It contains a set of authors each one with a set of known documents.

[5]All graph visualizations were created using Gephi: http://gephi.github.io/

[6]http://www.uni-weimar.de/medien/webis/events/pan-14/

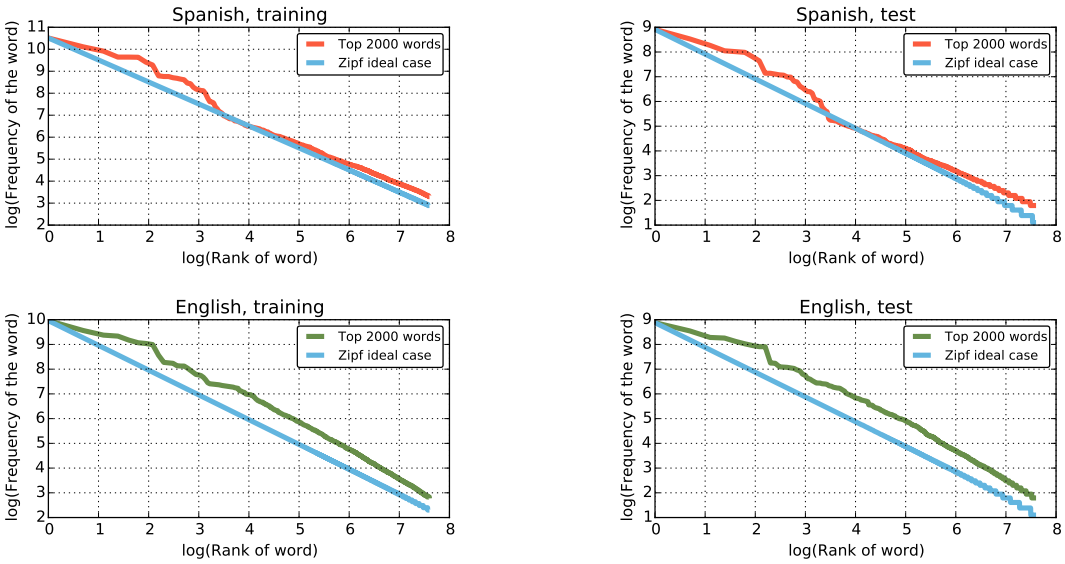

Fig. 8. Similarity of term distributions according to the Zipf law.

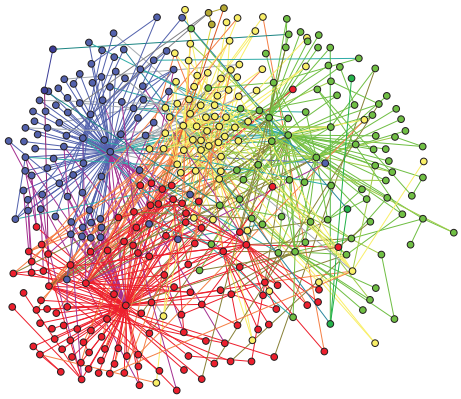

Fig. 6. Relevant subgraphs.

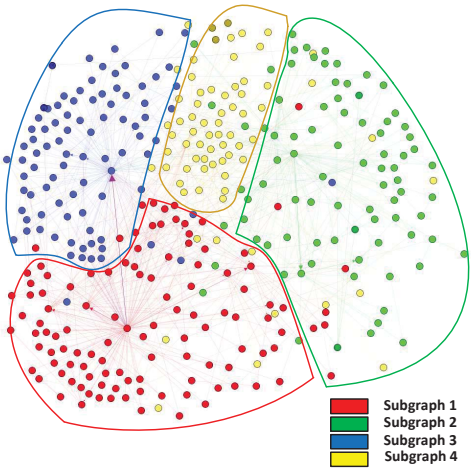

Fig. 7. Selected representative subgraphs.

- **Test documents:** It contains a set of unknown documents, each one with a possible author and a label that indicates whether the document belongs to that author. These documents are used to test the methodology taking into account the writing style samples of the training documents.

