Interactive generic learning method (IGLM)

Ameni Bouaziz, Célia da Costa Pereira, Christel Dartigues Pallez, Frédéric Precioso

Interactive Generic Learning Method (IGLM): A New Approach to Interactive Short Text Classification

Ameni Bouaziz
Univ. Nice Sophia Antipolis,
CNRS, I3S, UMR 7271,
06900 Sophia Antipolis France
bouaziz@i3s.unice.fr

Célia da Costa Pereira
Univ. Nice Sophia Antipolis,
CNRS, I3S, UMR 7271,
06900 Sophia Antipolis France
celia.pereira@unice.fr

Christel Dartigues Pallez
Univ. Nice Sophia Antipolis,
CNRS, I3S, UMR 7271,
06900 Sophia Antipolis France
Christel.DARTIGUES-PALLEZ@unice.fr

Frédéric Precioso
Univ. Nice Sophia Antipolis,
CNRS, I3S, UMR 7271,
06900 Sophia Antipolis France
precioso@unice.fr

ABSTRACT

We propose a method to improve the performance of Random Forests for classifying short texts interactively. In short text classification, the principle of learning algorithms is to build a static model using a training dataset, then to use this model to classify new texts. Many works concentrate on improving the representation of the data as a way to build better models. We intend to tackle the problem in two ways: first by abstracting data to solve the problem of sparseness, and second by taking benefit from already classified data to continuously improve the model. Besides, in order to alleviate the amount of manual annotation, we propose an interactive method in which a manual correct annotation is required only for some misclassified texts, which are then incorporated into the training data to build an updated model. An important challenge is then to determine when to trigger this operation and how to perform the update. Applied on the standard search-snippets dataset, our method allowed a significant improvement.

Keywords

Short Text, Classification, Random Forest, Semantic Abstraction, Interactive learning.

1. INTRODUCTION

There is a consensus in the literature that the large amount of data available poses new challenges to computational methods whose aim is to extract models from these data in order to classify new data. The challenge becomes even more difficult when the aim of the computational method is also to capture massive information arriving continuously [13, 14, 11, 7]. We focus on the issue of classifying short texts which arrive continuously.

Random Forests (RFs) [4] is a powerful classification method used in many machine learning applications. The result of a classification by a RF method is a set of decision trees — the forest —, in which each tree is constructed from independent and identically distributed random training samples. The quality of the forest depends both on the strength of their individual trees and on the correlation between the trees. Like bagging and boosting, the main aim of RFs is to improve the performance of a single classifier. Although such a classification method has proven to be very efficient in many applications, we are convinced that it can be further improved. Here, we propose a method to improve the performance of RFs when they are used to classify short texts[1] interactively. The aim of our work is to tackle three different problems in a single framework. Indeed, we consider the fact that:

- different words can have a similar meaning and, therefore, different features can be considered as belonging to the same semantic concept — *room for improvement from a semantic point of view*;
- in some practical applications, like the analysis of live twitter channels, we need to be able to consider data streams arriving during the classification process. Therefore, a static method in which the number of examples in the training set is fixed *a priori* like in classical RFs is not suitable — *room for improvement from a dynamic point of view*;
- in order to alleviate the amount of manual annotation required, an "expert" may correctly label only a few misclassified short texts which are then incorporated into the training data to build an updated model.

The first point contributes both (i) to decrease the unsuitable consequences of sparsity when classifying short texts and (ii) to improve the capacity of a classification method for capturing the fact that some words can have the same meaning (or be used in the same context) even if they are spelled differently. The second point improves the final classification results in a dynamic setting, i.e., in cases in which

[1]In the rest of the paper, we will also refer to a "short text" as a "message".

SAC'16, April 4-8, 2016, Pisa, Italy

DOI: http://dx.doi.org/xx.xxxx/xxxxxxx.xxxxxxx

new short texts arrive continuously over time. The third point, instead, allows to consider the advantage provided by the knowledge of an expert, if available.

To the best of our knowledge, our method is the first RF-based method proposing such a combined solution to tackle the aforementioned issues.

The rest of the paper is organized as follows. We present some related works in Sections 2 and 3. Our approach is presented in Section 4. Experimental results as well as an analysis of the obtained results are presented in Section 5. Finally, Section 6 concludes the paper.

