Document embeddings learned on various types of n-grams for cross-topic authorship attribution

Article *in* Computing · January 2018

DOI: 10.1007/s00607-018-0587-8

4 authors:

Helena Gomez Adorno
Universidad Nacional Autónoma de México
47 PUBLICATIONS **531** CITATIONS
SEE PROFILE

Juan Pablo Francisco Posadas Durán
Instituto Politécnico Nacional
17 PUBLICATIONS **169** CITATIONS
SEE PROFILE

Grigori Sidorov
Instituto Politécnico Nacional
226 PUBLICATIONS **1,745** CITATIONS
SEE PROFILE

David Pinto
Benemérita Universidad Autónoma de Puebla
176 PUBLICATIONS **1,047** CITATIONS
SEE PROFILE

Preprint of an article accepted in Computing
https://doi.org/10.1007/s00607-018-0587-8

Document Embeddings Learned on Various Types of n-grams for Cross-Topic Authorship Attribution

Helena Gómez-Adorno · Juan-Pablo Posadas-Durán · Grigori Sidorov · David Pinto

Received: 19 June 2017 / Accepted: 6 January 2018

Abstract Recently, document embeddings methods have been proposed aiming at capturing hidden properties of the texts. These methods allow to represent documents in terms of fixed-length, continuous and dense feature vectors. In this paper, we propose to learn document vectors based on n-grams and not only on words. We use the recently proposed Paragraph Vector method. These n-grams include character n-grams, word n-grams and n-grams of POS tags (in all cases with n varying from 1 to 5). We considered the task of Cross-Topic Authorship Attribution and made experiments on *The Guardian* corpus. Experimental results show that our method outperforms word-based embeddings and character n-gram based linear models, which are among the most effective approaches for identifying the writing style of an author.

Keywords Document Embeddings · Authorship Attribution · doc2vec · neural networks · n-grams

1 Introduction

The Authorship Attribution (AA) is a useful and well-studied task in natural language processing. The aim of AA is to identify the author of a given text among a list of candidates. The AA task can be used in applications of electronic commerce [1], forensics [4] and humanities research [13]. In single-topic

H. Gómez-Adorno · G. Sidorov
Instituto Politécnico Nacional (IPN), Center for Computing Research (CIC), Mexico City, Mexico. E-mail: helena.adorno@gmail.com, sidorov@cic.ipn.mx

J. Posadas-Durán
Instituto Politécnico Nacional (IPN), Escuela Superior de Ingeniería Mecánica y Eléctrica Unidad Zacatenco (ESIME-Zacatenco), Mexico City, Mexico. E-mail: jposadasd@ipn.mx

D. Pinto
Benemérita Universidad Autónoma de Puebla (BUAP), Faculty of Computer Science, Puebla, Mexico. E-mail: dpinto@cs.buap.mx

AA, the topic and genre are the same in the training and the testing documents. Meanwhile in cross-topic AA, a more realistic scenario is evaluated, when the training and the testing documents belong to different topics and genres.

The AA task can be considered as a classification problem. A classifier is trained using training document (their corresponding vectors) and their author labels. The authors of a testing document are predicted using the vectors of the test documents and the classification model obtained in training. The quality of the document vectors directly affects the performance of the AA model. Character n-gram based methods have been widely used to represent documents for AA under both single and cross-topic AA conditions [30,31]. However, in these methods, each n-gram is taken as a unique symbol, which is absolutely different from other n-grams. Thus, the semantic information is lost. Representing n-grams (or other linguistic features) as unique, discrete tokens leads to data sparsity and in order to train successfully a statistical model much more data is necessary.

Neural network based distributed representations overcome some of these obstacles [2]. Distributed representations of word, phrases, sentences, paragraphs, and documents have the capability to encode semantic information of texts and represent them in a continuous vector space, when semantically similar objects are mapped to nearby points. Sentence and document embeddings have been proposed for tasks related with text analysis. For example, in sentiment analysis, recurrent neural network [29], convolutional neural network [9] and skip thought vectors [11] achieve state-of-the-art results.

The Paragraph Vector algorithm (also known as Doc2vec) [14] is a computationally-efficient predictive model for learning distributed vector representations at the document level by treating each document as a special word and learn both document vectors and word vectors simultaneously by predicting the target word. In this paper, we explore the document embeddings learned by Doc2vec using not only words, but also n-grams of characters, words and POS tags. In this way, we are able to model the semantics of documents along with their syntax, which have been proven to provide high accuracy in the authorship attribution related tasks [6,22,23].

The contributions of this work are:

- We introduce a new method for cross-topic authorship attribution based on document embeddings learned from n-grams of characters, POS tags and words.
- We demonstrate that the embeddings learned from n-grams by themselves and in combination improve the performance of cross-topic authorship attribution.

The rest of the paper is structured as follows. Section 2 presents a brief description of the authorship attribution problem and the most successful document embedding methods. Section 3 describes our method for the authorship attribution problem using document embeddings learned on n-grams of characters, POS tags and words. Section 4 describes the experimental settings and the corpus. Section 5 presents the obtained results and their evaluation. Finally, Section 6 draws the conclusions and points to the possible directions of future work.

2 Related Work

Various techniques were developed for solving the AA task. In [24], the authors present a reproducibility study on AA research. They evaluated state-of-the-art methods on three corpora and showed that only four out of fifteen (4/15) approaches are stable across corpora. The majority of the studies on AA perform extensive feature engineering, focusing on the extraction of stylometric features that represent the personal style of authors [7,28,25]. The approaches that employ character-based features (character n-grams) seem to be the most effective ones for the AA problem under both single and cross-topic conditions [5,25].

