Textual Event Detection using Fuzzy Fingerprints

Luís Marujo[1,2], Joao Paulo Carvalho[1], Anatole Gershman[2]

Jaime Carbonell[2], João P. Neto[1], David Martins de Matos[1]

[1] INESC-ID, IST – Universidade de Lisboa,

[2] Languages Tecnologies Institute, Carnegie Mellon University

lmarujo@cs.cmu.edu, joao.carvalho@inesc-id.pt, anatoleg@cs.cmu.edu

jgc@cs.cmu.edu, joao.neto@inesc-id.pt, david.matos@inesc-id.pt

Abstract. In this paper we present a method to improve the automatic detection of events in short sentences when in the presence of a large number of event classes. Contrary to standard classification techniques such as Support Vector Machines or Random Forest, the proposed Fuzzy Fingerprints method is able to detect all the event classes present in the ACE 2005 Multilingual Corpus, and largely improves the obtained G-Mean value.

Keywords: Event Detection; Support Vector Machines; Fuzzy Fingerprints; ACE2005.

1 Introduction

Automatic event detection is an important Information Extraction task that can be used to help finding specific content of interest to a user. By event detection we refer to the ability to properly classify text excerpts according to specific categories such as “Meet”, Transport”, Attack, etc. For example, the following news excerpt “The construction of the high speed train line from Madrid to Lisbon, scheduled to start operation in 2017 has been cancelled” should be automatically detected as a “Transport” event.

Event detection has been largely explored in the context of Question Answering, Topic Detection and Tracking (TDT), and Summarization (see Section 2). Here we specifically address the problem of single event detection when in the presence of a large number of classes’ scenario. So we have specific case of a single-label/multi-class classification problem, where each sentence in a document is classified to one target event or to “no event”.

In our experiments we use the ACE 2005 Multilingual Corpus [2], which was specifically developed with this task in mind. Even if this corpus was manually annotated for 27 different single label event types, usually only a few event types are used due to the (arguably) insufficient number of instances necessary to train more traditional classifiers. For example, in [13], only 6 events types are used due to the difficulties in obtaining results with more classes when using Support Vector

Machines (SVMs) [7] or Random Forest [9]; i.e., less than 20% of the possible event types are used.

In this paper we propose the use of Fuzzy Fingerprints [15] as a mechanism to improve automatic event detection for a large number of classes. When applying Fuzzy Fingerprints to the ACE 2005 Multilingual Corpus, it was possible to detect up to 100% of the event types, obtaining a much higher G – mean [12,22], an assessment measure especially adequate to imbalanced multiclass classification problems, when comparing to the best prior event detection method, the SVMs [13].

In order to obtain the results, we started by replicating and confirming the work described by Naughton, and then improved their results by adding several new features to the used Machime Learning Algorithms. We achieved a 4.6% improvement in F1 scores over the results reported by [13]. Then we created a Fuzzy Fingerprints Event library, and adapted the similarity score proposed in [5] to retrieve events in order to improve the results when using all the event types in the ACE 2005 corpus.

This paper is organized as follows: Section 2 introduces the related work. The corpus used to train and evaluate the methods is presented in Section 3. Section 4 details the Machine learning methods used to detect events, and the Fingerprint event detection method is presented in Section 5. Evaluation and results are shown in Section 6. Finally, Section 7 presents the overall conclusions discusses future work.

2 Related Work

In the late 1990s, the event detection problem was addressed under the Topic Detection and Tracking (TDT) trend [6,8,23,24]. The TDT project was divided into 2 primary tasks: First Story Detection or New Event Detection (NED), and Event Tracking. In the NED task, the goal was to discover documents that discuss breaking news articles from a news stream. The Event Tracking task was focused on the tracking of articles describing the same event or topic over time. More recent work using the TDT datasets [1,21,17] on Event Threading tried to organize news articles about armed clashes into a sequence of events, but still assumed that each article described a single event. Passage Threading [1] extends the event threading by relaxing the one event per news article assumption and uses a binary classifier to identify “violent” paragraphs.

