Update Frequency and Background Corpus Selection in Dynamic TF-IDF Models for First Story Detection

Fei Wang[1,2](✉), Robert J. Ross[1,2], and John D. Kelleher[1,2]

[1] School of Computer Science, Technological University Dublin, Dublin, Ireland
d13122837@mydit.ie, {robert.ross,john.d.kelleher}@dit.ie
[2] ADAPT Research Centre, Dublin, Ireland

Abstract. First Story Detection (FSD) requires a system to detect the very first story that mentions an event from a stream of stories. Nearest neighbour-based models, using the traditional term vector document representations like TF-IDF, currently achieve the state of the art in FSD. Because of its online nature, a dynamic term vector model that is incrementally updated during the detection process is usually adopted for FSD instead of a static model. However, very little research has investigated the selection of hyper-parameters and the background corpora for a dynamic model. In this paper, we analyse how a dynamic term vector model works for FSD, and investigate the impact of different update frequencies and background corpora on FSD performance. Our results show that dynamic models with high update frequencies outperform static model and dynamic models with low update frequencies; and that the FSD performance of dynamic models does not always increase with higher update frequencies, but instead reaches steady state after some update frequency threshold is reached. In addition, we demonstrate that different background corpora have very limited influence on the dynamic models with high update frequencies in terms of FSD performance.

Keywords: Novelty detection · First Story Detection · Nearest neighbour · TF-IDF · Update frequency · Background corpus

1 Introduction

Novelty detection is the task of identifying data that are different in some salient respect from other predominant chunks of data in a dataset [14]. In most cases, there is not an explicit definition for novelty or sufficient novel data to form a class of novelty before detection. Instead, novelty detection is normally treated as an unsupervised learning application, i.e. no labels are available and the detection is implemented based on only the intrinsic properties of the data [16].

Online novelty detection is a special case of novelty detection, in which input data are time-ordered streams. The online characteristic brings in two additional constraints [9]: 1) the detection should be made quickly, e.g. before subsequent

© Springer Nature Singapore Pte Ltd. 2020
L.-M. Nguyen et al. (Eds.): PACLING 2019, CCIS 1215, pp. 206–217, 2020.
https://doi.org/10.1007/978-981-15-6168-9_18

data arrives; and 2) looking forward is prohibited during detection, i.e. the detection can only be made based on the data that has already arrived. One important application of online novelty detection within Natural Language Processing (NLP) is to the task of First Story Detection (FSD). In FSD, the target text documents are all stories that discuss some specific events. Given a stream of stories in chronological order, the goal of FSD is to find out the very first story for each event [2]. The stories are processed in sequence, and for each incoming candidate story, a decision is made on whether or not it discusses an event that has not been seen in previous stories; the decision making process is normally based on a novelty score, namely, if the novelty score of an incoming story is higher than a given threshold, we say the candidate is a first story.

Since it was first defined within the Topic Detection and Tracking (TDT) competition series in 1998 [1,19], hundreds of models have been proposed for the task of FSD. Nearest neighbour-based models with the traditional term vector document representations currently achieve the state of the art in FSD [11,13,16]. Because of its online characteristic, a dynamic term vector model that is incrementally updated during detection is usually adopted for FSD instead of a traditional static model [7,11,12]. However, very little previous research has investigated how a dynamic term vector model works in practice for FSD, or has investigated how to select hyper-parameters (such as the model update frequency) and background corpora for such dynamic models.

In this paper, we first theoretically analyse how a dynamic term vector model works for FSD, and then empirically evaluate the impacts of different update frequencies and background corpora on FSD performance. Our results show that dynamic models with high update frequencies outperform static models and dynamic models with low update frequencies; and, importantly, also show the FSD performance of dynamic models does not always increase along with increases in the update frequency. Moreover, we demonstrate that different background corpora have very limited influence on the dynamic models with high update frequencies in terms of FSD performance.