2. TRAINING DATA PREPROCESSING

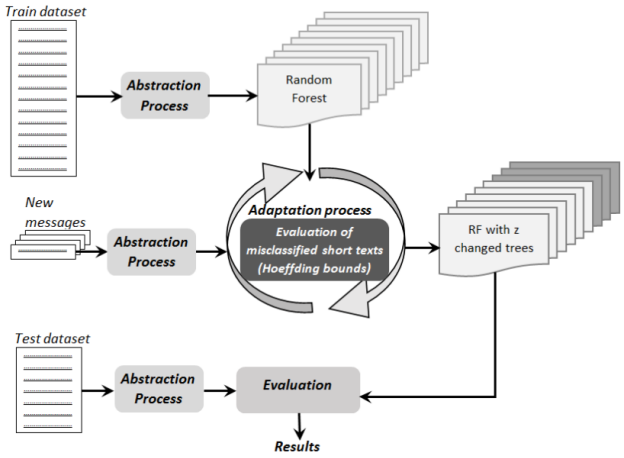
The quality of the training data has a direct impact on the built model. Many works on short text classification proposed several methods to transform texts before going into the construction of the model. One of the most common techniques for text transformation is enrichment, Phan *et al.* [10] propose to use Latent Dirichelet Allocation (LDA) to generate several clusters of words (topics) from a set of external documents, and then to add the words in these topics to the texts in the training dataset based on the semantic links between the words in the texts and the words in the topics. In a similar way, Vo and Ock [19], multiply the sources of external documents to better enrich scientific document titles. In a previous work [3], we proposed a two-level enrichment method which, on the one hand, adds to short texts words from topics generated with LDA and, on the other hand, adds words from topics according to the similarities between the topics and the texts as whole entities. Another text transformation technique is feature selection This method is used by Sun [18] for selecting, from each text to be classified, a set of few "important" words which are then used to search similar elements in a set of labeled texts. Sriram *et al.* [17] filter features from tweets and preserve only the features related to specific predefined domains. A third transformation technique is feature abstraction, which is about replacing current text features by more generic ones. Yang *et al.* [20] map text words to topics generated by LDA. They determine the importance of each topic in each text representation through the importance of the mapped words in the text. Selvescu *et al.* [15] work on the abstraction of complex features obtained from the combination of existing features. Their goal is to reduce the size of the text representation and to build simplified models.

3. LEARNING A DYNAMIC MODEL

There are many approaches to build dynamic models. Sarawagi and Bhamidipaty [12], for example, present the active learning method, whose idea is to start with an initial model built based on a small set of labeled data, then to rebuild it after adding suitable instances from a large pool of unlabeled data. Active learning has been widely used in text classification. Silva and Ribeiro [13] propose to combine active learning with background knowledge to classify texts with the Support Vector Machine (SVM) [6]. In [14], the same authors present an incremental classification method based on active learning. The selection of data to be added to the training set is determined by a confidence factor linked to the SVM margin and computed heuristically.

Another known approach is on-line learning, which is about building a classification model incrementally. On-line learning is very useful for cases where real time construction of the model is required. Saffari *et al.* [11] present the on-line RF: they first use on-line bagging, introduced by Osa [8]. Then they iteratively build tree models respecting some predefined conditions. The last step of on-line RF is to remove regularly some trees from the forest to guarantee the adaptability of the model. Domingos and Hulten [7] focus on the problem of mining high speed data streams. They propose an algorithm to build online trees based on Hoeffding bounds. This work is extended by Phahringer *et al.* [9], where the tree built can have several outputs and a majority vote determines which one is considered. This is a good compromise between RF and the Hoeffding-based tree proposed by Domingos. Another online algorithm to build RF is described by Abdusalam [1]. His algorithm is also based on Hoeffding bounds and a tree branch pruning method. Some other online methods are gathered by Bifet [2].

Figure 1: IGLM approach.



4. IGLM: OUR NEW APPROACH

This section details our IGLM method (Figure 1). The main idea is, first, to build an initial classification model from an initial training dataset. Then, as long as new data arrives/becomes available, an algorithm decides whether to keep the current classification model or to update it. This decision is based on the quality and quantity of the new data. We want to consider the possibility of allowing an expert to manually annotate misclassified short texts in order to interactively improve the quality of the learning data. We have been inspired by the way in which Domingos and colleagues use the Hoeffding bound [7] to decide the quantity of data necessary in the training set to choose iteratively the (best) root test node . Here, instead, we use the Hoeffding bound to decide exactly how many examples are necessary for improving the quality of the forest. As stated above, the data we are working with is a list of short texts. To improve the quality, the classification step is preceded by a preprocessing step which performs feature abstraction (features being the words of the short texts).

