Feature Bagging for Author Attribution

François-Marie Giraud and Thierry Artières

LIP6, Université Pierre et Marie Curie (UPMC), Paris, France
giraudf@poleia.lip6.fr, thierry.artieres@lip6.fr

Abstract The authorship attribution literature demonstrates the difficulty to design classifiers overcoming simple strategies such as linear classifiers operating on a number, most frequent, of lexical features such as character trigrams. We claim this comes, at least partially, from the difficulty to efficiently learn the contribution of all features, which leads to either undertraining or overtraining of classifiers. To overcome this difficulty we propose to use bagging techniques that rely on learning classifiers on different random subset of features, then to combine their decision by making them vote.

1 Introduction

A key issue in author attribution and verification lies in feature definition and selection, which motivated many studies [14], [8]. One conclusion is that despite many efforts to build smart features [6] very simple ones such as counts (or tfidf like features) of words and/or of character n-grams are commonly used. Moreover feature selection is performed using simple criterion such as choosing the most frequent words and character ngrams. Finally, simple classifiers such as linear SVM have been shown to perform well with above features and such simple systems appear to be difficult to outperform [8]. Our work is an attempt to outperform such a simple, and efficient, strategy. It is inspired from two key observations that have been made in the past.

First, it has been observed that learning rich models on few training data may yield a form of undertraining [15] where some relevant features are not fully taken into account by the model after training. This may happen when a number of features (not necessarily many) are sufficient, alone, for perfect discrimination of the training samples. In that case learning may focus on learning good weights for few of these relevant features while neglecting remaining relevant features. Then if only the neglected discriminative features occur in a test sample it will be misclassified. This has been observed in particular in the context of text processing with log linear models where one usually exploits a huge number of features and where training samples are often linearly separable with a small subset of the features.

Second, the work by [10] suggests that an author's witting style is characterized by a limited number of discriminative features and more importantly by the way the classifier performance behaves (i.e. accuracy drops) when most important features (e.g. having large weights after SVM learning) are iteratively removed.

We investigate here new methods that take into account the two above results to design efficient classifiers for authorship attribution. They both rely on bagging ideas where one combines the results of a number of classifiers that are learned on training samples represented with a random subset of features.

We first draw a panorama of related works in section 2 then we provide in section 3 details on the datasets that we used in this paper in addition to the PAN 2012 challenge datasets. Next we introduce our general idea and investigate the potential interest of feature bagging in section 4 and we present our approach in section 5.

2 Related works

2.1 Features

Designing good features is a key issue for author identification, many features have been investigated up to now. These may be grouped in a few categories; lexical features, syntactic features, structural features, and contextual features. We briefly review all these now.

Lexical features

- TF-IDF (term frequency - inverse document frequency): Tf-idf are standard features used in text processing, information retrieval, that consist in counting words' occurrences and weighting these counts by words' document frequency to decrease the influence of frequent and uninformative (with respect to the topic of the text) words [11]. Using Tf-idf yields representing a document in a very high dimensional space (there are one feature per word in the vocabulary). One may reduce the dimension of the feature space by selecting features using various measures such as information gain [8].
- Word length: Statistics on word length has been used in e.g. [4]. It is a simple and easy to compute feature but it has a low discriminative power.
- Sentence length.
- Richness of the vocabulary: This may be computed as the number of different words used by an author. Again it is a simple and easy to compute feature but with a low discriminative power.
- Word N-grams: These features are counts of the number of occurrences of N successive words. One only considers unigrams (N=1), bigrams (N=2) and trigrams (N=3). One can use simple counting or Tf-idf like scores [5]. Of course one cannot consider all N-grams which are much too numerous, and one has to select a priori the most useful ones.
- Character N-grams: These features are similar to previous ones but we are interested in tuples of characters not word [7]. Interestingly these features have been shown to be efficient in a number of tasks on text data.
- CW: This is a short name for TF-IDF features computed for the 1 000 words with highest information gain (after [8])
- CNG: This is a short name for TF-IDF features computed for the 1 000 trigrams (of words) with highest information gain (after [8])