2 First Story Detection

As mentioned in Sect. 1, a wide variety of models have previously been investigated for FSD. These models can generally be grouped into three categories [16]: Point-to-Point (P2P) models, Point-to-Cluster (P2C) models, and Point-to-All (P2A) models. Accordingly, their novelty scores are defined as the distance of the incoming data to: an existing data point for P2P models, a cluster of existing data points for P2C models, and all the existing data points for P2A models. The P2P models are normally nearest neighbour-based [2,3,19] or approximate nearest neighbour-based [7,12] that aim at finding the most similar existing story to the incoming story. The P2C models use clusters of existing stories to represent previous events and evaluate the incoming story by comparing it with these clustered events [2,19]. The P2A models typically build a machine learning system with all the existing stories and apply this system to the incoming story to generate a novelty score [8,15,18]. Generally speaking, the nearest neighbour-based P2P models outperform the other two categories of models.

When applying a nearest neighbour-based model to FSD, the first step is to represent each text story with a document representation vector so that quantitative comparisons can be made between stories. In recent years a large number of deep learning-derived distributed document representations have been proposed that have achieved excellent performance across many NLP tasks [6]. However, for the task of FSD, the state of the art is still achieved with traditional term vector document representations [16], in which each term is represented with a single feature (dimension) in the term vector space. The most well-known term vector model is TF-IDF, short for term frequency - inverse document frequency, in which the weight of each term in a specific document is calculated as the product of the TF (term frequency) component and the IDF (inverse document frequency) component. There are many schemes of calculating these two components, but a widely-applied scheme is shown as follows [3,7]:

$$tf\text{-}idf(t,d) = tf(t,d) \times idf(t) \tag{1}$$

$$idf(t) = log\frac{N}{df(t)} \tag{2}$$

where $tf(t,d)$, representing the TF component, is the number of times the term t occurs in document d, and $idf(t)$, representing the IDF component, is the logarithmic value of the proportion of the total number of documents N divided by $df(t)$, i.e., the number of documents that contain the term t. Briefly speaking, the more a term occurs in a target document, and the less it occurs in other documents, the bigger the TF-IDF weight is for that term for that document.

From the definition of TF-IDF, we can also see that the calculation of the IDF component requires a corpus with a number of documents. However, because of the "looking back only" constraint of online detection, the target corpus for FSD detection is always unavailable for the construction of the TF-IDF model prior to detection, and thus an additional background corpus is required. Specifically, based on the background corpus, the TF-IDF model builds a term vocabulary and calculates the IDF component for each term in the vocabulary. After this step, there are two different ways to implement TF-IDF models for FSD [3,19]. The first option is to apply this fixed model, i.e., the fixed vocabulary and IDF components, to the stories in the target FSD corpus. This type of model is called a *static* TF-IDF model. In this way, any term that is unseen in the background corpus will be ignored in the detection process. The second option is to incrementally update the model, i.e., the vocabulary and IDF components, during the detection process after a number of documents arrive. This type of model is called a *dynamic* TF-IDF model. In this way, the terms that are unseen in the background corpus but have been seen up to a certain point in the target data stream are also taken into account. Because of its online nature, a dynamic model should usually be adopted for FSD [7,11,12]. Given this, in the next section, we will analyse how a dynamic TF-IDF model works for FSD and its difference from a static model in this context.

3 Dynamic Term Vector Models for First Story Detection

For a dynamic TF-IDF model we adopt an adjusted form of Eqs. 1 and 2 from earlier. Specifically we adopt the following:

$$tf\text{-}idf(t,d) = tf(t,d) \times idf(t)' \tag{3}$$

$$idf(t)' = log\frac{N'}{df(t)'} \tag{4}$$

where $tf(t,d)$ remains the same as that in Eq. 1, but the calculation of the IDF component $idf(t)'$ now makes use of an N' that captures the total number of not only the documents in the background corpus but also the stories in the target FSD corpus up to the present point, and, similarly, $df(t)'$ refers to the number of documents across both the background corpus, and the portion of target corpus to the current point, that contain the term t.