In Table 1, main dataset features are shown, including the number of authors and documents for the training and test documents.
As can be seen in the table, the training and the test dataset are quite similar in terms of their features. Both were tailored for the particular problem of authorship verification. Figure 8 indicates that the term frequency distribution of the training and test dataset is close to the Zipf distribution (Power law) [21], leading to consider that the dataset documents have been written in a general writing style, which highlights the challenge of discovering the writing patterns for each author.

4.2 Experiments performed

Experimentation consists of one hundred and twenty different combinations of the main features associated to the methodology (see table 2). For each combination a threshold between 0 and 1 is used to decide if a document belongs to an author. In the case of the combinations that use the subgraph detection process, the best four subgraphs are extracted considering a preliminary phase in which is tested a range between 1 and 10 possible subgraphs. For the dice similarity measure, the edge betweenness algorithm and the implementation of the different graphs, the igraph[7] tool [22] implemented in Python is used [8] with the **default input values**.

[7] http://igraph.org/python/doc/igraph.Graph-class.html
[8] https://www.python.org/

Table 3. Evaluation of the methodology using the test dataset.

Spanish dataset							
Graph option	**Vector similarity**	**Vertex similarity**	**Threshold Δ**	**Accuracy %(correct)**	**Precision, yes %(correct)**	**Precision, not %(correct)**	**Methodology runtime**
C	Euclidean		0.7	79	76.8	75.9	00:04:12
C	Cosine	Dice	0.6	76	71.4	68.5	00:03:55
B	Euclidean	similarity	0.7	75	73.5	69.5	00:05:11
C	Euclidean		0.6	73	67.3	70.5	00:03:35
A	Euclidean		0.5	68	65.3	68.7	00:02:29
English dataset							
Graph option	**Vector similarity**	**Vertex similarity**	**Threshold Δ**	**Accuracy %(correct)**	**Precision, yes %(correct)**	**Precision, not %(correct)**	**Methodology runtime**
B	Euclidean		0.8	68.5	63.1	67.2	00:07:32
B	Euclidean	Dice	0.6	66	71.4	65.5	00:07:12
B	Cosine	similarity	0.8	64.5	68.3	73.4	00:06:58
A	Euclidean		0.7	62.5	54	69.3	00:01:52
C	Euclidean		0.8	60	58.9	74.6	00:04:37

A: One graph per document.
B: Multiple subgraphs for each document.
C: Multiple subgraphs for multiple documents.

Table 1. Dataset features.

Language dataset	**Feature**	**Training**	**Test**
Spanish	Number of documents	500	100
	Number of authors	100	100
	Documents per author	5	1
	Avg. words per document	1135	1121
	Avg. words per sentence	25	21
	Vocabulary size	41583	12764
	Genre	Articles	
English	Number of documents	518	200
	Number of authors	200	200
	Documents per author	1-5	1
	Avg. words per document	848	833
	Avg. words per sentence	26	22
	Vocabulary size	52421	14129
	Genre	Essays	

Table 2. Experimental features.

Language dataset	**Feature**	**A**	**B**	**C**
Spanish	# Experiments performed	20	20	20
English	# Experiments performed	20	20	20
Both languages	# Subgraphs extracted	0	4	4
	Threshold Δ	0.1, 0.2, ..., 1.0		
	Supgraph detection algorithm	Edge betweenness		
	Vertex similarity measure	Dice similarity		
	Vector similarity measure	Euclidean/Cosine		
	Programming language used	Python		
	Computer processor	Intel core 2 duo 2.00GHz		
	Ram memory	2 GB		

A: One graph per document.
B: Multiple subgraphs for each document.
C: Multiple subgraphs for multiple documents.

Finally, for the vector similarity measures (cosine and Euclidean) the scikit learn tool is used[9] [23] (also implemented in Python).
It is important to notice that it is applied the same number of experimental combinations for both language datasets. The idea is to test the performance of the graph options proposed (see section 3), in text documents that have different lexical and syntactic elements.