Even though the TDT program ended in 2004, new event detection research followed. Automatic Content Extraction (ACE) is the most pertinent example for this work. The goal of ACE Event is detection and recognition of events in text. In addition to the identification events, the ACE 2005 task gives identifies participants, relations, and attributes of each event. This extraction is an important step towards the overarching goal of building a knowledge base of events [4]. The most recent research [19,20] explores bootstrapping techniques and cross-document techniques augmenting the ACE 2005 with other corpora, including MUC-6 (Message Understanding Conference). In this work, the identification task is treated as a word classification task using 34 labels (33 event types + off-event). It is also common to use various lexical features, WordNet[1] synonyms, dependency parsing (only to identify relations), and named-entities extraction [19,20].

3 Corpus

The ACE2005 Corpus was created for the ACE evaluations, where an event is defined as a specific occurrence involving participants described in a single sentence. The corpus has a total of 12,298 sentences. Each sentence is identified with event types or off event. There are 33 event types: Be-Born, Marry, Divorce, Injure, Die, Transport, Transfer-Ownership, Transfer-Money, Start-Org, Merge-Org, Declare-Bankruptcy, End-Org, Attack, Demonstrate, Meet, Phone-Write, Start-Position, End-Position, Nominate, Elect, Arrest-Jail, Release-Parole, Trial-Hearing, Charge-Indict, Sue, Convict, Sentence, Fine, Execute, Extradite, Acquit, Appeal, Pardon.

From these 33 events, only the following 6 have a high number of instances or sentences in the corpus: Die, Attack, Transport, Meet, Injure, and Charge-Indict. These are the only ones used in the previous referred works.

About 16% of the sentences contain at least 1 event, and 15% of those sentences are classified as multi-event (or multi-label); for example, the sentence "three people died and two were injured when their vehicle was attacked" involves 4 event types (or one event with 4 event type labels). This means that multi-event sentences correspond to only 2.50% of the corpus, and were removed since we are addressing single label classification. Also, six event types only occur in multi-event sentences, which means they are not present in the used test and training datasets. Finally, the event Extradite only occurs once in the corpus, and was removed since it is not possible to separate 1 instance into test and training sets. As a result, the dataset used in this work contains 26 different event types.

4 Machine Learning Event detection

A state-of-art way to solve a text multi-class problem, like single event detection, is to use Support Vector Machines (SVM) techniques [25]. Random Forests (RF) [9] are also seen as an alternative to SVM because they are considered one of the most accurate classifier [16]. RF has also advantages on datasets where the number of features is larger than the number of observations [16]. In our dataset, the number of features extracted is between two and three times more than the total number of examples of events.

We followed a fairly traditional approach of training a SVM and RF classifier to classify each sentence into an event label. We used Weka [11] implementations of SVM and RF. These implementation enabled us to test several features, which to the best of our knowledge, have not been used for this purpose. These features include the

[1] http://wordnet.princeton.edu

use of signal words, sentiment analysis, etc. Some of these features lead to significant improvements in the previous G-mean results.

4.1 Feature Extraction

The spoken transcripts documents found in the ACE2005 corpus contain raw Automatic Speech Recognized (ASR) single-case words with punctuation. This means that the transcriptions were either manually produced or were generated by a standard ASR with minimal manual post-processing. Absence of capitalization is known to negatively influence the performance of parsing, sentence boundaries identification, and NLP tasks in general. Recovery of capitalization entails determining the proper capitalization of words from the context. This task was performed using a discriminative approach described in [3]. We capitalized every first letter of a word after a full stop, exclamation, and question mark. After true-casing, we automatically populate three lists for each article: list K of key phrases, list V of verbs, and list E of named entities. The key phrase extraction is performed using a supervised automatic key phrase extraction method [10]. Verbs are identified using Stanford Parser, and named-entities using Stanford NER. This extraction is performed over all English documents of the corpus. The K, V, and E lists are used in the extraction of lexical features and dependency parsing-based features. The lists K and V were also augmented using WordNet synsets to include less frequent synonyms. Furthermore, we manually created list M of modal verbs, and list N of negation terms.

The feature space for the classification of sentences consists of all entries in the lists V, E, K, M, and N which are corpus specific. The value of each feature is the number of its occurrence in the sentence. These numbers indicate the description of events by numbering the number of participants, actions, locations, and temporal information. We have also explored other uncommon types of features: Rhetorical Signals [10] and Sentiment Scores [14]. Finally, we removed all features with constant values across classes. This process reduced by half the number of features and improved the classification results.