Syntactic features

- Linking words: Counting features (simple counts or TF-IDF like normalized counts) for particular words: conjunction, preposition, pronoun, modal verbs, ...
- Part Of Speech (POS): Counting features (simple counts or TF-IDF like normalized counts) on a tagged representation of the text; nouns, adjectives, verbs, singular, plural, ... [12]
- POS N-grams: N-gram on POS tags [2]

Structural features

- Font size [1]
- Font color [1]
- Number of images in the document [1]
- Number of hyper links [1]

Contextual features

- Topic(s) of the document [1]
- Elongation and inflexion of Arabic words [1]

The difficulty to find good features for author identification lies in that author signature is embedded in many other information in the text that concern the topic, the opinion, etc. As far as we can imagine from our own way to guess the author of a text we focus on very particular construction, the use of particular words etc, in other words we look at any unusual difference with a *mean* way of writing. Also it is more likely that the most discriminative features for one author are very dependent on the author and cannot be guessed a priori.

Then, one most often uses an eventually large number of features and let the classifier decide which ones are useful or not for the targeted author identification task.

2.2 State of the art

Few classification methods have been investigated to operate on documents representated by a subset of features taken from the list above. Mainly two families of methods have been used: methods coming from the information retrieval field (dot product or any similarity measure on vectorial representations of documents), and methods from the statistical machine learning field such as Support Vector Machines (SVM). Table 1 compares few results from the literature in terms of corpus, of classification method, and of the features used to represent a document.

We want to comment a little on this table and on the studies from which these results are taken.

First of all, the work in [16] on a French corpus focused on measuring the relevance of using kernelized SVM instead of simpler linear ones. Although they showed improved accuracy it is an isolated work. The literature shows on the contrary that linear SVM are popular and efficient in the author identification field. These models are powerful enough when used with a large number of features, as it as been demonstrated

Language	# authors	Context	Features	Classifier	Error rate	Ref
French	28	Literature (≈ 4 books/author)	Character N-grams	KSVM	12,30 %	[16]
English	2	Email (240 / author)	CW+CNG	LSVM	21,80%	[8]
English	9	Literature (2 books/author)	CW+CNG	LSVM	13,70%	[8]
English	20	Blogs (≈ 400 posts/author)	CW+CNG	LSVM	14,30%	[8]
English	8000	Blogs	TF-IDF on words	Dot Product	72,00%	[9]
English	5	Post in forums (20 messages/author)	87 Lex	SVM	12,00%	[1]
English	5	Idem	301 (Lex, Syt, Str, Cnt)	SVM	3,00%	[1]
Arabic	5	idem	79 Lex	SVM	12,30%	[1]
Arabic	5	Idem	418 (Lex, Syt, Str, Cnt)	SVM	5,20%	[1]

Table 1. Comparison of literature results. LSVM stands for Linear SVM and KSVM stands for Kernel SVM (i.e. nonlinear). Lex, Syt, Str and Cnt stand respectively for Lexical features, syntactic features, structural features and contextual features.

in many text classification tasks, they often allow perfect classification on the training set.

[8] compared the efficiency of various feature sets with different classification methods and showed that best results were achieved with SVM working with CW and CNG features on a few datasets. Besides, the studies of extremist posts (KKK for English and Palestinian and Al-Aqsa Martyrs group for Arabic) concerned 5 authors only but the study demonstrated the usefulness of structural and contextual features in this particular context [1]. A conclusion of studies on the discriminative power of various features vary and it appears difficult to determine definitely a set of discriminative features.

At the end, few works aimed at designing new classification methods dedicated to author identification. Linear SVMs appear to be a good compromise, and the key issue is rather to determine which are the moste useful features, this appears as the main question to get good results.

3 Datasets and experimental settings

3.1 Datasets

We report experimental results gained on the PAN 2012 challenge datasets and on two additional datasets on which we have been able to perform numerous experiments in order to characterize the behaviour of our method. We provide here details on the two additional datasets, details one the PAN 2012 challenge may be found on the challenge website.