Previous work on AA focused mainly on the single-topic condition, i.e., the training and testing datasets have similar thematic properties. However, there are studies that tackle the AA problem under cross-topic conditions. Stamatatos [31] demonstrated that high frequency character n-grams allow to discriminate effectively between authors not only for single-topic AA, but also for cross-topic AA. The unmasking method yields reliable results under cross-genre [10] and cross-topic conditions [12]. Sapkota *et al.* [26] improve the prediction results in cross-topic AA using an enriched training corpus in order to predict authors on a corpus with different topics.

The role of preprocessing steps was evaluated in [18]. The approach proposed in that paper is considered to be more topic-neutral by their authors, because they replace the named entities and some topic-related words while preprocessing the corpus. Their approach showed the importance of preprocessing, because it gave the improvement of 4%.

In [30], it is mentioned that the use of semantic features for the authorship attribution task usually improves the obtained results, however, very few attempts have been done to exploit high-level features for stylometric purposes. In this paper, we consider the usage of the distributed document representation for the cross-topic AA task, because of its capability to encode the semantic information of texts in a low dimension vector.

Different document embeddings methods were introduced in recent studies, each tackling specific natural language processing tasks: weighted concatenation of word vectors [16], deep averaging network [8], paragraph vector n-gram model [15], recurrent neural networks [29], and convolutional neural networks [9].

Recently, the Paragraph Vector (Doc2vec) model was proposed by Le & Mikolov [14] for learning distributed representation for both sentences and documents. The Doc2vec model basically treats each document as a special word and learn both document vectors and word vectors simultaneously by predicting the target word. Vectors obtained by the Doc2vec model outperforms both bag-of-words and word n-grams models producing the new state-of-the-art results for several text classification and sentiment analysis tasks.

Li *et al.* [15] presented a model that is able to capture both semantics and word order in documents for document level sentiment analysis, by predicting not only its belonging words, but word n-gram features as well. The model outperforms previous deep learning models and bag-of-n-gram based models for sentence level sentiment analysis.

In this work, we introduce the use of document embeddings learned on different types of n-grams (characters, POS tags and words) using the well-known Doc2vec model. We show that these document embeddings achieve state-of-the-art results on the cross-topic and cross-genre authorship attribution task. We also give some insights about the parameter selection of the Doc2vec algorithm.

3 Authorship Attribution with Various Types of Document Embeddings

The Authorship Attribution (AA) task consists in identifying the author of a given text among a list of candidates authors. Two variations of the AA task are known: the closed AA, when it is known that the author of the text is in the list of the candidates, and the open AA, when the author may or may not be in the list of candidate authors. In this work, we focus on the closed AA task.

Different approaches have been proposed to solve the AA task, but the best results nowadays are obtained with machine learning approach. In this approach, the problem is treated as a supervised classification task, when a classifier is built using a training set and the task consists in classifying correctly the samples from a testing set.

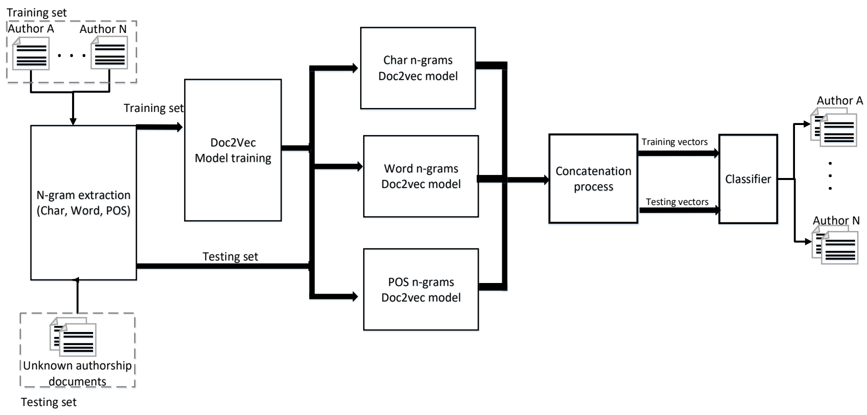


Fig. 1: General description of the proposed approach

Figure 1 gives general description of the proposed method to solve the AA task based on a machine learning approach. At the training stage, the corpus is processed in order to represent each text with one of the following types of n-grams: character n-grams, n-grams of POS tags, and word n-grams (in all cases with n varying from 1 to 5). For illustration purposes, we present an example of the types of n-grams for the phrase *"Hello I am here"* with n equal to 3 (note that the space is represented with the character "_"). For obtaining the POS tags of words in a sentence, we use the third party module of POS tagging from the NLTK toolkit [1].

Table 1: Various types of n-grams for the phrase "*Hello I am here*"

Input type	**Phrase representation**
Char 3-grams	Hel,ell,llo,lo_,o_I,I_a,_am,am_,m_h,_he,her,ere
Word 3-grams	Hello_I_am, I_am_here
POS 3-grams	NNP_PRP_VBP, PRP_VBP_RB

Note that the obtaining of each one of the types of n-grams presented in Table 1 is independent. We consider that the types of n-grams are independent of each other and therefore it is worth analyzing the efficiency of the document embeddings learned on each type of n-gram for solving the AA task independently or in combination.

After representing the training data in terms of n-grams, we use the Doc2vec method to obtain the document embeddings of the training documents. The Doc2vec module offers two possible approaches to build the model, the Distributed Model (DM), which tries to predict the context of a given element and the Distributed Bag of Words (DBOW), which tries to predict the word given the context [14].