5 Fingerprint Event detection

In this work, we propose the use of an adaptation of the Fuzzy Fingerprints classification method described in [15,5] to tackle the problem of Event detection. In [15] the authors approach the problem of text authorship by using the crime scene fingerprint analogy to claim that a given text has its authors writing style embedded in it. If the fingerprint is known, then it is possible to identify whether a text whose author is unknown, has a known author’s fingerprint on it.

The algorithm itself works as follows:

1) Gather the top-*k* word frequencies in all known texts of each known author;

2) Build the fingerprint by applying a fuzzifying function to the top-*k* list. The fuzzified fingerprint is based on the word order and not on the frequency value;

3) Perform the same calculations for the text being identified and then compare the obtained text fuzzy fingerprint with all available author fuzzy fingerprints. The most similar fingerprint is chosen and the text is assigned to the fingerprint author;

The proposed fuzzy fingerprint method for Event Detection, while similar in intention and form, differs in a few crucial steps.

Firstly it is important to establish the parallel between the context of author ownership and Event Detection. Instead of author fingerprints, in this work we are looking to obtain the fingerprints of Events. Once we have an event fingerprint library, each unclassified sentence can be processed and compared to the fingerprints existing in the event library.

Secondly, a different criterion was used in ordering the top-k words for the fingerprint. While in [15] it is used the word frequency as the main feature to create and order the top-k list, here we use an adaptation of an Inverse Document Frequency (*idf*) technique, aiming at reducing the importance of frequent terms that are common across several events.

Finally, the similarity score differs from the original, based on the fact that the source sentence are, by design, very short texts, while the original Fuzzy Fingerprint method was devised to classify much longer texts (newspaper articles, books, etc. ranging from thousands to millions of characters). Here we propose the use of a score with values between 0 and 1, where the lowest score indicates that the sentence in question is not related to the topic fingerprint, and the highest value indicates that the sentence and the event fingerprints are the same.

5.1 Building the Event Fingerprint Library

In order to build the Event fingerprint library, the proposed method goes over the event training set, which is composed of 80% of the sentences annotated as describing the event in question, and counts the word frequency. Only the top-k most frequent words are considered.

The main difference between the original method and the one used here is due to the small size of each sentence: in order to make the different event fingerprints as unique as possible, its words should also be as unique as possible. Therefore, in addition to counting each word occurrence, we also account for of its Inverse Topic Frequency (itf), an adaptation of the traditional inverse document frequency – idf:

$$itf_v = \frac{N}{n_v}, \tag{1}$$

where N is the cardinality of the event fingerprint library (i.e., the total number of events), and n_v becomes the number of fingerprint events where the specific word v is present. The ITF allows moving the position of common words down on top-k list, therefore decreasing their relevance.

After obtaining the top-k list for a given event, we follow the original method and apply a fuzzy membership function to build the fingerprint. The selected membership function is a Pareto-based linear function, where 20% of the top k elements assume 80% of the membership degree:

$$\mu(i) = \begin{cases} 1 - (1-b)\frac{i}{k}\ , \ i \leq a \\ a\left(1 - \frac{i-a}{k-a}\right), i > a \end{cases}, a, b = 0.2 \qquad (2)$$

The fingerprint is a k sized bi-dimensional array, where the first column contains the list of the top-k words ordered by their tf-ITF, and the second column contains the word i membership value $\mu(i)$ obtained by the application of (2).

5.2 Retrieving the Sentence-to-Event Score

In the original method, checking the authorship of a given document would proceed as follows: build the document fingerprint (using the exact procedure described above); compare the document fingerprint with each fingerprint present in the library and choose the highest score. Within the Event detection context, such approach would not work due to the very small number of words contained in one sentence since it simply does not make sense to count the number of individual word occurrences. Therefore we developed a Sentence-to-Event score (S2E) that tests how much a sentence fits to a given event fingerprint. The S2E function (3), provides a normalized value ranging between 0 and 1, that takes into account the size of the (preprocessed) sentence (i.e., its number of features). In the present work, the features are simply the set of words present in the sentence. Not even stop-word removal is performed (empirical results shown that the best results were obtained without stop-words removal, or imposing a minimum word size).

$$S2E(\Phi, S) = \frac{\sum_v \mu_\Phi(v) : v \epsilon (\Phi \cap S)}{\sum_{i=0}^{j} \mu_\Phi(w_i)} \qquad (3)$$

In (3), Φ is the Event fingerprint, S is the set of words of the sentence, $\mu_\Phi(v)$ is the membership degree of word v in the event fingerprint, and j is the is the number of features of the tweet. Essentially, S2E divides the sum of the membership values $\mu_\Phi(v)$ of every word v that is common between the sentence and the event fingerprint, by the sum of the top j membership values in $\mu\Phi(w_i)$ where $w \in (\Phi)$. Eq. (3) will tend to 1 when most to all words of the sentence belong to the top words of the fingerprint, and tend to 0 when none or very few words of the sentence belong to the bottom words of the fingerprint.