The first additional dataset that we use is an english literature corpus used in some previous publications ([8], [10]). It is very similar to the corpus used in the PAN 2012 challenge: There are 9 authors and 2 complete books per author. There are an average of 100 thousands words by book and every book was divided manually in about a hundred documents, keeping integrity of chapters and of text sections. A large majority of the documents are about 500 to 3000 words length.

The second additional corpus is a subset of a corpus of blogs with about 18 000 authors [9]. We worked on a subset of this corpus considering only the 60 main authors (bloggers), i.e. those who post frequently (at least 20 posts) posts that are longer than 100 words.

3.2 Experimental settings

In all reported experiments we used linear support vector machines (SVMs) as classifiers since they have been shown to provide state of the art results in many text processing and classification tasks and in particular for author authentification.

We used the LibSVM library [3] where a multiclass classifier for a N class classification problem is implemented through the learning of $N \times (N-1)/2$ one-to-one binary SVMs. All classifiers are learned with a standard L2 regularization term (to avoid overfitting) whose weight is set on the validation dataset.

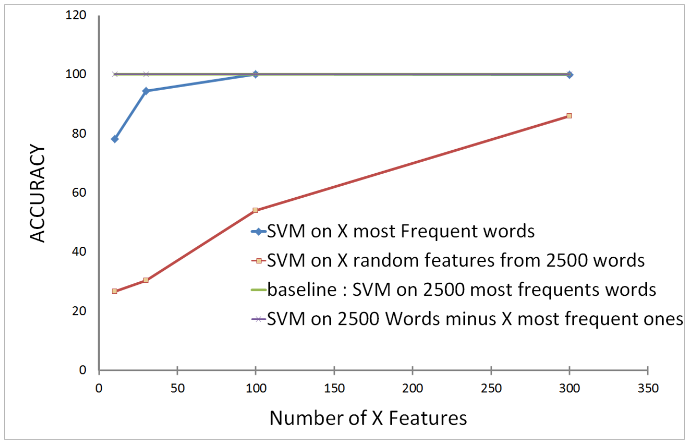
Note also that we exploited the probabilistic variant of SVMs as implemented by LibSVM. When the outputs of multiple SVMs are to be compared in order to take a decision (see section 6) we naturally take the decision corresponding to the SVM with the biggest output.

4 Unexploited features and classifier undertraining

We hypothesize that undertraining as described in [15] often occurs in authorship attribution tasks. Actually we observed on many datasets that SVMs working on many lexical features (word or character trigrams counts or tf-idf) easily reach 100% accuracy on the training set while performance on the test set may be significantly below, which is symptomatic of an overtraining problem. Yet, as suggested by [15] it may be more accurately understood as an undertraining problem when considering linear models exploiting a large number of features.

Indeed when the classifier is rich enough (e.g. a linear classifier exploiting a number of discriminative features) it may happen that some relevant features are not fully taken into account by the learned model. This may happen when a number of features (not necessarily many) are sufficient, alone, for perfect discrimination of the training samples. Then training may focus on exploiting some of the relevant features allowing perfect classification on the training set, while ignoring some other relevant features. In such a case, if a test sample includes neglected discriminative features only it will be misclassified. It is a form of undertraining that may occur when training samples are linearly separable with a small subset of the features, e.g. when having many features and/or few training data.

Figure 1 reports preliminary experimental results that yield thinking there is actually some kind of undertraining in SVMs learned on authorship attribution tasks with many simple (lexical) features. It plots the accuracy of a linear SVM (Support Vector Machine) exploiting a limited number of features, X, ranging from 10 to 350 chosen from a set of 2 500 features (2 500 most frequent character trigrams) as a function of X. Plots are for the training dataset (top) and for the test set (bottom). Both plots provide two curves corresponding to choosing the X features at random or by selecting the X