The Doc2vec model transforms the raw text, which in our case is represented in terms of n-grams (characters, POS tags, or words) and returns a vector representation according to the selected model. The full explanation of how the vectors are built is described in [14, 19]. In this work, both model representations (DM and DBOW) are used separately and in combination: each model is trained independently and their outputs are appended (i.e., we got a new vector from the concatenation of the two vectors, e.g., the vector [a,b] and the vector [c,d] gives the vector [a,b,c,d]). In this way, we try to use the advantages of each model, leaving the task of the construction of the best model to the classifier.

Embeddings are based on context modeling. It means that the order of elements (characters, POS tags and words) is essential for modeling the writing style of an author using embeddings. In order to model these orders, the distributed representation of documents is learned by predicting word and character sequences. In this work, n-gram features are directly used as word or character sequence features. Each n-gram is treated as a special token and it

[1] `http://www.nltk.org/`

is directly considered as element for training of embeddings in each document. In this way, documents containing semantically similar n-grams also tend to be closer to each other in vector space.

We want to evaluate the effectiveness of the document embeddings trained on different types of n-grams, so apart from using both models of the of Doc2vec (DM and DBOW), we also consider the combinations of different types of n-grams (by appending (concatenating) their respective models (vectors)).

After the document embeddings are obtained for the training texts, a Logistic Regression classifier is trained. The Logistic Regression classifier is used because it simple, fast and has reported good results in the state-of-the-art works for authorship attribution task [17,27].

Finally, at the testing stage, the document embeddings for the testing corpus are obtained. There are two ways to obtain them: (1) Inferring the vector from the model (the vocabulary of the model is restricted to the training set), or (2) Retraining the model with the testing text (the vocabulary of the model considers the terms contained in the testing set). Note that retraining can be considered incorrect, because the network sees all instances during training (though at different stages), but it remains the common practice. We give results for both inferring and retraining.

Previous researches using the Doc2vec representation [16,15] suggested to retrain the model several times using the unlabeled corpus. Each time the model is retrained, the entry order of the documents is different from the one used in previous times. The model obtained in the last iteration tend to be more robust than the one obtained with just one iteration. Following the previous recommendation, we retrain the model ten times using only the training set with no additional resources, so our result is comparable with other researches.

Since the reproducibility of the experiments is an important concern, we decide to use a pseudo-random function named Fisher-Yates shuffle (also known as Knuth shuffle) [3] to generate entry order combinations of the documents to be used in the retrain of the model, but with fixed random seed equal to zero (0), so the function always returns the same combinations.

4 Datasets and Experimental Settings

The experiments were conducted on a cross-topic AA corpus introduced in [31]. The corpus contains a set of articles gathered from 1999 to 2009 from the English newspaper *The Guardian*[2]. The articles corresponding to 13 authors were collected and grouped into five topic categories: *Politics*, *Society*, *World*, *UK*, and *Book reviews*.

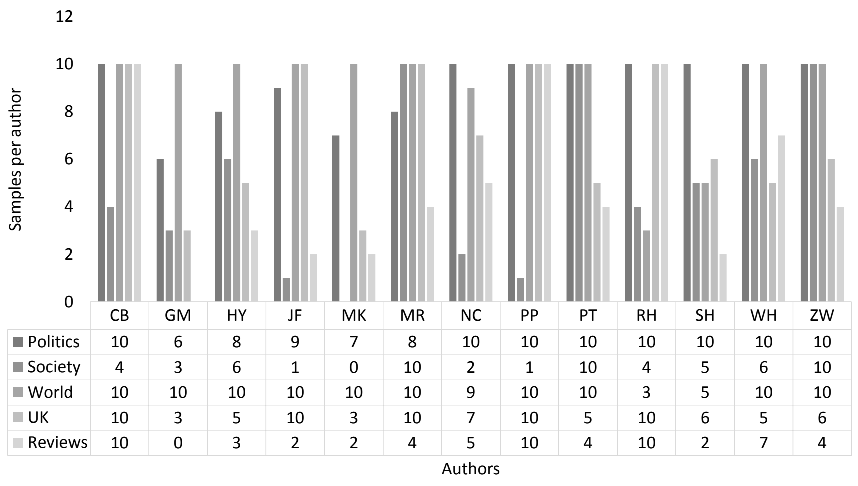
In order to avoid category overlapping, those articles whose content includes more than one category were discarded. In this way, each category is

[2] `https://www.theguardian.com`

mutually exclusive. An important remark about the proposed categories is that all the categories are composed by articles, but the *Book reviews* are considered different text genre. *The Guardian* corpus is a specific collection of samples, which provides both a cross-topic scenario (five different topics) and a cross-genre scenario (articles and reviews) for AA task.

The number of samples in *The Guardian* corpus is not balanced. The gathered samples correspond to a realistic scenario that considers the production of each author over a period of 10 years.

In order to test and compare our approach, we reproduce the testing scenario described in the previous research [31] using the *Guardian corpus*. The experimental scenario is as follows: (1) select at most ten samples per author in each topic category (in Figure 2 the distribution of the samples per author and per category after considering the restriction of ten samples per author is shown), (2) use the samples in the *Politics* category as training set and train the classifier, and (3) finally, test the classifier using another topic category different from *Politics* (four possible pairings).