6 Evaluation and Results

6.1 Evaluation metrics

The standard evaluation metrics used in text classification tasks are Precision (P_i), Recall (R_i), and F1-measure ($F1_i$), where i is the class index. Precision is the fraction of sentences correct classified (a.k.a. true positives, *tp*) over all sentences classified with the same event label, i.e., the sum of *tp* with false positives (*fp*):

$$P_i = \frac{\#tp}{\#tp + \#fp}$$

Recall is the fraction of sentences belonging to an event label that were successfully identified:

$$R_i = \frac{\#tp}{\#tp + \#fn}$$

The F1-measure combines the precision and recall in the following way:

$$F1_i = \frac{2\,P_i R_i}{P_i + R_i}$$

The disadvantage of these metrics is in the sensibility to imbalanced distribution of the data. The average F1-measure is not the best evaluation metric for datasets with several classes, because it is possible to obtain high F1-measure values while still failing to detect a relevant number of classes. To overcome this limitation, Kubat et al. proposed the G-mean metric (4)[12] to evaluate imbalanced binary classification problems. The extension of G-mean to imbalanced multiclass classification problems was proposed by Sun et al. in [22]. G-mean is defined as the geometric mean of the recall values R_i, and therefore has the disadvantage of assuming the value zero when at least one recall value R_i is zero. To overcome this limitation, we introduce a smoothing G-Mean version, the SG-Mean (5). A smoothing constant (e.g., δ = 0,001) added to each R_i solves the problem of multiplication by zero if a class is not detected. With this metric it is possible to evaluate the performance of a method while still considering the loss of classes.

$$G - Mean = \left(\prod_{i=1}^{n} R_i\right)^{\frac{1}{n}} \tag{4}$$

$$SG - Mean = \left(\prod_{i=1}^{n} R_i + \delta\right)^{\frac{1}{n}}, \delta > 0 \tag{5}$$

To complement these metrics, we also report the number of classes that the methods fail to detect ($\#R_i$=0).

6.2 Results

The SVM performed better than Random Forest to detect events in low to medium number of classes. For these reason we chose SVM to investigate the inclusion of the additional features over the baseline set proposed by Naughton [3]. We have also investigated the influence of the new features introduced in this work by using all features except for the ones under test. These novel features raised the G-Mean scores by 16.6% (Table 1) when detecting six events. The average F1 value was also improved to 0.496.

The inclusion of the dependency parse based features raised the G-Mean score by 16,3 %, which is the highest contribution among the new features. The second best result, using domain-Id features, is 2,93%. The relevance based features, such as the sentiment analysis, and rhetorical features had the lowest contribution with respectively 2,7% and 1,7%. As expected, the introduction of new features reduced the recall of the majority class (no-event or off event) between -1,9% and -1,3%, but improved the recall of the remaining labels. The exception to this fact is the detection of “Die” events that was also penalized. This can be explained in part by the imbalanced distribution of the event types, which bias the classifier towards more frequent event types. In this case, the classifier is bias towards “Attack” events, which is three times more frequent than “Die” and share similar new features values.

When increasing the number of event types to cover all the 26 events present in the ACE 2005 database, the SVM performed very poorly, failing to detect 11 of the 26 events. This implied a G-Mean = 0, and the F1 decreased to 0.241. The SG-Mean was also a rather poor 0.0029.