Due to the dynamic nature of the TF-IDF model, the length and features captured by a document vector now vary as we move through events, and this has potential implications to the FSD process. To illustrate, let us consider two documents (one being the candidate story and the other some story that has already been processed by our model). The comparison of these two documents is typically achieved with a distance metric; here we will assume the widely-used cosine distance:

$$cosine\_distance(\boldsymbol{d},\boldsymbol{d'}) = 1 - \frac{\boldsymbol{d}\cdot\boldsymbol{d'}}{|\boldsymbol{d}||\boldsymbol{d'}|} \tag{5}$$

where $\boldsymbol{d}$ is the TF-IDF vector for the candidate story and $\boldsymbol{d'}$ is the TF-IDF vector for a historic story that we are comparing to.

In order to better understand how a dynamic model performs for FSD, in Table 1, we unfold these two document vectors to m term features from t_1 to t_m, where m is the length of the vocabulary. In a dynamic model, the vocabulary includes both terms that were present in the background corpus and new terms that are added during the updates to the model. However, irrespective of whether a term is a new term or not, the value for a term in the TF-IDF document representation is the weight of the specific term based on the dynamic TF-IDF model.

In order to analyse how a TF-IDF representation treats both old and new terms in a document representation we group the features in our document representation into two parts: Range A which includes terms that have been present in the model for a substantial amount of time (because they were present in the background corpus or were added to the model several updates previously); and Range B which includes terms that have been added to the model recently.

In a static TF-IDF model there are only terms in Range A coming from the background corpus, and no term in Range B since the target corpus is not incorporated into the TF-IDF representation. Thus, the performance of the TF-IDF

Table 1. Two document representation vectors based on a dynamic TF-IDF model

	Range A			Range B		
	$\boldsymbol{t_1}$	**...**	$\boldsymbol{t_i}$	$\boldsymbol{t_{i+1}}$	**...**	$\boldsymbol{t_m}$
$\boldsymbol{d}$	v_1	...	v_i	v_{i+1}	...	v_m
$\boldsymbol{d'}$	v'_1	...	v'_i	v'_{i+1}	...	v'_m

model depends on how well the weights from the background corpus represent the terms in our target corpus. As the term weights are only calculated based on the background corpus, the selection of the background corpus has a great impact on the static TF-IDF model, and also influences the FSD performance [17]. Thus, a large-scale domain-related background corpus is normally adopted to generate realistic weights for the terms.

For a dynamic TF-IDF model, however, although it can use a large background corpus initially, new terms that are unseen in the background corpus will emerge and be incorporated into the model as detection proceeds – thus forming Range B. By definition these new terms did not occur in the background corpus, this may be because the new terms are genuinely rare in language, or else it may be because the selected background corpus was not representative of the language in the target data stream that the model is processing, or finally the new term may be a true neologism in a language. Whatever the true cause for why a particular term is a new term for a model, the weights of these new terms may be not well calibrated with respect to the weights for the terms in Range A. In Eq. 4, $df(t)'$ denotes the number of documents that contain the term t not only in the already-processed target stories, but also in the documents of the background corpus. However, by definition new terms in Range B will not have appeared in the background corpus and will only have appeared in the most recent documents in the target data stream. Therefore, the value of $df(t)'$ of a new term in Range B will be very small compared to N' in Eq. 4, and thus the TF-IDF weights for these new terms are normally very large, so we call these the rough weights with respect to the realistic weights in Range A. In the calculations of cosine distance (Eq. 5) more attention is focused on the features with larger values, and thus, the terms in Range B have a bigger effect on comparison calculations based on a dynamic TF-IDF model than they are expected to have based on the language.[1]

From the analysis above, we find that a key difference between dynamic and static TF-IDF models, when making comparisons between document vectors, is that the focus of dynamic models is more on the new terms with large rough

[1] It is worth noting that if looking at the whole FSD process rather than the comparison between two specific document vectors, new terms keep on being added into Range B as the updates are implemented. On the other hand, the terms already existing in Range B keep on being moved to Range A as more and more new stories arrive and the number of stories since the term's first appearance becomes large enough to generate realistic weights.

weights that emerge during detection, whereas static models focus on the existing terms whose weights are calculated only based on the background corpus.