4.1 Feature Abstraction

Data are classically organized into a matrix, whose rows are the different short texts and whose columns are all the

words chosen carefully from short texts (after lemmatization and stop word elimination, for instance). Previous works [10, 20, 16] showed that such kind of representation is not sufficient to build a robust classification model for short texts, mainly owing to the resulting sparseness. Some authors, like Yang and colleagues [20], propose a solution to overcome this limitation, in particular by reducing the size of the representative matrix by combining lexical (weighting word method) and semantic features (topics generated from background knowledge). We propose here a similar approach, which reduces the number of columns of the representative matrix. Our proposal is grounded on the method used for feature abstraction. Here, by *feature abstraction*, we mean grouping the words that are semantically close into a set. Each word in the short texts is then replaced by the most representative word of its set. Thanks to this feature abstraction, a short text set can be represented by a much smaller matrix. More precisely, if $\mathbf{M}_{(m\times n)}$ is the representative matrix before abstraction, the representative matrix after abstraction will be $\mathbf{M}'_{(m\times l)}$ where $l < n$. The transformation from $\mathbf{M}$ to $\mathbf{M}'$ uses a mapping matrix $\mathbf{M}''_{(n\times l)}$ in which columns correspond to topics and rows correspond to words:

$$\mathbf{M}'_{(m\times l)} = \mathbf{M}_{(m\times n)} \cdot \mathbf{M}''_{(n\times l)}, \quad (1)$$

where n is the size of the feature space before abstraction, and l is the size of the feature space after abstraction.

To define a set of *abstracted words* that are used for representing short texts, a large scale of external documents semantically close to the training dataset is collected. LDA [10] is then applied to this set of documents in order to generate the groups of words — the *topics*. LDA is a topic model technique which generates a set of meaningful topics from documents. It considers a topic as a probability distribution over words and a document as a probabilistic mixture of topics. Each word in a topic has a weight representing its importance in that topic. We consider the word having the highest weight in a topic as the *representative word* of this topic. It will be used to abstract all the other words belonging to the same topic. The short texts of the dataset are then "abstracted" by replacing each word in the short text by its representative.

4.2 Condition for classification model update

An initial classification model is based on the training set of the short texts obtained after the feature abstraction step. In this section, we will explain how this model is updated when new short texts are received. This step is the most important in IGLM. Indeed, it decides when to update the current model by determining the *optimal number* of examples to add to the training set. The main idea behind this step is to only add to the training set the new short texts that will enrich the training set with the most valuable new information. More precisely, when a new short text is classified by the current model, either it is assigned to the correct class and we consider that the current model (and then the training set) is valid enough, or it is assigned to a wrong class, and in this case, we consider that the model (and then the training set) may need to be updated. The decision whether the classification is correct or not is made interactively by the user. When the classification of a short text fails, the algorithm adds a row to the representative matrix to integrate the misclassified short text. Notice that we decided to update the model only by the misclassified texts in order to keep a low correlation between the RF trees. Indeed, adding well classified texts would introduce redundant information in the model which might result in correlated trees and thus a decrease of the overall RF performance. The algorithm then calculates the difference between the sum of the information gain of all features in the new matrix and the sum of the information gain of all feature in the matrix of the previous update. To illustrate the idea, let us consider a matrix $\mathbf{M}_{(m\times l')}$ and its updated version, $\mathbf{M}_{(m+k\times l')}$, after k misclassified texts. The overall information gain corresponding to $\mathbf{M}_{(m\times l')}$ and $\mathbf{M}_{(m+k\times l')}$ respectively is given by the sum of the information gains corresponding to each single feature in the matrix:

$$SumGain(\mathbf{M}_{(m\times l')}) = \sum_{i=1}^{l'} Gain^{m}(F_i), \quad (2)$$

$$SumGain(\mathbf{M}_{(m+k\times l')}) = \sum_{i=1}^{l'} Gain^{m+k}(F_i), \quad (3)$$

where *Gain* refers to Information Gain (IG). The IG of a feature F_i in a given training set can be computed thanks to the entropy measure: the lower the entropy is, the higher IG of F_i is and the more releveant F_i is.