	CB	GM	HY	JF	MK	MR	NC	PP	PT	RH	SH	WH	ZW
■ Politics	10	6	8	9	7	8	10	10	10	10	10	10	10
■ Society	4	3	6	1	0	10	2	1	10	4	5	6	10
■ World	10	10	10	10	10	10	9	10	10	3	5	10	10
■ UK	10	3	5	10	3	10	7	10	5	10	6	5	6
■ Reviews	10	0	3	2	2	4	5	10	4	10	2	7	4

Fig. 2: Author distribution in *The Guardian* corpus selecting at most ten samples per author

Table 2 presents general statistics of the five datasets in *The Guardian* corpus. We show the number of documents in each dataset (*No. of docs*), the statistics of the average number (*Avg.*) of words and characters per document, as well as standard deviation (*Std.*). In terms of corpus statistics, it can be observed that with respect to the *Politics* dataset, the datasets of *UK* and *Books* presents the most different statistics, being the later more different than the others. With respect to the to content in the datasets, Stamatatos [31] observed that Politics and Society topics are relatively distant with each other. Whereas in the cases of *UK* and *World* datasets belong to unrelated thematic areas with respect to the training dataset (i.e. *Politics*). The *Books* dataset belongs to a different genre from that of the training set, thus, the writing styles are different.

Table 2: *The Guardian* corpus statistics

Dataset	No. of docs	Words Avg.	Words Std.	Chars Avg.	Chars Std.
Politics	112	1,036.86	275.89	6,159.59	1,683.79
Society	62	1,027.84	242.72	6,130.26	1,484.63
World	117	1,067.07	264.94	6,357.15	1,583.36
UK	90	1,096.88	311.86	6,500.04	1,906.85
Books	63	1,136.41	565.63	6,748.56	3,180.52

The results are reported in terms of the accuracy obtained for each pair of topic categories using document embeddings learned on different n-gram types: characters, POS tags, and words (in all cases with n varying from 1 to 5).

In this work, we did not take into account the effect of the overlapping of n-grams over the embeddings (due to sequential construction of n-grams). The idea of non-overlapping of n-grams consists in the following: we consider as the context only the n-grams that already have no overlapping with the given one. This effect is not considered in this work, but we suppose that the overlapping effect is not dominant since the size of the window used to create the embedding is high (more than 5). Nevertheless, it is the interesting direction of future work.

All experiments were conducted using the Python package Scikit-Learn [20]. For training the Doc2vec model, we use the third party library GENSIM[3], which implements the Paragraph Vector algorithm (Doc2vec) [14]. The interface that builds the Doc2vec model requires three parameters: the number of features to be returned (length of the vector), the size of the window that captures the neighborhood, and the minimum frequency of words to be considered in the model.

Since there are no previous works on how to tune the described parameters neither for a specific corpus nor for a specific task, we replicate the steps presented in our previous research [21] to tune these parameters: perform grid search over the fixed ranges for each parameter. The ranges of parameters are the following: number of features in $[50, 350]$, size of window in $[3, 19]$, and minimum frequency in $[3, 4]$.

5 Experimental Results

In this section, we present the obtained results for the authorship attribution task using a cross-topic dataset. The experiments were conducted using

[3] `https://radimrehurek.com/gensim`

documents embeddings trained on n-grams (with n varying between 1 and 5) of characters (Char), POS tags (POS) and words (Words) as features and a logistic regression classifier with default parameters.

Table 3 shows the author classification accuracy when using the *Politics* category for training and the *Society* category for testinging. While Tables 4, 5 and 6 show the accuracy while testing with *UK*, *World* and *Books reviews* categories respectively. The tables also show the results of two methods for obtaining the document embeddings of testing documents: inferred vectors and retrained vectors. The best overall results in each table are highlighted in bold, while the best results for each individual document embeddings method are underlined.

Table 3: Results in terms of accuracy (%) when training document embeddings on n-grams (with n varying between 1 and 5) of characters (Char), POS tags (POS) and words (Words). Considering *Politics* as the training set and *Society* as the testing set

Size of n-gram					**Accuracy with inferred vectors (%)**			**Accuracy with retrained vectors (%)**		
1	**2**	**3**	**4**	**5**	**Char**	**POS**	**Words**	**Char**	**POS**	**Words**
✓					64.52	62.90	95.16	59.68	56.45	91.94
	✓				67.74	79.03	79.03	64.52	77.42	90.32
		✓			66.13	79.03	20.97	62.90	83.87	19.35
			✓		77.42	70.97	14.52	79.03	69.35	12.90
				✓	82.26	35.48	16.13	<u>90.32</u>	41.94	16.13
✓	✓				75.81	82.26	<u>**96.77**</u>	66.13	79.03	<u>**96.77**</u>
✓	✓	✓			83.87	87.10	<u>**96.77**</u>	70.97	<u>87.10</u>	95.16
✓	✓	✓	✓		83.87	90.32	<u>**96.77**</u>	79.03	<u>87.10</u>	95.16
✓	✓	✓	✓	✓	<u>87.10</u>	<u>91.94</u>	<u>**96.77**</u>	87.10	<u>87.10</u>	95.16

Tables 3, 4, 5 and 6 present similar pattern in the results. An interesting observation is that the best results are obtained by appending the document embeddings of different n-gram sizes. For the embeddings trained on character and POS tags n-grams, larger sizes of n achieved the best results. While appending of document embeddings trained on word 1-grams and word 2-grams are in general more effective than higher level word n-grams.

The parameter configuration for the best results obtained with inferred vectors are presented in Table 7. For each experiment the three parameters are shown, the size of the feature vector (size), the size of the window (win) and the minimum frequency (mf). It can be observed that the parameters are more stable when using the combination of document embedding learned on word n-grams (for all train-test pairs). Recall that the best results are obtained with this document embeddings (learned on word n-grams). Specifically, for the cross-topic scenario is can be seen that the optimal vector size is 50 or 100 elements at most, while for the cross-genre scenario the vector size increases to 250 elements. The windows size parameter is found to be optimal between 4 and 7 and the minimum frequency parameter is 2 in the majority of cases.