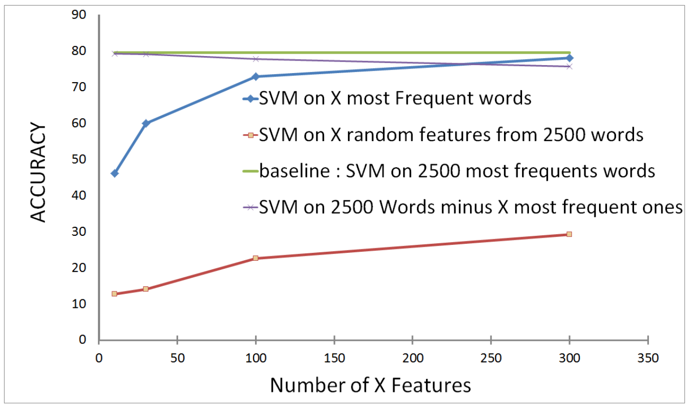


Figure 1. Accuracy (on a 9 authors dataset [8]) on the training set (top) and on the test set (bottom) of linear SVMs when selecting a subset of X features at random or by selecting the X most frequent features from a set of 2 500 features. Curves represent accuracy plotted as a function of X for $X \in [10, 300]$. Accuracy of a SVM using all 2 500 features and of a SVM exploiting all 2 500 features minus the X most frequent features are given for comparison.

most frequent features, a line which corresponds to a linear SVM exploiting all 2 500 features, and an additional curve for the accuracy of a SVM using all 2 500 features minus the X most frequent features.

These figures put in evidence some interesting facts. As may be seen the performance of classifiers exploiting only few features is very high on the training set when using the most frequent features, quickly reaching 100% perfect classification, (which is also true when using all features) while it reaches a plateau on the test set at about 80% accuracy. There is then a strong gap between the performance on the training set and on the test set which show an overtraining problem. Also the accuracy of a SVM exploiting all 2 500 features minus the most frequent ones is very high both on the training set and on the test set, which shows these features contain discriminative information too.

Indeed the figures also show that using few random features also allow to discriminate between authors up to a certain extent, which means all features (including less frequent ones) contain some discriminative information. It is likely that the learning of a SVM will focus on exploiting most frequent features so that at the end one may expect that SVMs will not necessarily fully exploit all discriminative features since only few of them (most frequent) already allow reaching 100% accuracy on the training set. As a consequence there exist a number of discriminative features that are neglected by during learning and that might improve generalization.

5 Bagging features for improved author authentification

Based on the discussion above we aim at designing approaches able to fully exploit the potential of all available features. We investigated methods relying on bagging features, i.e. learning many classifiers on different subsets of the features then combining their predictions.

5.1 Principle

Many methods have been proposed for combining classifiers such as co-training, boosting, bagging, a number of which have been designed or adapted for working with classifier exploiting different subsets of features [17], [15]. In particular, feature bagging has been investigated by a few researchers in the past [13], [15]. Viola & jones [17] used boosting with extremely weak classifiers (learned on a single feature each) every iteration. [13] also used boosting with an adaptation of AdaBoost to feature weighting instead of samples weighting as in AdaBoost.

In this preliminary study we decided to investigate a standard bagging combination where an eventually large number of base classifiers that are learned on random subsets of the features (with eventual overlap) and that are combined at test time thourgh a voting procedure. In practice we investigated using a majority vote decision process with a number of SVM classifier trained on many (hundreds to thousands) random subsets of few (tens to hundreds) features. SVM classifier are learned with libsvm toolbox(see section 3).

5.2 Experimental results

Preliminary experiments Preliminary results were obtained on the 60 bloggers corpus.

All the results presented in this section have been obtained on a single learning/validation/test split to limit computational complexity. All the models are trained on the learning set and the validation test is used to determine the optimal value of the SVM regularization parameter.