4.3 Obtained results

In Table 3 the best five results obtained using the methodology are presented. The results were evaluated according to the accuracy measure [17] and the macro precision measure[10] [16], which are well known evaluation measures for binary classifiers.

The approach obtained an acceptable accuracy and execution time. The **best results** for the English dataset were obtained because more positive graph samples were extracted, considering that each author had few documents. For the Spanish dataset the training graphs were combined, taking into account that the majority of the documents associated to an author are related to each other.

The threshold (Δ) used to verify the authorship shows that most of the English documents related to an author must have high similarity (between 0.6 and 0.8) with each one of the unknown documents. In the case of the Spanish documents the similarity expressed in the threshold is lower (between 0.5 and 0.7), which evidences that it is necessary to have more positive samples of the author's writing style, in order to verify the authorship of documents that have words with different meanings in the syntactic sequence of texts[11].

The precision for both datasets indicates that the methodology obtained balanced results. In other words, the methodology does not tend to classify one of the possible answers more often,

[9] http://scikit-learn.org/stable/
[10] Calculates the precision for the two possible classifier results.
[11] For example the Spanish word *banco* has 4 different meanings in the dataset.

despite the fact that some authors do not have the same number of documents, especially in the English dataset.

Further analysis in the use of the subgraph detection algorithm and in the construction of the graph will allow us to find more accurate features that can be used in a classification method for the authorship verification problem.

5. CONCLUSIONS AND FUTURE WORK

An approach that uses a classification method with a graph-based representation has been presented. The results obtained show a competitive performance compared with other approaches that use the same dataset [20, 24, 25]. In particular, the methodology implemented presents an accuracy greater than 70% for the Spanish dataset and over 60% for the English dataset. Research on graph theory is continuing in order to improve the results obtained. Further work includes:

- Experiment with other graph representations for texts that include alternative levels of language descriptions such as the use of sentence chunks, PoS tags, etc [26].
- Use different algorithms to find subgraphs, based on the patterns of a graph representation like leading eigenvector [27] or label propagation [28].
- Propose a similarity measure that uses the semantic information of a graph representation [29].
- Compare the methodology presented with other classical approaches like the N-gram model [30].
- Test the proposed methodology on the authorship attribution problem using the best experimental combinations applied in this paper.
- Test the proposed methodology on the sentiment analysis problem [31], where text documents could be smaller and the use of slang and genre-specific terminology is usual.
- Explore different supervised/unsupervised classification algorithms [7] in order to improve the results presented in this paper.

Acknowledgements

This work has been supported by the CONACyT grant with reference #373269/244898 and the CONACYT-PROINNOVA project no. 0198881.

6. REFERENCES

[1] Patrick Juola. Authorship attribution. *Foundations and Trends in Information Retrieval*, 1(3):233–334, 2008.

[2] Moshe Koppel, Jonathan Schler, and Shlomo Argamon. Computational methods in authorship attribution. *Journal of the American Society for Information Science and Technology*, 60(1):9–26, 2009.

[3] Rada Mihalcea and Dragomir Radev. *Graph-based natural language processing and information retrieval*. Cambridge University Press, 2011.

[4] S. S. Sonawane and P. A. Kulkarni. Article: Graph based representation and analysis of text document: A survey of techniques. *International Journal of Computer Applications*, 96(19):1–8, 2014.

[5] Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D. Manning. Generating typed dependency parses from phrase structure trees. In *LREC*, pages 449–454, 2006.

[6] Aria Haghighi, Andrew Y. Ng, and Christopher D. Manning. Robust textual inference via graph matching. In *EMNLP*. The Association for Computational Linguistics, 2005.

[7] Diane J. Cook and Lawrence B. Holder. Graph-based data mining. *IEEE Intelligent Systems*, 15(2):32–41, 2000.

[8] L.C. Freeman. *The Development of Social Network Analysis: A Study in the Sociology of Science*. BookSurge Publishing, 2004.