Table 4: Results in terms of accuracy (%) when training document embeddings on n-grams (with n varying between 1 and 5) of characters (Char), POS tags (POS) and words (Words). Considering *Politics* as the training set and *UK* as the testing set

Size of n-gram					**Accuracy with inferred vectors (%)**			**Accuracy with retrained vectors (%)**		
1	**2**	**3**	**4**	**5**	**Char**	**POS**	**Words**	**Char**	**POS**	**Words**
✓					58.89	73.33	85.56	57.78	64.44	88.89
	✓				63.33	78.89	80.00	63.33	82.22	82.22
		✓			64.44	83.33	21.11	57.78	82.22	23.33
			✓		78.89	72.22	18.89	83.33	72.22	17.78
				✓	73.33	42.22	11.11	92.22	48.89	12.22
✓	✓				74.44	82.22	90.00	66.67	84.44	**93.33**
✓	✓	✓			80.00	88.89	88.89	70.00	88.89	91.11
✓	✓	✓	✓		85.56	92.22	88.89	88.89	87.78	91.11
✓	✓	✓	✓	✓	85.56	88.89	88.89	92.22	85.56	91.11

Table 5: Results in terms of accuracy (%) when training document embeddings on n-grams (with n varying between 1 and 5) of characters (Char), POS tags (POS) and words (Words). Considering *Politics* as the training set and *World* as the testing set

Size of n-gram					**Accuracy with inferred vectors (%)**			**Accuracy with retrained vectors (%)**		
1	**2**	**3**	**4**	**5**	**Char**	**POS**	**Words**	**Char**	**POS**	**Words**
✓					47.86	54.70	82.91	54.70	63.25	84.62
	✓				58.97	78.63	72.65	49.57	82.91	82.91
		✓			53.85	76.92	17.95	49.57	81.20	19.66
			✓		66.67	64.10	12.82	70.09	67.52	11.11
				✓	66.67	34.19	09.40	74.36	41.88	11.11
✓	✓				63.25	77.78	86.32	61.54	82.91	**90.60**
✓	✓	✓			66.67	83.76	83.76	64.96	89.74	**90.60**
✓	✓	✓	✓		74.36	82.05	83.76	77.78	86.32	**90.60**
✓	✓	✓	✓	✓	73.50	84.62	83.76	79.49	86.32	**90.60**

Table 6: Results in terms of accuracy (%) when training document embeddings on n-grams (with n varying between 1 and 5) of characters (Char), POS tags (POS) and words (Words). Considering *Politics* as the training set and *Books* as the testing set

Size of n-gram					**Accuracy with inferred vectors (%)**			**Accuracy with retrained vectors (%)**		
1	**2**	**3**	**4**	**5**	**Char**	**POS**	**Words**	**Char**	**POS**	**Words**
✓					52.38	55.56	82.54	52.38	52.38	76.19
	✓				49.21	79.37	65.08	52.38	76.19	76.19
		✓			46.03	69.84	20.63	55.56	79.37	22.22
			✓		63.49	65.08	15.87	63.49	65.08	20.63
				✓	69.84	34.92	19.05	71.43	42.86	20.63
✓	✓				61.90	77.78	85.71	61.90	73.02	**88.89**
✓	✓	✓			61.90	79.37	85.71	69.84	82.54	80.95
✓	✓	✓	✓		69.84	82.54	85.71	71.43	82.54	80.95
✓	✓	✓	✓	✓	74.60	79.37	85.71	79.37	82.54	80.95

Table 7: Parameter configuration for the best results using the inferred vectors, for all training-testing pairs

Size of n-gram					**Parameter for inferred vectors**								
					Train: Politics, Test: Society								
1	**2**	**3**	**4**	**5**	**Char**			**POS**			**Words**		
					size	**win**	**mf**	**size**	**win**	**mf**	**size**	**win**	**mf**
✓	✓				200	14	2	100	4	2	50	3	2
✓	✓	✓			250	7	5	100	8	10	50	7	2
✓	✓	✓	✓		100	7	5	300	8	10	50	7	2
✓	✓	✓	✓	✓	150	6	4	50	8	2	50	7	2
					Train: Politics, Test: UK								
✓	✓				250	5	2	150	4	5	50	3	2
✓	✓	✓			50	5	4	200	7	3	100	5	2
✓	✓	✓	✓		100	3	3	100	8	3	100	5	2
✓	✓	✓	✓	✓	100	4	3	250	14	5	100	5	2
					Train: Politics, Test: World								
✓	✓				150	8	2	100	3	5	50	4	3
✓	✓	✓			250	3	3	150	4	10	50	4	3
✓	✓	✓	✓		200	3	3	50	4	2	50	4	3
✓	✓	✓	✓	✓	100	5	2	250	4	10	50	4	3
					Train: Politics, Test: Books								
✓	✓				50	3	3	150	3	3	50	5	2
✓	✓	✓			50	3	3	50	3	5	250	5	2
✓	✓	✓	✓		50	3	2	150	3	4	250	5	2
✓	✓	✓	✓	✓	50	3	10	50	3	5	250	5	2

We also evaluated the performance of the different document embeddings methods by learning them individually on various types of n-grams (character, POS tags and words) and using a combined representation appending different document embedding. In order to construct the vectors' appending, we used the document embeddings obtained with the inferred vector method. Different approaches could be applied to obtain a combined representation (adding, averaging, multiplying, to name some), we use appending because this approach let us use the information encoded in each type of n-grams independently and obtain an extended representation of the documents.