Let Δ, the *quantified difference* in terms of absolute value between the old and the new data, can be defined as follows:

$$\Delta = |SumGain(\mathbf{M}_{(m+k\times l')}) - SumGain(\mathbf{M}_{(m\times l')})|. \quad (4)$$

The gap between information provided by old data and new data is considered large enough to update the classification model if the difference Δ is greater than ϵ, where ϵ is computed thanks to the Hoeffding bound. The Hoeffding bound allows us to answer the question: "how many iterations do we need to update our model?" Let us consider a real random variable r with range R. For n observations of r, with mean value $\bar{r}$, the Hoeffding bound states that, with probability $1-\delta$, the true mean of r is at least $\bar{r} - \epsilon$, where

$$\epsilon = \sqrt{\frac{R^2 \cdot \ln \frac{1}{\delta}}{2N}} \quad (5)$$

One property of the Hoeffding bound shown in [7], is that k new messages into the matrix are enough to update the classification model if the difference between the best feature and the second best feature is greater than ϵ. In our particular case of short text classification we take into consideration the information brought by all the features. The update will be therefore when $\Delta > \epsilon$. The range R is $\log c$, c being the number of classes and N the number of short texts in the training set, increasing with new messages arriving.

4.3 Model adaptation

As explained in the previous step, if the Hoeffding bound condition is verified after a misclassified short text is found, the algorithm triggers the model adaptation, which updates the classification model to consider the added data. Our update algorithm drops old trees and replaces them with new ones as follows: the efficiency of each tree is calculated separately by computing its accuracy on a short text sample test set. Trees are sorted and the z worst of them are removed from the model and z new trees are built based on the bagging and the random feature selection (RFS) principles. The value z is the same for all the updates:

$$z = 0.1 \times \text{total number of trees}, \quad (6)$$

where the 0.1 factor has been determined empirically. Bagging and RFS are two methods which generate diversity and randomness in RF. Bagging is a selection with repetition of sample subsets from the training set, each subset is used to build a tree. RFS is a selection without repetition of feature subsets from the feature space to build a node in a tree.

5. EXPERIMENTS AND ANALYSIS

5.1 Experimental Dataset

The validation of IGLM was done using the search-snippets dataset, built by Phan [10]. This dataset contains two parts:

- *Short texts corpus*: short texts are the top 20 to 30 replies given by the Google search engine to different queries. They are obtained after the processing of an URL, a title and a short description. Each short text is afterwards assigned to a class determined by the query submitted to obtain it. The whole process led to a corpus of 8 categories, divided each into a training and test set.
- *Universal dataset*: this is a big set of documents collected from the Wikipedia encyclopedia. It is obtained after submitting queries to Wikipedia with some specific keywords. After the application of LDA on these documents, Phan obtained 200 topics containing each 200 words. These topics were used as input of our abstraction process.

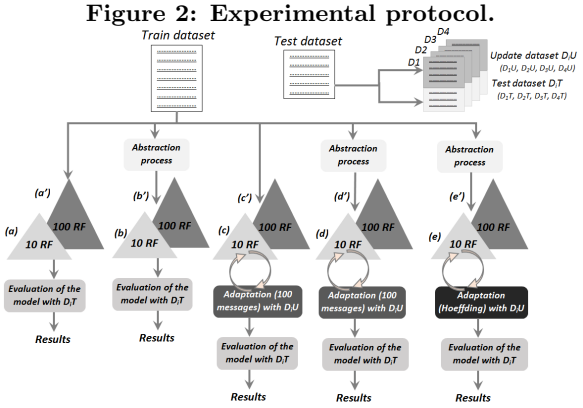
5.2 Experimental Protocol

In order to evaluate the contribution of interactive learning strategy IGLM, we have modified the standard setting of search-snippets dataset. Indeed, we have split the test set into two sets: one we will call the *update set* which will be used to simulate streamed short texts as if learning an adaptive model from a live microblog (as Twitter), and the other, we will call the *true test set* against which the model is indeed evaluated. We thus define the following experimental protocol summarized in Figure 2:

- The training set is the one defined in the original search-snippets setting. We apply the abstraction process on these training data then build an initial RF [2]. This process represents the first iteration of IGLM.
  To build this initial RF, we used the Gini criterion for partitioning data. We set the number of randomly selected features used for each node construction to the square root of the total number of features. We used also the bagging process and the default values of the scikit learn library for the remaining RF parameters. Our tests with different values of these parameters didn't change the outcome of our algorithms.
- Short texts in the update set will be classified by the already built model. Each time a number of them is misclassified and the update condition is satisfied, a new model is built to replace the current one. Each update is considered as an iteration. The true test set will be used to evaluate the updated models.
- The splitting of the *original test set* into an *update set* and a *true test set* is randomly repeated 4 times. We obtain 4 different benchmarks: D_1, D_2, D_3, and D_4.