Table 1 Feature Extraction analysis using R_i results in ACE 2005 using SVM with improved features

Labels	All Features	All - Rhetorical Signals	All - Dependency Parsing	All - Sentiment Analysis	All - Domain Id	Baseline Features
Injure	0,444	0,444	0,333	0,444	0,444	0,444
Transport	0,233	0,219	0,123	0,233	0,247	0,164
Attack	0,435	0,413	0,449	0,406	0,449	0,406
N	0,932	0,930	0,935	0,936	0,930	0,948
Meet	0,583	0,583	0,500	0,583	0,583	0,542
Charge-Indict	0,444	0,444	0,444	0,444	0,333	0,222
Die	0,375	0,375	0,375	0,333	0,375	0,417
G-Mean	**0,456**	0,448	0,392	0,444	0,443	0,391

Several tests were done in order to find the best Fuzzy Fingerprint parameters. The best empirical results led to the inclusion of all words in the fingerprints (i.e., include stop words and small sized words). The fingerprint size *K* was optimized for the best SG-mean. Figs 1-3 show the obtained SG-Mean and number of undetected event classes for 6 and 26 events for several values of *K*.

With 6 events, the best G-Mean=0.564 and SG-Mean=0.565, were obtained for *K*=200, and represent an improvement of around 25% when compared to the best SVM result. However, the F1=0.367, was lower.

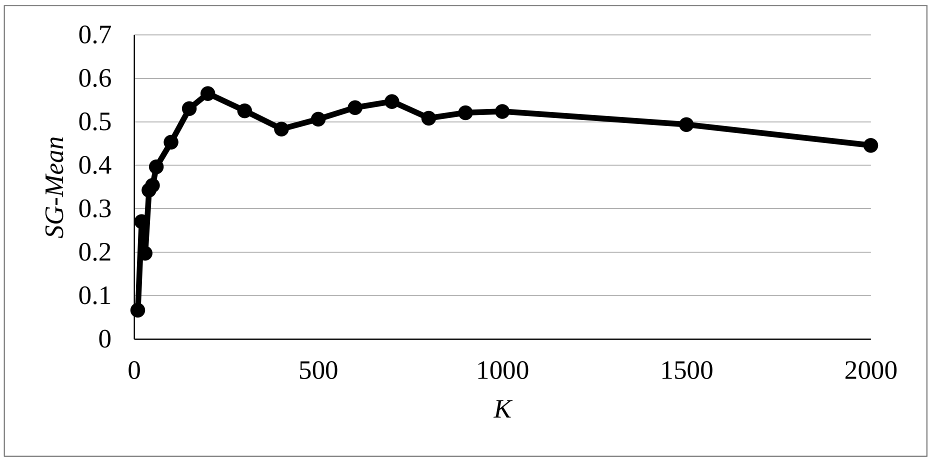

Fig. 1 SG-Mean results in the ACE 2005 with 6 events using Fuzzy Fingerprints with several *K* values

When tested for the 26 events database, the Fuzzy Fingerprints method is able to detect all the classes for the values of *K*>=400, while the SVM only detects 15 out of 26 classes. The best SG-Mean is 0.441, a 15x improvement over the best SVM result. The best F1=0.244 is similar for both methods.

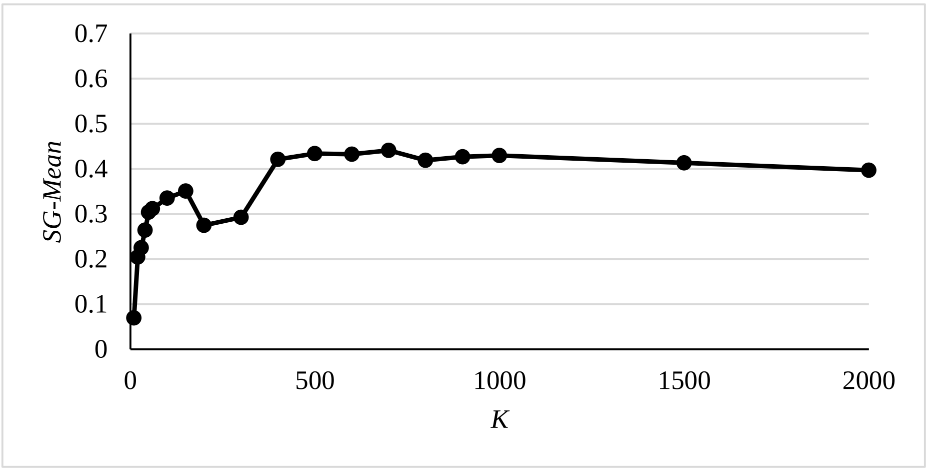

Fig. 2 SG-Mean results in the ACE 2005 with 26 events using Fuzzy Fingerprints with several *K* values