The best results are obtained when using the embeddings trained on n-grams of POS tags appended with the embeddings trained on word n-grams, which is consistent with the individual behavior of each document embedding independently. The results of these experiments are presented in Table 8. It is worth mentioning that for the experiments presented in this table, we only used the inferred vector for obtaining the test document vectors and also the combination of 1 to 5 character n-grams gave relatively good results, so we just considered it. Note that the results of the appending of the embeddings trained on character n-grams with the other two showed lower accuracy for the AA task.

The parameter configuration for the best results obtained with inferred vectors when appending document embeddings trained on different features are presented in Table 9. The values presented in this table correspond to triplets

Table 8: Results in terms of accuracy (%) appending the document embeddings trained on n-grams (with n varying between 1 and 5) of characters (Char), POS tags (POS) and words (Words). Considering *Politics* as the training set and and the rest of the categories as the testing sets.

Features			**Accuracy with inferred vectors (%)**			
Char	**POS**	**Words**	**Society**	**UK**	**World**	**Books**
-	1,2	1,2	**96.77**	93.33	88.89	88.89
-	1,2,3	1,2,3	95.16	93.33	**91.45**	88.89
-	1,2,3,4	1,2,3,4	95.16	93.33	**91.45**	**90.48**
-	1,2,3,4,5	1,2,3,4,5	**96.77**	**94.44**	89.74	**90.48**
1,2,3,4,5	1,2,3	1,2,3	90.32	92.22	83.76	82.54
1,2,3,4,5	1,2,3,4	1,2,3	90.32	92.22	83.76	87.30
1,2,3,4,5	1,2,3,4,5	1,2,3	91.94	92.22	83.76	85.71

representing the vector size, the windows size and the minimum frequency of a term. We can highlight some patterns in this table, for example when combining document embedding learned on character, POS and word n-grams the vector size tends to decrease, being 50 the optimal size of the vector when using all feature sets. With respect to the testing sets, it can be observed that when testing with *Society* and *Books*, the majority of the experiments yield optimal results with a vectors size of 50. When testing with *World* dataset the best results are obtained with a vector size of 100. For the *UK* dataset, different vector sizes yield optimal results for different features, but we believe with a vector size of 50 is enough for all configurations. With respect to the windows size and minimum frequency parameters it can me observed that a windows size of 4 and minimum frequency of 2 in general achieve the best results in each configuration.

Table 9: Parameter configuration for the best results using the inferred vectors, for all train-test pairs when appending document embeddings trained on different features

Features			**Parameters for inferred vectors**			
Char	**POS**	**Words**	**Society**	**UK**	**World**	**Books**
-	1,2	1,2	50,4,2	50,4,2	100,4,2	200,4,2
-	1,2,3	1,2,3	50,4,2	100,4,2	100,4,2	200,4,2
-	1,2,3,4	1,2,3,4	50,4,2	150,8,3	100,4,2	50,3,5
-	1,2,3,4,5	1,2,3,4,5	50,5,2	250,8,3	100,4,2	50,3,5
1,2,3,4,5	1,2,3	1,2,3	50,5,2	50,5,2	50,3,5	50,4,5
1,2,3,4,5	1,2,3,4	1,2,3	100,3,2	100,3,2	50,3,5	50,4,5
1,2,3,4,5	1,2,3,4,5	1,2,3	50,4,2	50,4,2	50,3,5	50,4,5

We used the results of the algorithm by Stamatatos [31] as baseline for the proposed method. Table 10 presents the final comparison of the best doc-

Table 10: Comparison of the best results in terms of accuracy (%) of the different documents embeddings methods. Considering *Politics* as the training and the rest of the categories as the testing sets

Method	**Society**	**UK**	**World**	**Books reviews**	**Average Acc.**
Single feature (retrained)	96.77	93.33	90.60	88.89	92.40
Single feature (inferred)	96.77	92.22	86.32	85.71	90.26
Concatenating (inferred)	**96.77**	**94.44**	**91.45**	**90.48**	**93.29**
Char 3-grams (from [31])	$\simeq$ 91.00	$\simeq$ 88.00	$\simeq$ 82.50	$\simeq$ 79.50	$\simeq$ 85.50

ument embedding methods on both individual features and the appending of different document embeddings for the authorship attribution task. The appending method yields the best results for all testing sets, except for the *Society* set, when embeddings trained on single features achieved equal results. The proposed method also outperforms the baseline when using different genre of texts (newspaper articles and book reviews). We can observe that the document embeddings-based method is more robust under both cross-topic and the cross-genre scenario. We observed a difference of 6.29% between the best performance (testing with the *Society* dataset) and the worst performance (testing with the *Books* dataset). Whereas, with the character 3-grams based method this difference is 11.50%, validating our previous claim. It is worth noting that the same pattern is maintain with both methodologies (the n-gram baseline and Doc2vec), the ranking of accuracies is maintained in the same testing sets.

Table 11 presents the accuracy when fixing the parameters with the following values ($size = 50$,$win = 4$,$mf = 2$) for all the possible train-test pairs. The fixed parameters where selected from an analysis of the results presented in table 9. It can be observed that the average accuracy when training with *UK* and *World* is maintained, however, when training with *Society* and *Books* the average accuracy drops significantly. The average accuracy of our best model (POS 1,2,3-grams and Word 1,2,3-grams) was calculated, achieving 69.83%. Finally, we compare our results with previous research by *Sapkota et al.* [25], when type character n-grams features are evaluated for cross-topic authorship attribution. We achieved an improvement in accuracy of 12.83% with respect this previous work, where the authors reported 57% of accuracy on the *The Guardian* corpus (averaging all train-test pairs). From this results we conclude that the parameters settings for the Doc2vec model can be learn from a given experimental scenario, maintaining state-of-the-art results on different experimental scenario.