[2]We use the library "Scikit Learn": http://scikit-learn.org to build the Random Forest

Figure 2: Experimental protocol.



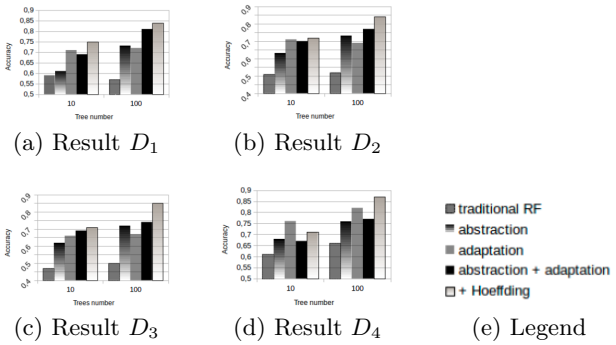
- For each experimental setup, IGLM was run twice on each of these 4 benchmarks: for each benchmark, the first run is done for a RF of 10 trees and with replacement of one tree at each iteration (cf. Figure 2-a, 2-b, 2-c, 2-d, and 2-e); the second run is done for a RF of 100 trees and with replacement of 10 trees at each iteration (cf. Figures 2-a' to 2-e', and Formula 6).
- To validate the different steps of the IGLM, the first experimental setup considers updating the model after a fixed number of misclassified short texts. In this first experimental setup, we set this number to 100. The adaptation has been applied as a first step without applying the abstraction (cf. Figure 2-c, 2-c') and as a second step after applying data abstraction (cf. Figure 2-d and 2-d'). This allows us to determine the improvement obtained by each of the two processes. In the second experimental setup, we introduced the Hoeffding bound principle to let IGLM find the suitable updating time for the model (cf. Figure 2-e and 2-e'), as explained in the Section 4.2. These two experimental setups allowed us to evaluate the contribution of the two proposed adaptation processes.
- In [10, 15, 5], the authors used the accuracy for evaluating their classification algorithms on the "search-snippets" dataset. Here, we have decided to use the same evaluation metric.
- When the Hoeffding bound was considered, at each iteration, we stored in a CSV file the accuracy, the threshold ϵ, the value that triggered the update, and the number of short texts before each update. The adaptation was tested as a first step without applying the abstraction and as a second step after applying data abstraction. This allows us to determine the improvement obtained by each of the two processes.

5.3 Results

As explained in the experimental protocol, in the first experimental setup, the update of the classification model is done after every 100 misclassified short texts. The goal is to measure the classification improvement of the adaptation process compared to the "traditional RF" [4] (based on the classic bag of words representation) and to the abstraction process alone (first iteration). Figure 3 shows the results of these experiments on our 4 datasets D_1, D_2, D_3, and D_4. We can see that after the first iteration of IGLM there is

already a first classification improvement that reaches 40% in the case of D_2 and 100-tree RF. This proves that the abstraction process of IGLM really reduces data sparseness and, therefore, improves the quality of classification.

The use of the adaptation process alone also shows an improvement in short text classification. Indeed, on the D_3 dataset and for 100-tree RF, the adaptation raises the classification accuracy by 34% compared to traditional RF. Our best results in this first experimental setup were generally obtained when combining abstraction and adaptation.[3] We see an improvement for all the datasets: (42% for D_1, 48% for D_2 and D_3, and 16% for D_4, for 100 trees).



(a) Result D_1 (b) Result D_2

(c) Result D_3 (d) Result D_4 (e) Legend

Figure 3: a summary of obtained results with the different protocol steps.