In order to improve a static model, or indeed the static elements of a dynamic model, we can try to find a more suitable background corpus in order to generate realistic weights for the terms in the target data stream. However, for the dynamic approach it is hard to improve performance from a theoretical perspective due to the way in which weights are calculated for newly encountered terms. To overcome this limitation and try to optimise the dynamic aspects of TF-IDF modelling for FSD, in the next section we present an experimental analysis to investigate the impact of update frequency and background corpus on the model.

4 Experimental Design

In the following, we present our experimental design for evaluating the impact of the dynamic aspects of a TF-IDF model in the context of the FSD task. We focus on the impact of different update frequencies and the relevance of background corpora selection.

4.1 Target Corpus

In our experiments, we use the $TDT5$ corpus[2] as the target corpus for FSD detection. This corpus contains approximately 278 thousands newswire stories generated from April to September 2003.

4.2 Background Corpora

For the evaluation of the impact of different background corpora on the FSD results, we selected $COCA$ (The Corpus of Contemporary American English) [4] and $COHA$ (Corpus of Historical American English) [5] as the basic background corpora to be used for evaluation. The former is a comprehensive contemporary English documents collection from 1990 to present in different domains such as news, fiction, academia and so on. The latter is similar to $COCA$ in themes but covers the historical contents from 1810 to 2009. The numbers of documents in $COCA$ and $COHA$ are approximately 190,000 and 115,000 respectively. In both cases we only make use of the subsets of the two corpora that predate 2003, i.e., the year of $TDT5$'s collection. Additionally, to tease out the influence of domain relevance, we also divide the $COCA$ corpus into two distinct subsets - $COCA\_News$ and $COCA\_Except\_News$. $COCA\_News$ contains only the documents in the domain of news, which is the same as the domain of the target $TDT5$ corpus, while $COCA\_Except\_News$ contains the documents in all other domains apart from news. With these four corpora, we can investigate out how the temporality ($COCA$ vs. $COHA$) or domain specificity ($COCA\_News$ vs. $COCA\_Except\_News$) of the background corpora influence the dynamic TF-IDF models for FSD.

[2] https://catalog.ldc.upenn.edu/LDC2006T18.

4.3 Update Frequencies

There is no standard update frequency for a dynamic TF-IDF model. Typically, updates are implemented so as to be less frequent than every 100 documents [7], as the update process is very expensive if the update frequency is higher than every 100 documents. In our experiments, we evaluate a range of update frequencies - specifically, every 100, 500, 1000, 10000 and 100000 documents, and also implement a static TF-IDF model as a baseline. The static model can be interpreted as a dynamic model where updates are extremely infrequent. For each update frequency we build TF-IDF models for all background corpora.

4.4 FSD Evaluation

Our implementation of FSD is based on the nearest neighbour algorithm, with the cosine distance algorithm adopted as the dissimilarity measure between documents. The preprocessing of data and the evaluation of FSD results are similar to our previous research [17]. In order to reduce the effect of useless terms and different term forms, we remove terms with very high and very low document frequency, i.e., stop words and typos, for all the background and target corpora, and subsequently stem all remaining terms. Aligning with previous research [19], comparisons are only implemented with the 2000 most recent stories for each candidate story. The output of each FSD model is a list of novelty scores, one for each story in the target corpus $TDT5$. Based on these outputs, the standard evaluation process for FSD is implemented by applying multiple thresholds to sweep through all the novelty scores. For each threshold, a missing rate and a false alarm rate are calculated; then for all thresholds, the missing and false alarm rates are used to generate a DET (Detection Error Tradeoff) curve [10], which shows the trade-off between the false alarm error and the missing error in the detection results. The closer the DET curve is to the origin, the better the FSD model is said to perform. Thus, from the DET curves we calculate Area Under Curve (AUC) for each FSD model, and the model with the lowest AUC is judged to be best.