6 Conclusions

We proposed a new method for learning document embeddings from texts, which uses n-grams of various types and sizes as features instead of words.

Table 11: Results in terms of accuracy (%) appending the document embeddings trained on n-grams (with n varying between 1 and 5) of characters (Char), POS tags (POS) and words (Words). Considering *Politics* as the training set and and the rest of the categories as the testing sets.Parameter configuration set as $size$ = 50,win = 4, mf = 2

Features		**Accuracy with inferred vectors (%)**				
		Train: Politics				**Average**
POS	**Words**	**Society**	**UK**	**World**	**Books**	**Acc.**
1,2	1,2	**93.54**	87.77	85.47	**82.53**	87.32
1,2,3	1,2,3	88.70	87.77	82.90	73.01	83.09
1,2,3,4	1,2,3,4	88.70	87.77	**86.32**	79.36	85.53
1,2,3,4,5	1,2,3,4,5	**93.54**	**88.88**	**86.32**	77.77	86.62
		Train: Society				**Average**
POS	**Words**	**Politics**	**UK**	**World**	**Books**	**Acc.**
1,2	1,2	66.96	**62.22**	**64.10**	38.09	57.84
1,2,3	1,2,3	**70.53**	61.11	63.24	**41.26**	59.03
1,2,3,4	1,2,3,4	64.28	60.00	61.53	36.50	55.57
1,2,3,4,5	1,2,3,4,5	63.39	61.11	62.39	36.50	55.84
		Train: UK				**Average**
POS	**Words**	**Politics**	**Society**	**World**	**Books**	**Acc.**
1,2	1,2	76.78	90.32	77.77	**69.84**	78.67
1,2,3	1,2,3	**81.25**	90.32	**82.05**	58.73	78.08
1,2,3,4	1,2,3,4	78.57	**91.93**	77.77	61.92	77.54
1,2,3,4,5	1,2,3,4,5	77.67	**91.93**	76.92	65.07	77.89
		Train: World				**Average**
POS	**Words**	**Politics**	**Society**	**UK**	**Books**	**Acc.**
1,2	1,2	75.89	83.87	76.66	68.25	76.16
1,2,3	1,2,3	**83.03**	**85.48**	**78.88**	**74.60**	80.49
1,2,3,4	1,2,3,4	**83.03**	83.87	**78.88**	73.01	79.69
1,2,3,4,5	1,2,3,4,5	82.14	83.87	**78.88**	69.84	78.68
		Train: Books				**Average**
POS	**Words**	**Politics**	**Society**	**UK**	**World**	**Acc.**
1,2	1,2	**46.42**	**46.77**	**55.55**	**47.00**	48.93
1,2,3	1,2,3	**46.42**	**46.77**	54.44	46.15	48.44
1,2,3,4	1,2,3,4	41.96	40.32	50.00	42.73	43.75
1,2,3,4,5	1,2,3,4,5	42.85	33.87	48.88	40.17	41.44

It is based on the well-known Paragraph Vector algorithm (Doc2vec). Our model uses Doc2vec for learning document vectors and n-gram vectors (character, POS tag and word) by predicting both document and n-gram features. We tested the method on the task of authorship attribution under cross-topic settings. The inclusion of n-grams features for learning the document embeddings allows us take advantage of the efficiency of the Doc2vec model to learn semantic, syntactic and grammatical patterns of an author that are hidden in the documents.

Experimental results show that the proposed model outperforms the traditional Doc2vec embeddings trained on word unigrams (words). Furthermore, comparing with a traditional character n-gram model, our Doc2vec model based on n-grams features captures information that is important for the authorship attribution task. This capability is observed in the obtained results, when the character-based models are outperformed by 7% in average.

The experiments demonstrate that the use of POS n-grams as input type for document embeddings produces comparable results with the use of character or word n-grams' embeddings. Still, the best results are obtained when combining them with word n-grams embeddings. Note that in the state-of-the-art POS n-grams do not report good results.

In future, we plan to conduct experiments with document embeddings learned on other features such as syntactic n-grams [28, 22, 23] and typed n-grams [25, 18]. It would also be interesting to evaluate different composition methods for the document embeddings (deep averaging methods, convolutional neural networks and recurrent neural networks) learned on various n-gram types.

Acknowledgements This work was done under partial support of the Mexican Government (CONACYT PROJECT 240844, SNI, COFAA - IPN, SIP - IPN 20162204, 20151406, 20144274).