In the previous experimental setup, we have fixed the condition that the model update is triggered after 100 misclassified short texts. In order to relax this condition, we introduce the Hoeffding bound, which determines the suitable step (number of misclassified short texts) to trigger an update. This leads us to the second experimental setup. Table 1 summarizes the results of these experiments. We can observe for the 4 datasets and for both RFs of 10 and 100 trees, the accuracy of the traditional RF, the accuracy of IGML after the first iteration, as well as those concerning the best and the last iterations by using the Hoeffding bound: the best accuracy obtained along the process and the final accuracy obtained. Figure 4 shows the evolution of the accuracy over time. Table 2 and Figure 3 present the results of the two experimental setups (with the (fixed) number of misclassified short texts before updating triggered to 100 and the Hoeffding-bound-based triggering). These two tables show that the introduction of the Hoeffding bound in IGLM considerably improves the accuracy of our classification model. This improvement varies from 14% to 51% for the 10-tree RF and from 31% to 70% for the 100-tree RF, compared to traditional RF, when we consider the last iteration results, and it varies from 18% to 59% for the 10-tree RF and from 34% to 74% for the 100-tree RF, when we consider the best iteration results. Compared to the update triggered by a fixed number of misclassified texts, the Hoeffding-based trigger is more accurate by 14%. The experiments also show that the average number of misclassified texts suitable for updating the model is 11, which is far from 100, used in the first experimental setup. We studied also

[3]Notice that the results shown for "+ Hoeffding" correspond to the second experimental setup with "abstraction + adaptation + Hoeffding.

Figure 4: Accuracy evolution

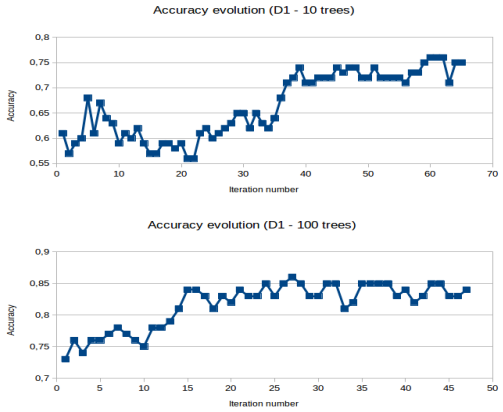
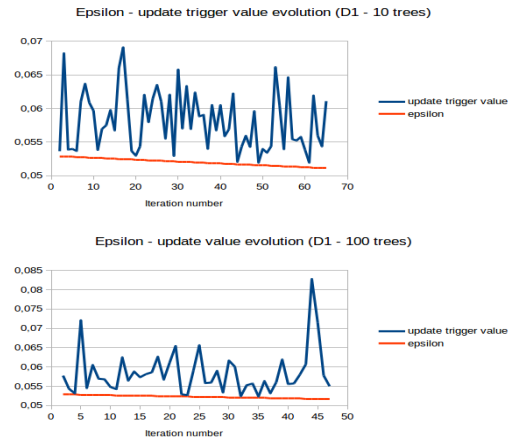


Figure 5: Epsilon - Update value evolution



the evolution of parameter ϵ (cf. Formula 5) over time for our 8 tests. Figure 5 shows that ϵ decreases slowly as long as new short texts arrive. This small variation is explained by the fact that the number of short texts needed for updtaing the model is not large. Figure 5 also shows that, after a given number of newly arrived short texts, the value of ϵ verifies the condition for triggering the model update.

6. CONCLUSION

We have proposed a combined method to improve the performance of RFs in classifying short texts interactively. Our approach tackles both the problems of sparseness and of adapting the classification model when new messages arrive during the classification process. The sparseness has been addressed thanks to a feature abstraction phase. We reduce the number of features considered for representing a short text by replacing each of its words by a *representative word* or *semantic concept* — the most important word in the topic the considered word belongs to. Unlike in classical approaches, all the words in the short texts are considered, even the ones without a representative word. This particularity avoids the loss of information behind such kind of word. The second aforementioned problem has been addressed thanks to an adaptation phase: the Hoeffding bound condition has been used to decide "the suitable iteration"

Table 1: A summary of the obtained results using classical RF, and the different results obtained with IGLM.

	10				100			
	T.RF	1^{st} iter	max	last iter	T.RF	1^{st} iter	max	last iter
D_1	0.59	0.61	0.76	0.75	0.57	0.73	0.86	0.84
D_2	0.51	0.63	0.76	0.72	0.52	0.73	0.85	0.84
D_3	0.47	0.62	0.75	0.71	0.5	0.72	0.87	0.85
D_4	0.61	0.68	0.72	0.7	0.66	0.76	0.89	0.87

Table 2: IGLM results before and after Hoeffding.