5 Results and Analysis

Below we present our experimental results in Fig. 1, and analyse the impacts of different update frequencies and background corpora on the dynamic TF-IDF models for the FSD task.

5.1 Comparisons Across Different Update Frequencies

We begin by examining the FSD performance results as influenced by update frequency. From the results shown in Fig. 1, we firstly see a trend that for each background corpus, the dynamic TF-IDF models with high update frequencies, i.e., every 100, 500 and 1000 documents, outperform the static model and dynamic

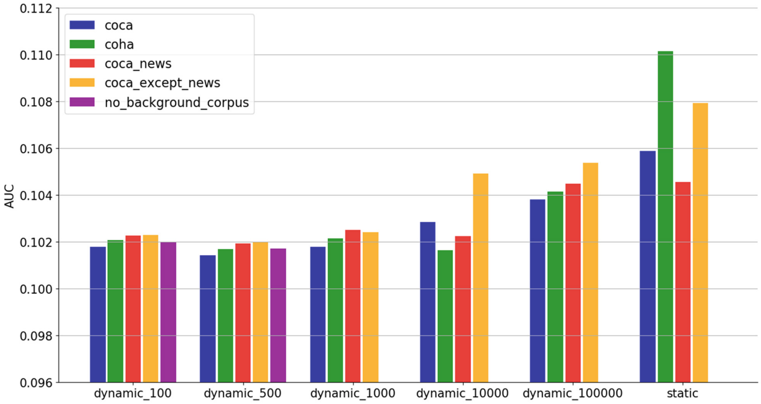

Fig. 1. Comparisons across different update frequencies and background corpora

models with low update frequencies, i.e., every 10000 and 100000 documents. As explained in Sect. 3, the dynamic models with high update frequencies focus more on the terms with large rough weights (i.e., the terms in Range B in Table 1), while the static models only focus on the terms with what we believe are realistic weights (i.e., the terms in Range A in Table 1). From this perspective, we can conclude that the terms with large rough weights play a more important role in FSD than the terms with realistic weights. Similarly, as the update frequency of a dynamic model becomes very low, the weights of most new terms are also well calibrated, and thus this dynamic model has fewer terms with rough weights, but more terms with realistic weights, which thus leads to poor FSD performance.

Secondly, we also find that for each background corpus, the FSD performance does not always improve but instead stays steady with a difference of less than 1% between models with a update frequency higher than every 1000 documents. One potential reason for this may be that as we increase the update frequency there are two counteracting processes with respect to rough weights: (a) a high update frequency means that new terms with rough weights are introduced into the model frequently, but (b) a high update frequency also means that the already-existing rough weights will themselves be updated incrementally and so may be smoothed frequently and so they don't stay rough for long. This is only our hypothesis of what might be happening.

5.2 Comparisons Across Different Background Corpora

We also make comparisons from the perspective of background corpora. From Fig. 1, it can also be seen that the differences caused by different background corpora are only noteworthy in the static model and dynamic models with low update frequencies. In the dynamic models with high update frequencies such as every 100, 500, 1000 stories, the influences are minor (less than 1%). This raises the possibility that models with high update frequencies are not affected by the choice of background corpus, in which case it may be possible to achieve good performance with a relatively small background corpus.