References

1. Abbasi, A., Chen, H.: Applying authorship analysis to extremist-group web forum messages. IEEE Intelligent Systems **20**(5), 67–75 (2005)
2. Bengio, Y., Ducharme, R., Vincent, P., Jauvin, C.: A neural probabilistic language model. Journal of machine learning research **3**(Feb), 1137–1155 (2003)
3. Black, P.E.: Fisher-yates shuffle. Dictionary of algorithms and data structures **19** (2005)
4. Coulthard, M.: On admissible linguistic evidence. Journal of Law & Policy **21**, 441 (2012)
5. Escalante, H.J., Solorio, T., Montes-y Gómez, M.: Local histograms of character n-grams for authorship attribution. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1, ACL '11, pp. 288–298 (2011)
6. Gómez-Adorno, H., Sidorov, G., Pinto, D., Markov, I.: A graph based authorship identification approach. In: Working Notes Papers of the CLEF 2015 Evaluation Labs, *CLEF '15*, vol. 1391 (2015)
7. Gómez-Adorno, H., Sidorov, G., Pinto, D., Vilariño, D., Gelbukh, A.: Automatic authorship detection using textual patterns extracted from integrated syntactic graphs. Sensors **16**(9), 1374 (2016)
8. Iyyer, M., Manjunatha, V., Boyd-Graber, J.L., Daumé III, H.: Deep unordered composition rivals syntactic methods for text classification. In: Association for Computational Linguistics, ACl '15, pp. 1681–1691 (2015)
9. Kalchbrenner, N., Grefenstette, E., Blunsom, P.: A convolutional neural network for modelling sentences. arXiv preprint arXiv:1404.2188 (2014)
10. Kestemont, M., Luyckx, K., Daelemans, W., Crombez, T.: Cross-genre authorship verification using unmasking. English Studies **93**(3), 340–356 (2012)
11. Kiros, R., Zhu, Y., Salakhutdinov, R.R., Zemel, R., Urtasun, R., Torralba, A., Fidler, S.: Skip-thought vectors. In: Advances in neural information processing systems, NIPS '15, pp. 3294–3302 (2015)
12. Koppel, M., Schler, J., Bonchek-Dokow, E.: Measuring differentiability: Unmasking pseudonymous authors. Journal of Machine Learning Research **8**(Jun), 1261–1276 (2007)

13. Koppel, M., Seidman, S.: Automatically identifying pseudepigraphic texts. In: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP '13, pp. 1449–1454 (2013)
14. Le, Q.V., Mikolov, T.: Distributed representations of sentences and documents. In: Proceedings of the 31th International Conference on Machine Learning, ICML '14, pp. 1188–1196 (2014)
15. Li, B., Liu, T., Du, X., Zhang, D., Zhao, Z.: Learning document embeddings by predicting n-grams for sentiment classification of long movie reviews. arXiv preprint arXiv:1512.08183 (2015)
16. Maas, A.L., Daly, R.E., Pham, P.T., Huang, D., Ng, A.Y., Potts, C.: Learning word vectors for sentiment analysis. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1, ACl '11, pp. 142–150 (2011)
17. Madigan, D., Genkin, A., Lewis, D.D., Fradkin, D.: Bayesian multinomial logistic regression for author identification. In: AIP Conference Proceedings, vol. 803, pp. 509–516. AIP (2005)
18. Markov, I., Stamatatos, E., Sidorov, G.: Improving cross-topic authorship attribution: The role of pre-processing. In: 18th International Conference on Computational Linguistics and Intelligent Text Processing, CICLING '17 (2017)
19. Mikolov, T., tau Yih, W., Zweig, G.: Linguistic regularities in continuous space word representations. In: Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL '13, pp. 746–751 (2013)
20. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al.: Scikit-learn: Machine learning in python. Journal of Machine Learning Research **12**(Oct), 2825–2830 (2011)
21. Posadas-Durán, J.P., Gómez-Adorno, H., Sidorov, G., Batyrshin, I., Pinto, D., Chanona-Hernández, L.: Application of the distributed document representation in the authorship attribution task for small corpora. Soft Computing pp. 1–13
22. Posadas-Duran, J.P., Sidorov, G., Batyrshin, I.: Complete syntactic n-grams as style markers for authorship attribution. In: Mexican International Conference on Artificial Intelligence, MICAI '14, pp. 9–17 (2014)
23. Posadas-Durán, J.P., Sidorov, G., Batyrshin, I., Mirasol-Meléndez, E.: Author verification using syntactic n-grams. In: Working Notes Papers of the CLEF 2015 Evaluation Labs, *CLEF '15*, vol. 1391 (2015)
24. Potthast, M., Braun, S., Buz, T., Duffhauss, F., Friedrich, F., Gülzow, J.M., Köhler, J., Lötzsch, W., Müller, F., Müller, M.E., Paßmann, R., Reinke, B., Rettenmeier, L., Rometsch, T., Sommer, T., Träger, M., Wilhelm, S., Stein, B., Stamatatos, E., Hagen, M.: Who wrote the web? revisiting influential author identification research applicable to information retrieval. In: Advances in Information Retrieval - 38th European Conference on IR Research, ECIR '16, pp. 393–407 (2016)
25. Sapkota, U., Bethard, S., Montes-y Gómez, M., Solorio, T.: Not all character n-grams are created equal: A study in authorship attribution. In: Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL '2015, pp. 93–102 (2015)
26. Sapkota, U., Solorio, T., Montes-y Gómez, M., Bethard, S., Rosso, P.: Cross-topic authorship attribution: Will out-of-topic data help? In: The 25th International Conference on Computational Linguistics: Technical Papers, COLING '14, pp. 1228–1237 (2014)
27. Schwartz, M.B.: An examination of cross-domain authorship attribution techniques (2016)
28. Sidorov, G., Velasquez, F., Stamatatos, E., Gelbukh, A., Chanona-Hernández, L.: Syntactic n-grams as machine learning features for natural language processing. Expert Systems with Applications **41**(3), 853–860 (2014)
29. Socher, R., Perelygin, A., Wu, J.Y., Chuang, J., Manning, C.D., Ng, A.Y., Potts, C.: Recursive deep models for semantic compositionality over a sentiment treebank. In: Proceedings of the 2013 conference on empirical methods in natural language processing, EMNLP '13, pp. 1631–1642 (2013)
30. Stamatatos, E.: A survey of modern authorship attribution methods. Journal of the American Society for Information Science and Technology **60**(3), 538–556 (2009)

31. Stamatatos, E.: On the robustness of authorship attribution based on character n-gram features. Journal of Law and Policy **21**(2) (2013)