Hoeffding		100 short texts	
10	100	10	100
0.75	0.84	0.69	0.81
0.72	0.84	0.7	0.77
0.71	0.85	0.69	0.74
0.71	0.87	0.67	0.77

(after how many misclassified short texts and/or new messages) when the current model ought be updated. The z worst trees are replaced by other new z trees built based on the bagging and random feature selection principles.

Experiments with the "search snippets" dataset have shown the validity of our approach. In particular, we have compared the results obtained with classical RFs with the results obtained with the following configurations (with 10 and 100 trees in the forest): adaptation alone, abstraction alone, adaptation and abstraction (IGLM method). The results show that IGLM with 100 trees outperforms the result by up to 70% compared to classical RF.

Here, we have used the single-granularity topic space to represent short-texts. However, as pointed out by Chen and colleagues [5], by considering a multiple granularity, short texts may be represented more precisely.

7. ACKNOWLEDGMENTS

This work has been co-funded by Région Provence Alpes Côte d'Azur (PACA) and Semantic Grouping Company (SGC).

8. REFERENCES

[1] H. Abdulsalam, D. B. Skillicorn, and P. Martin. Streaming random forests. In *Database Engineering and Applications Symposium, 2007. IDEAS 2007. 11th International*, pages 225–232. IEEE, 2007.

[2] A. Bifet, G. Holmes, B. Pfahringer, R. Kirkby, and R. Gavaldà. New ensemble methods for evolving data streams. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 139–148. ACM, 2009.

[3] A. Bouaziz, C. Dartigues-Pallez, C. da Costa Pereira, F. Precioso, and P. Lloret. Short text classification using semantic random forest. In *Data Warehousing and Knowledge Discovery*, pages 288–299. Springer, 2014.

[4] L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[5] M. Chen, X. Jin, and D. Shen. Short text classification improved by learning multi-granularity topics. In *IJCAI*, pages 1776–1781, 2011.

[6] C. Cortes and V. Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[7] P. Domingos and G. Hulten. Mining high-speed data streams. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 71–80. ACM, 2000.

[8] N. C. Oza. Online bagging and boosting. In *Systems, man and cybernetics, 2005 IEEE international conference on*, volume 3, pages 2340–2345. IEEE, 2005.

[9] B. Pfahringer, G. Holmes, and R. Kirkby. New options for hoeffding trees. In *AI 2007: Advances in Artificial Intelligence*, pages 90–99. Springer, 2007.

[10] X.-H. Phan, L.-M. Nguyen, and S. Horiguchi. Learning to classify short and sparse text & web with hidden topics from large-scale data collections. In *Proceedings of the 17th international conference on World Wide Web*, pages 91–100. ACM, 2008.

[11] A. Saffari, C. Leistner, J. Santner, M. Godec, and H. Bischof. On-line random forests. In *Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on*, pages 1393–1400. IEEE, 2009.

[12] S. Sarawagi and A. Bhamidipaty. Interactive deduplication using active learning. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 269–278. ACM, 2002.

[13] C. Silva and B. Ribeiro. On text-based mining with active learning and background knowledge using svm. *Soft Computing*, 11(6):519–530, 2007.

[14] C. Silva and B. Ribeiro. Improving text classification performance with incremental background knowledge. In *Artificial Neural Networks–ICANN 2009*, pages 923–931. Springer, 2009.

[15] A. Silvescu, C. Caragea, and V. Honavar. Combining super-structuring and abstraction on sequence classification. In *Data Mining, 2009. ICDM'09. Ninth IEEE International Conference on*, pages 986–991. IEEE, 2009.

[16] G. Song, Y. Ye, X. Du, X. Huang, and S. Bie. Short text classification: A survey. *Journal of Multimedia*, 9(5):635–643, 2014.

[17] B. Sriram, D. Fuhry, E. Demir, H. Ferhatosmanoglu, and M. Demirbas. Short text classification in twitter to improve information filtering. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, pages 841–842. ACM, 2010.

[18] A. Sun. Short text classification using very few words. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, pages 1145–1146. ACM, 2012.

[19] D.-T. Vo and C.-Y. Ock. Learning to classify short text from scientific documents using topic models with various types of knowledge. *Expert Systems with Applications*, 42(3):1684–1698, 2015.

[20] L. Yang, C. Li, Q. Ding, and L. Li. Combining lexical and semantic features for short text classification. *Procedia Computer Science*, 22:78–86, 2013.