5.3 Comparisons Across Mini Corpora

Based on the results seen in Sect. 5.1 and 5.2, we might conclude that background corpora have very limited influence on dynamic models with high update frequencies in terms of FSD performance. The experiments thus validated our hypothesis about large-scale background corpora. However they have said little about the influence of very small background corpora. Given this, we can also propose the hypothesis that even a small background corpora can achieve as competitive a performance for FSD as a large-scale domain-related corpus.

To investigate the influence of corpus size at a more fine grained level, we extracted two small sets of documents, i.e., the first 500 stories and the last 500 stories, from each of the four background corpora to form eight very small background corpora. After that, eight dynamic TF-IDF models are built based on these corpora, and the update frequency was set to every 500 documents (the update frequency that leads to the best results in Sect. 5.1 and 5.2). The comparisons of FSD results are shown in Fig. 2 with the results of static models as the baseline.

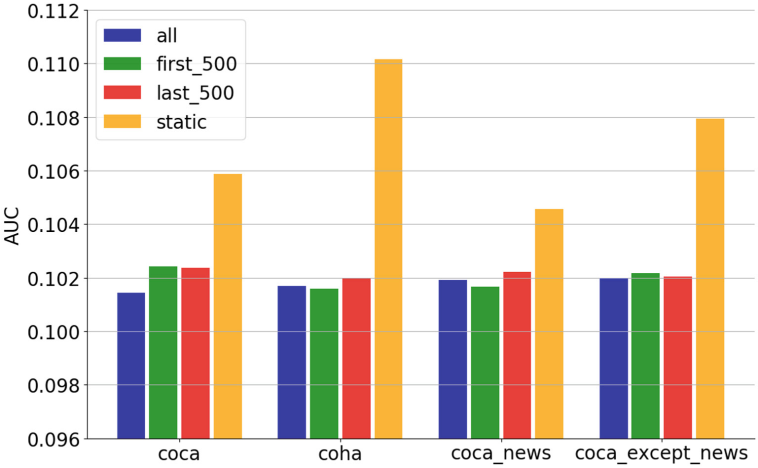

Fig. 2. Comparisons across mini background corpora with the update frequency set as every 500 stories

From the results, we can see that even based on background corpora that are quite different in scale, domain or collection time, there is no big difference (also within 1%) in the FSD results. Especially, the FSD result generated by the model based on the $First\_500\_COHA$ corpus is a little bit better than the full $COHA$ corpus, even though the stories in the $First\_500\_COHA$ corpus are collected around the year 1810 from various domains.

It is also worth mentioning that the corpus size 500 was not a crucial factor. It could have been 100, 1000 or any other number within this range. For further comparison, we also implemented pure dynamic TF-IDF models[3], i.e. the dynamic models that do not use any background corpus or with the corpus size set as 0, as shown with the tick "no_background_corpus" in Fig. 1. Unsurprisingly, the results show that the pure dynamic TF-IDF models with update frequency set as 100 or 500 do not make any big difference in FSD performance in comparison to the dynamic models based on any other background corpus with a similar update frequency, and this finding supports our conclusion that background corpora have very limited influence on dynamic models with high update frequencies in terms of FSD performance.

6 Conclusion

In this paper we empirically validated that the dynamic TF-IDF models with high update frequencies outperform the static model and the dynamic models with low update frequencies, and set out some factors that may explain this finding. However, a key element of these explanations is the observation that a high update frequency can result in new terms with relatively large weights being introduced into the TF-IDF representations. We also found that the FSD performance of dynamic models does not always improve but stays steady as the update frequency goes beyond some threshold, and that the background corpora have very limited influence on the dynamic models with high update frequencies in terms of FSD performance. Finally, we conclude that the best term vector model for FSD should be a dynamic model whose weights are initially calculated based on any small-size corpus but updated with a reasonable high frequency, e.g., for our scenario we found an update frequency of every 500 stories results in good performance.