First we investigate the influence of the number of random features K exploited by the base classifiers and of the number of base classifiers M. Figure shows the evolution ot the system's accuracy as a function of the number of base models. There are few curves corresponding to a different number of random features used by a base classifier. The features used are chosen from a set of 3 000 most frequent character trigrams. As may be seen the value of K influences the performance of the overall approach and it seems better to use a small value here K, probably yielding more variability between all base classifiers. Besides, it looks like the more there are base classifiers the higher the accuracy, in particular when designing base classifiers working on a small number of random features.

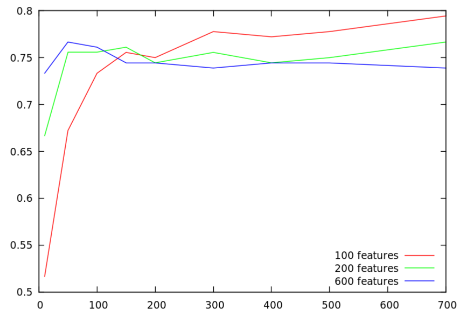


Figure 2. Performance of the bagging approach as a function of the number of models. The three curves stand for the number of random features that base classifiers exploit.

Table 2 provides some interesting statistics on the base classifiers, the mean accuracy, the minimum accuracy (among all base classifiers) and the maximum accuracy on the training, the validation and the test datasets. It shows in particular that the more features the base classifier exploit the higher is the average accuracy, but it shows also that a single base classifier is not a good performer alone.

Table 3 compares the performance of the bagging feature approach with that of a single SVM working witl all features. One may see here that whatever the number of random features used by base classifiers, the bagging approach systematically outperforms a SVM exploiting all the features. This justifies a posteriori our discussion on the undertraining phenomenon of classifiers in the context of author identification.

These results show a significative improvement of the bagging feature approach over a single SVM classifier exploiting all the features.

Table 2. Statistics on base classifiers.

	Minimum			Mean			Maximum		
# features	Train	Valid	Test	Train	Valid	Test	Train	Valid	Test
100	99.8	33.2	32.2	99.8	45.6	42.7	100	56.1	53.3
225	99.8	50	46.1	99.9	60.5	55.8	100	69.4	64.4
600	99.8	55	48.9	99.9	65.5	60.1	100	75.5	67.8

Table 3. Performance of the Bagging feature approach.

Model	Train	Valid	Test
Bagging (100 features)	99.9	82.2	79.4
Bagging (225 features)	100	83.9	76.7
Bagging (600 features)	100	83.9	76.1
Single SVM with all 3000 features	100	79.4	71.6

PAN'12 challenge For the PAN challenge dataset, we learned models on a number of splits of the provided training corpus into pairs of learning/validation datasets. For each of these S training/validation splits, we learned M models based on K randomly selected features from the initial set of features (word or caracter trigram counts) (see Table 4). We finally take decision over $(S \times M)$ models learned each with K random selected features in the T initial features. For closed problem, we simply used a majoritary vote to design prediction on test data. For open problems we use same models and vote method but fixed a threshold on each author based on validation results below wich we consider that none of the condidates is the real author. The table below summarize by task ours submitted methods. In the table the initial set of features is described by the feature type (i.e word-count or character-n-gram count) and by the number T of most frequent features keeped from the training set. Accuracy of our approach for a variety of hyperparameters (M, K, etc) values are given in the table.

6 Two stage approach

To go further we built on the work of [10] who suggested that an author's witting style is characterized by a limited number of discriminative features and more importantly by the way the classifier performance behaves (i.e. drops) when most important features (having large weights after learning) are iteratively removed. We designed a method that is inspired by this work and exploits weak SVM classifiers learned on random subsets of features.

Table 4. Accuracy of the feature bagging approach on PAN 2012 datasets.