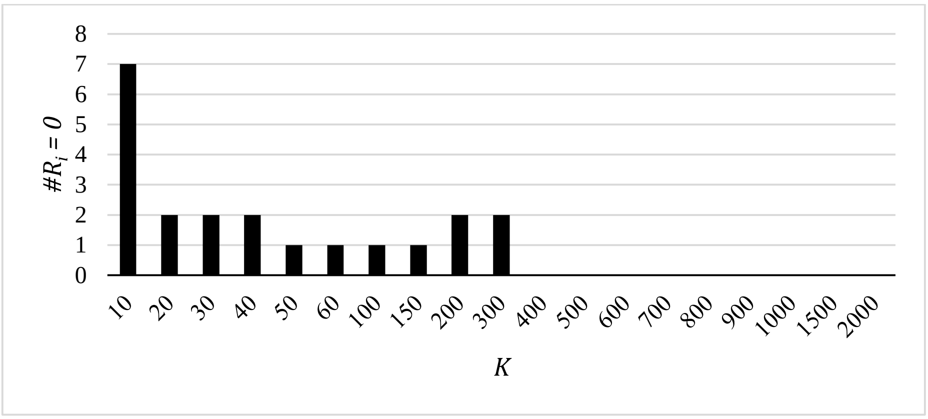

Fig. 3 #R_i=0 (number of event classes missed) results in the ACE 2005 with 26 events using Fuzzy Fingerprints with several *K* values

Finally, Tables 4 and 5 show the comparative results (including RF) for the 6 and 26 events ACE2005 database. Best results are shown in bold.

Table 4 – Results in the ACE 2005 corpus with 6 events

Measure	**Random Forest**	**SVM (SMO)**	**Fuzzy Fingerprints (best)**
F1 (Avg.)	0.371	**0.496**	0.367
G-Mean	0	0.456	**0.564**
SG-Mean	0.008	0.457	**0.565**
#R_i = 0	4	**0**	**0**

Table 5 – Results in the ACE 2005 corpus with 26 events

Measure	**Random Forest**	**SVM (SMO)**	**Fuzzy Fingerprints (best)**
F1 (Avg.)	0.037	**0.241**	**0.244**
G-Mean	0	0	**0.440**
SG-Mean	0.002	0.029	**0.441**
#R_i = 0	23	11	**0**

7 Conclusions and Future work

In this paper, we approached the problem of detecting events at sentence level, a single label classification procedure whose results can be used to improve several NLP tasks such as personalization, recommendation, question answering, and/or summarization. We are specifically interested in the cases where a large number or event classes must be detected, since more traditional classifiers, such as SVM or RF, usually loose a large number of classes. We started by improving the best previously known approaches, and then proposed the use of Fuzzy Fingerprints. The ACE2005 corpus, which contains 26 different single event classes was used throughout the experiments. The results show that it is possible to detect all 26 different event types when using the Fuzzy fingerprints approach, while the best competitor, an SVM with enhanced features, only detects roughly 60% of the different types of events. This leads to a huge increase in the G-Mean results when using the Fuzzy Fingerprints method.

Even if not mentioned throughout the paper, the Fuzzy Fingerprints method also has the advantage of being much more efficient in computational terms. In our test conditions, more than 20x faster to classify 26 event types than when using SVMs.

The application of the Fuzzy Fingerprints is still in an early development phase. Future work includes using advanced features such as key phrases to build the fingerprints, and also the fuzzification of key phrases. It is also in our plans to apply the method to the detection of multiple events in single sentences.

8 Acknowledgements

This work was supported by national funds through FCT Fundação para a Ciencia e a Tecnologia, project PTDC/IVC-ESCT/4919/2012, project PEstOE/EEI/LA0021/2013, and through the FCT-Carnegie Mellon Portugal Program under grant SFRH/BD/33769/2009.