Acknowledgement. The authors wish to acknowledge the support of the ADAPT Research Centre. The ADAPT Centre for Digital Content Technology is funded under the SFI Research Centres Programme (Grant 13/RC/2106) and is co-funded under the European Regional Development Funds.

[3] Actually, pure dynamic TF-IDF models should not be applied to the TDT task, because this specific task requires the detection to start from the very first story in the target data stream. However, as the first one story to be evaluated is on the 577^{th}, if we use the stories before it as the background documents to calculate the initial TF-IDF weights, there will be no influence on the detection results. Therefore, we apply pure dynamic models with a update frequency equal to or higher than every 500 stories to the task, but just for the analysis and the proof of our hypothesis.

References

1. Allan, J., Carbonell, J. G., Doddington, G., Yamron, J., Yang, Y.: Topic detection and tracking pilot study final report (1998)
2. Allan, J., Lavrenko, V., Malin, D., Swan, R.: Detections, bounds, and timelines: UMASS and TDT-3. In: Proceedings of Topic Detection and Tracking Workshop, pp. 167–174, February 2000. sn
3. Brants, T., Chen, F., Farahat, A.: A system for new event detection. In: Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 330–337. ACM, July 2003
4. Davies, M.: The corpus of contemporary american english as the first reliable monitor corpus of English. Liter. Linguist. Comput. **25**(4), 447–464 (2010)
5. Davies, M.: Expanding horizons in historical linguistics with the 400-million word Corpus of Historical American English. Corpora **7**(2), 121–157 (2012)
6. Goldberg, Y.: Neural network methods for natural language processing. Synthesis Lect. Hum. Lang. Technol. **10**(1), 1–309 (2017)
7. Kannan, J., Shanavas, A.M., Swaminathan, S.: Real time event detection adopting incremental TF-IDF based LSH and event summary generation. Int. J. Comput. Appl. 975, 8887
8. Leveau, V., Joly, A.: Adversarial autoencoders for novelty detection (2017)
9. Ma, J., Perkins, S.: Online novelty detection on temporal sequences. In: Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 613–618. ACM, August 2003
10. Martin, A., Doddington, G., Kamm, T., Ordowski, M., Przybocki, M.: The DET curve in assessment of detection task performance. National Inst of Standards and Technology Gaithersburg MD (1997)
11. Moran, S., McCreadie, R., Macdonald, C., Ounis, I.: Enhancing first story detection using word embeddings. In Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 821–824. ACM, July 2016
12. Petrović, S., Osborne, M., Lavrenko, V.: Streaming first story detection with application to Twitter. In: Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics, pp. 181–189. Association for Computational Linguistics, June 2010
13. Petrović, S., Osborne, M., Lavrenko, V. : Using paraphrases for improving first story detection in news and Twitter. In: Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 338–346. Association for Computational Linguistics, June 2012
14. Pimentel, M.A., Clifton, D.A., Clifton, L., Tarassenko, L.: A review of novelty detection. Sig. Process. **99**, 215–249 (2014)
15. Schölkopf, B., Platt, J.C., Shawe-Taylor, J., Smola, A.J., Williamson, R.C.: Estimating the support of a high-dimensional distribution. Neural Comput. **13**(7), 1443–1471 (2001)
16. Wang, F., Ross, R.J., Kelleher, J.D.: Bigger versus similar: selecting a background corpus for first story detection based on distributional similarity. In: International Conference on Intelligent Data Engineering and Automated Learning, pp. 107–116. Springer, Cham, November 2018
17. Wang, F., Ross, R. J., Kelleher, J.D.: Exploring online novelty detection using first story detection models. In: Proceedings of International Conference Recent Advances in Natural Language Processing 2019, pp. 1312–1320 (2019)

18. Wurzer, D., Lavrenko, V., Osborne, M.: Twitter-scale new event detection via k-term hashing. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pp. 2584–2589, September 2015
19. Yang, Y., Pierce, T., Carbonell, J.G.: A study on retrospective and on-line event detection (1998)