TASK	Run Name	K (# splits)	N (# Models / split)	# Models overal	Type of feature	# Random features	Open/Closed task	Accurracy
A	Lip6 1	8	100	800	WORDS-1500	200	open	100
B	Lip6 1	8	100	800	WORDS-1500	200	closed	70
A	Lip6 2	8	1	8	3CHAR-3500	3500	open	100
B	Lip6 2	8	1	8	3CHAR-3500	3500	closed	60
A	Lip6 3	8	100	800	WORDS-1500	300	open	100
B	Lip6 3	8	100	800	WORDS-1500	300	closed	70
C	Lip6 1	10	100	1000	WORDS-1500	400	open	100
D	Lip6 1	10	100	1000	WORDS-1500	400	closed	41.18
C	Lip6 2	10	300	3000	3CHAR-3500	1000	open	75
D	Lip6 2	10	300	3000	3CHAR-3500	1000	closed	52.94
C	Lip6 3	10	300	3000	3CHAR-3500	1250	open	62.5
D	Lip6 3	10	300	3000	3CHAR-3500	1250	closed	35.29
I	Lip6 1	12	1	12	WORDS-1500	1500	open	85.71
J	Lip6 1	12	1	12	WORDS-1500	1500	closed	81.25
I	Lip6 2	12	1	12	WORDS-2000	2000	open	78.57
J	Lip6 2	12	1	12	WORDS-2000	2000	closed	68.75
I	Lip6 3	12	1	12	WORDS-2500	2500	open	78.57
J	Lip6 3	12	1	12	WORDS-2500	2500	closed	75

We explain now the principle of this approach. Consider an author classification problem with N authors (classes) and where documents are represented by p-dimensional feature vectors. The system we propose is a two stage system.

In a first stage we learn N linear multiclass SVMs exploiting random subsets (of size X) of the p original features as before in section 5. We note $S_i \subset [1, d]$ the set of indices of features used by the i^{th} SVM and SVM_i the i^{th} classifier. These classifiers are learned to affect a document to one of the N authors.

Then we use the classifiers of the first stage to build new vectors (that we call profiles) that will be processed in a second stage by another classifier. We start by describing how the first stage is used to build a new training dataset which we call the *second stage* dataset. For any author $a \in [1, N]$ and for any document d, we build a new p-dimensional feature vector (a profile) whose j^{th} component is the proportion of classifiers (among the N classifiers that exploit feature j) that predict author a. More formally the profile for a particular pair (document,author) which we note $u(d, a)$ is a vector whose components are defined as:

$$u_j(d, a) = \frac{1}{Z(j)} \sum_{i=1:K} \delta(j \in S_i) \times \delta(SVM_i(d) == a) \qquad (1)$$

where $SVM_i(d)$ stands for the output (a class number in [1..N]) of the i^{th} SVM for document d, where $\delta(P)$ equals one if predicate P is true and 0 otherwise, and where $Z(j)$ is a normalization factor $Z = \sum_{i=1:K} \delta(j \in S_i)\times$. At the end $u_j(d, a)$ stands for the percentage of classifiers, among those that exploit the j^{th} feature, that predict author a.

We then use such profiles as an input to a second stage classification system. Those profiles may be sorted (from the highest value to the lowest) so that the numbering of the components are lost. Figure 3 shows such profiles for the 60 authors of the blog dataset. As suggested by [10] the rate the performance decreases may be relevant of a particular author.

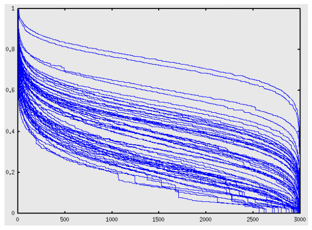


Figure 3. Example of profile vectors on the blog dataset (one curve per author).

At this point, each document in the train and the validation corpus correspond to N profiles, one for every author. Based on this new dataset we learn a new classifier to discriminate between positive examples (profiles that are built for the actual author) and negative examples. The classifier is a prototype based method where one author (ine class) is represented as the mean vector profile of this class computed on the training set. At test time one computes a similarity between a test profile and the reference profile for every class. We investigated euclidean distance and correlation similarity. Table

When classifying a test document we first run the set of K SVM classifiers of the first stage, tehn we build N p-dimensional profiles as above. We take the final decision based on the second stage classifier that is run on these N *second stage* feature vectors (looking for the highest similatity, lowest distance).