References

1. A. Feng and J. Allan (2007). Finding and linking incidents in news. In Proceedings of the sixteenth ACM conference on Conference on information and knowledge management, pp. 821-830, ACM.
2. C. Walker, S. Strassel, J. Medero. ACE2005 Multilingual training Corpus. LDC2006.
3. F. Batista, H. Moniz, I. Trancoso, N. Mamede. Bilingual Experiments on Automatic Recovery of Capitalization and Punctuation of Automatic Speech Transcripts. IEEE Transactions on Audio, Speech, and Language Processing, 20(2):474 - 485, 2012.
4. H. Ji and R. Grishman. Knowledge base population: Successful approaches and challenges. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL 2011), pages 1148 - 1158, Portland, Oregon, USA, 2011.
5. H. Rosa, F. Batista, J.P. Carvalho, *Twitter Topic Fuzzy Fingerprints*, Proc. of the WCCI2014 – World Congress of Computational Intelligence, 2014, Beijing, China, IEEE Xplorer
6. J. Allan, J. Carbonell, G. Doddington, J. Yamron, Y. Yang, B. Archibald, and M. Scudder. Topic Detection and Tracking Pilot Study Final Report. In Proceedings of the Broadcast News Transcription and Understanding Workshop, 1998.
7. J. C. Platt, Fast training of support vector machines using sequential minimal optimization, Advances in kernel methods, 1999, pp 185-208

8. J. Carbonell, Y. Yang, J. Laerty, R. Brown, T. Pierce, and X. Liu. CMU Approach to TDT: Segmentation, Detection, and Tracking. In Proceedings of the 1999 Darpa Broadcast News Conference, 1998.
9. L. Breiman, (2001). Random forests. Machine learning, 45(1), 5-32.
10. L. Marujo, A. Gershman, J. Carbonell, R. Frederking, and J. P. Neto. Supervised Topical Key Phrase Extraction of News Stories using Crowdsourcing, Light Filtering and Co-reference Normalization Pre-Processing. In Proceedings of 8th International Conference on Language Resources and Evaluation (LREC), 2012.
11. M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, I. H. Witten (2009); The WEKA Data Mining Software: An Update; SIGKDD Explorations, Volume 11, Issue 1.
12. M. Kubat, S. Matwin. Addressing the curse of imbalanced training sets: one-sided selection. In Proceedings of ICML, pages 179-186, 1997.
13. M. Naughton, N. Strokes, J. Carthy. Investigating Statistical Techniques for Sentence-Level Event Classification. Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1. Association for Computational Linguistics, 2008.
14. M. Thelwall, K. Buckley, G. Paltoglou, and D. Cai. Sentiment strength detection in short informal text. Journal of the American Society for Information Science and Technology, 61(12):2544 - 2558, 2010.
15. N. Homem, J.P. Carvalho, *Authorship Identification and Author Fuzzy Fingerprints*; Proc. of the NAFIPS2011 - 30th Annual Conference of the North American Fuzzy Information Processing Society, 2011, IEEE Xplorer
16. R. Diaz-Uriarte, S.A. De Andres, Gene selection and classification of microarray data using random forest, 2006, BMC bioinformatics, 7(1):3.
17. R. Nallapati, A. Feng, F. Peng, and J. Allan. Event threading within news topics. In Proceedings of the Thirteenth ACM conference on Information and knowledge management - CIKM '04, pp 446-453, New York, New York, USA, 2004. ACM Press.
18. R. Saurí, R. Knippen, M. Verhagen, J. Pustejovsky. Evita: a robust event recognizer for QA systems. HLT/EMNLP2005.
19. S. Liao and R. Grishman. Filtered ranking for bootstrapping in event extraction. In Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), number August, pages 680 - 688, Beijing, 2010.
20. S. Liao and R. Grishman. Using document level cross-event inference to improve event extraction. In Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL 2010), number July, pages 789-797, Uppsala, Sweden, 2010.
21. Y. Hong, J. Zhang, B. Ma, and J. Yao. Using cross-entity inference to improve event extraction. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistcs (ACL 2011), number 4, pages 1127 - 1136, Portland, Oregon, USA, 2011.
22. Y. Sun, M.S. Kamel, and Y. Wang. Boosting for learning multiple classes with imbalanced class distribution. In Proc. of ICDM'06, pages 592-602. IEEE, 2006.
23. Y. Yang, J. Carbonell, R. Brown, T. Pierce, B. Archibald, and X. Liu. Learning approaches for detecting and tracking news events. IEEE Intelligent Systems and their Applications, 14(4):32-43, 1999.
24. Y. Yang, T. Pierce, and Carbonell. A Study of Retrospective and On-line Event Detection. Proceedings of the 21st annual international ACM SIGIR 98, 1998.
25. Y. Yang, X. Liu. A re-examination of text categorization methods. In Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval, 1999, pp. 42-49. ACM.