[9] S. Wasserman and K. Faust. *Social network analysis: Methods and applications*. Cambridge Univ Pr, 1994.

[10] Santo Fortunato. Community detection in graphs. *Physics Reports*, 486:75–174, 2010.

[11] M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Phys. Rev. E*, 69(2):026113, February 2004.

[12] R. Arun, V. Suresh, and C. E. Veni Madhavan. Stopword graphs and authorship attribution in text corpora. In *ICSC*, pages 192–196. IEEE Computer Society, 2009.

[13] Darnes Vilariño, David Pinto, Helena Gómez-Adorno, Saul León, and Esteban Castillo. Lexical-syntactic and graph-based features for authorship verification notebook for pan at clef 2013. In *CLEF (Working Notes)*, volume 1179 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2013.

[14] Efstathios Stamatatos. A survey of modern authorship attribution methods. *Journal of the American Society for Information Science and Technology*, 60(3):538–556, 2009.

[15] Lada A. Adamic and Eytan Adar. Friends and neighbors on the web. *Social Networks*, 25(3):211–230, 2003.

[16] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[17] Christopher D. Manning and Hinrich Schütze. *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, Massachusetts, 1999.

[18] L. P. Cordella, P. Foggia, C. Sansone, and M. Vento. An improved algorithm for matching large graphs. In *In: 3rd IAPR-TC15 Workshop on Graph-based Representations in Pattern Recognition, Cuen*, pages 149–159, 2001.

[19] L.C. Freeman. Centrality in Social Networks: Conceptual Clarification. *Social Networks*, 1:215–239, 1979.

[20] Efstathios Stamatatos, Walter Daelemans, Ben Verhoeven, Benno Stein, Martin Potthast, Patrick Juola, Miguel A. Sánchez, and Alberto Barrón. Overview of the author identification task at PAN 2014. In *Working Notes for CLEF 2014 Conference, Sheffield, UK, September 15-18, 2014.*, pages 877–897, 2014.

[21] G. Zipf. *Selective Studies and the Principle of Relative Frequency in Language*. Harvard University Press, Cambridge, MA, 1932.

[22] Gabor Csardi and Tamas Nepusz. The igraph software package for complex network research. *InterJournal*, Complex Systems:1695, 2006.

[23] Fabian Pedregosa. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[24] Mahmoud Khonji and Youssef Iraqi. A slightly-modified gi-based author-verifier with lots of features (asgalf). In *CLEF (Working Notes)*, volume 1180 of *CEUR Workshop Proceedings*, pages 977–983. CEUR-WS.org, 2014.

[25] Esteban Castillo, Ofelia Cervantes, Darnes Vilariño, David Pinto, and Saul León. Unsupervised method for the authorship identification task. In *CLEF (Working Notes)*, volume 1180 of *CEUR Workshop Proceedings*, pages 1035–1041. CEUR-WS.org, 2014.

[26] S. P. Abney. Parsing by chunks. In Robert C. Berwick, Steven P. Abney, and Carol Tenny, editors, *Principle-Based Parsing: Computation and Psycholinguistics*, pages 257–278. Kluwer, 1991.

[27] MEJ Newman. Finding community structure in networks using the eigenvectors of matrices. *Physical Review E*, 74(3):36104, 2006.

[28] Usha Nandini Raghavan, Réka Albert, and Soundar Kumara. Near linear time algorithm to detect community structures in large-scale networks. *Phys. Rev. E*, 76, 2007.

[29] Marco A. Alvarez and Changhui Yan. A graph-based semantic similarity measure for the gene ontology. *J. Bioinformatics and Computational Biology*, 9(6):681–695, 2011.

[30] Efstathios Stamatatos. Author identification: Using text sampling to handle the class imbalance problem. *Inf. Process. Manage.*, 44(2):790–799, March 2008.

[31] Bo Pang and Lillian Lee. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1-2):1–135, 2008.