Table 5 shows that this second approach also significantly outperforms the single SVM approach and it allows reaching similar results as the standard bagging approach while working on a very different representation of the documents. This let some hope that the two methods could be advantageously combined which we did not investigate by lack of time.

Table 5. Accuracy of the two stage approach.

Similarity measure	Samples	Accuracy Valid	Accuracy Test
Euclidean distance	Raw vectors	82.2	79.4
Correlation	Raw Profiles	80.2	78.9
Correlation	Sorted profiles	82.2	79.4

7 Conclusion

We presented an experimental investigation that show that one of the most competitive method for author identification may suffer from undertraining. We built on this idea to propose new approaches for author identification that rely on the idea of bagging features. The first method is a rather traditional method for bagging features and achieved interesting results on the PAN 2012 challenge, reaching the third place among eleven participants on closed identification tasks. The second method extends the bagging feature strategy and provides preliminary promising results.

8 Aknowledgment

Special thanks to Moshe Koppel of Bar-Ilan Univeristy in Israel for filling us with corpus. This work has been done in the context of the SAIMSI project (reference ANR-09-CSOSG-SAIMSI) funded by the French Research Agency (ANR).

References

1. Abbasi, A., Chen, H.: Applying authorship analysis to extremist-group web forum messages. IEEE Intelligent Systems 20(5), 67–75 (Sep 2005)
2. Argamon-Engelson, S., Koppel, M., Avneri, G.: Style-based text categorization: What newspaper am I reading? In: Proceedings of the AAAI Workshop on Text Categorization. pp. 1–4 (1998)
3. Chang, C.C., Lin, C.J.: LIBSVM: A library for support vector machines. ACM Transactions on Intelligent Systems and Technology 2, 27:1–27:27 (2011), software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm
4. FUCKS, W.: On mathematical analysis of style. Biometrika 39(1-2), 122–129 (1952)
5. Hoover, D.L.: Frequent Word Sequences and Statistical Stylistics. Literary and Linguistic Computing 17, 157–180 (2002)
6. Kim, S., Kim, H., Weninger, T., Han, J., Kim, H.D.: Authorship classification: a discriminative syntactic tree mining approach. In: SIGIR (2011)
7. KJELL, B.: Authorship determination using letter pair frequency features with neural network classifiers. Literary and Linguistic Computing 9(2), 119–124 (1994)
8. Koppel, M., Schler, J., Argamon, S.: Computational methods in authorship attribution. Journal of the American Society for Information Science and Technology 60(1) (2009)
9. Koppel, M., Schler, J., Argamon, S., Messeri, E.: Authorship attribution with thousands of candidate authors. In: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval. pp. 659–660. SIGIR '06, ACM, New York, NY, USA (2006)
10. Koppel, M., Schler, J., Bonchek-Dokow, E.: Measuring differentiability: unmasking pseudonymous authors. Journal of Machine Learning Research (2007)
11. Martindale, C., McKenzie, D.: On the utility of content analysis in author attribution: lt;igt;the federalistlt;/igt;. Computers and the Humanities 29, 259–270 (1995), 10.1007/BF01830395
12. MEALAND, D.L.: Correspondence analysis of luke. Literary and Linguistic Computing 10(3), 171–182 (1995)
13. O'Sullivan, J., Langford, J., Caruana, R., Blum, A.: Featureboost: A meta learning algorithm that improves model robustness. In: In Proceedings of the Seventeenth International Conference on Machine Learning. pp. 703–710 (2000)
14. Stamatatos, E.: A survey of modern authorship attribution methods. Journal of the American Society for Information Science and Technology 60(3), 538–556 (2009)
15. Sutton, C., Sindelar, M., Mccallum, A.: Feature bagging: Preventing weight undertraining in structured discriminative learning. Tech. rep., CIIR (2005)
16. Teytaud, O., Jalam, R.: Kernel-based text-categorization. In: In International Joint Conference on Neural Networks (IJCNNŠ2001. pp. 1–0 (2000)
17. Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. In: Computer Vision and Pattern Recognition, 2001