Enhancing First Story Detection using Word Embeddings

Sean Moran, Richard McCreadie, Craig Macdonald, Iadh Ounis
{firstname.secondname}@glasgow.ac.uk
University of Glasgow, UK

ABSTRACT

In this paper we show how word embeddings can be used to increase the effectiveness of a state-of-the art Locality Sensitive Hashing (LSH) based first story detection (FSD) system over a standard tweet corpus. Vocabulary mismatch, in which related tweets use different words, is a serious hindrance to the effectiveness of a modern FSD system. In this case, a tweet could be flagged as a first story even if a related tweet, which uses different but synonymous words, was already returned as a first story. In this work, we propose a novel approach to mitigate this problem of lexical variation, based on tweet expansion. In particular, we propose to expand tweets with semantically related paraphrases identified via automatically mined word embeddings over a background tweet corpus. Through experimentation on a large data stream comprised of 50 million tweets, we show that FSD effectiveness can be improved by 9.5% over a state-of-the-art FSD system.

1. INTRODUCTION

First Story Detection (FSD) is the task of identifying the first document that is related to a particular topic in a voluminous stream of documents. FSD has wide applicability across many disciplines ranging from the security industry to news reporting [11]. For example, a financial analyst may want the system to immediately flag up the first story that day relating to a stock of interest, so that they can make an informed buy/sell decision before the market shifts. The task of FSD was initially popularised by the Topic Detection and Tracking (TDT) initiative, which examined FSD over low volume newswire streams [2]. However, FSD has recently garnered considerable renewed attention with the availability of large-scale social media streams such as Twitter.

The most effective approaches to FSD have generally involved nearest neighbour search. Under this strategy, the most recent document in the stream is compared to a set of previous documents. If the nearest neighbour is sufficiently dissimilar to the current document, it is flagged as a first story. However, nearest neighbour search can often fail when working with social media data due to lexical variation. Indeed, consider the tweets 'Imogen lashing parts of England and Wales #bbc' and 'Storm hits Cornish Coast, waves of up to 19.1m (63ft) reported'. These tweets discuss the same event, but do not share any terms, hence both of these tweets would be emitted as first stories, causing the same event to be reported multiple times. Prior work [11] has proposed the use of paraphrases as a means to expand short social media posts with related terms from knowledge-bases such as WordNet [6]. However, while this improves FSD performance, the gain is much smaller than that observed when the same technique is applied to newswire data, likely due to lexical mismatch between the knowledge bases and social media [11]. Hence, an alternative paraphrase expansion approach would be advantageous.

Word embeddings have been proposed as a method for producing more effective word representations. In the Word2-vec model [9], a shallow neural network learns dense real-valued vectors for each word in the vocabulary by attempting to maximise the probability of seeing that word within a fixed context window. Word embeddings have shown to be an effective means to improve a variety of tasks that involve the representation of text items in a vector space, such as text classification [7]. In this paper, we propose to leverage word embeddings to enhance the representation of social media posts for the purposes of FSD. In our method, we use a background corpus of tweets to learn a set of word embeddings. These word embeddings are then used to find semantically related terms with which to expand each tweet. We conjecture that by expanding posts in this manner, we will be able to reduce cases where textually distinct posts about a single event are erroneously reported as first stories.

The primary contribution of this paper is a simple and effective method for using word embeddings (WE) to automatically compute good paraphrase pairs for the purposes of mitigating the problem of vocabulary mismatch in FSD. We show that our approach can enhance FSD effectiveness by approximately 9.5% over a state-of-the-art FSD model without expansion [11].

2. RELATED WORK

The task that we examine in this paper is *first story detection* (FSD). FSD is an application of nearest neighbour search as a means to identify novel textual documents. Each new document in the stream is compared to a set of previously observed documents. If the nearest neighbour to the

SIGIR '16, July 17-21, 2016, Pisa, Italy

DOI: http://dx.doi.org/10.1145/2911451.2914719

current document is sufficiently dissimilar then the current document is considered novel and hence emitted as a first story for a new topic. FSD was initially examined over low volume news article streams [2], but has recently been extended for use over high volume social media streams [10]. In particular, traditional implementations of nearest neighbour search are too computationally expensive to apply to high volume streams [10]. Hence, recent works have focused on making the nearest neighbour search process scalable while minimising the loss in effectiveness [5, 10]. For instance, Petrovic *et al.* [10] proposed the use of a locality sensitive hashing (LSH) algorithm [4, 8] to perform FSD in bounded time and memory.

Lexical variation is a significant barrier to achieving effective FSD performance over social media streams, as it can result in the same event being reported multiple times. State-of-the-art FSD approaches use paraphrase expansion to overcome this issue [11]. A paraphrase expresses the meaning of a written piece of text using different words. There are three common levels of paraphrasing: lexical paraphrases (single word paraphrases); phrasal paraphrases (multi-word paraphrases); and sentential paraphrases (sentence-length paraphrases). Petrovic *et al.* [11] used lexical paraphrases extracted from WordNet [6], Microsoft Research (MSR) paraphrase tables [12] and syntactically constrained paraphrases [3] for document expansion. They showed that text expansion using these paraphrases were effective over newswire, but performance improvements were much smaller when applied over tweets. We propose an alternative approach based on learning word embeddings directly from Twitter data.

3. IMPROVING FSD WITH PARAPHRASES

An FSD system must produce a novelty score for each tweet $\mathbf{x}_i$ indicating the likelihood that the tweet describes a first story, i.e. a topic not previously described by an earlier tweet observed within the stream. High novelty scores indicate a greater likelihood that the tweet is reporting a first story. We propose a novel approach to improve the estimation of these novelty scores by expanding each tweet $\mathbf{x}_i$ using lexical paraphrases mined via word embeddings from a background corpus of tweets. We aim to answer the following research question:

RQ-1: Can word embeddings be used to automatically mine lexical paraphrases that are effective at mitigating the problem of lexical mismatch for FSD in Twitter?

3.1 Expanding tweets using paraphrases

To explore **RQ-1** we build upon the Locality Sensitive Hashing (LSH)-based FSD model of [11]. Their model computes the cosine similarity between tweets, and uses that score as a measure of novelty. To drastically reduce the number of required comparisons from $\mathcal{O}(N)$ to $\mathcal{O}(1)$, LSH-FSD applies LSH to bucket the tweets into the buckets of L hashtables. The cosine similarity is only computed between tweets that collide in the same bucket as the current tweet in the stream. To compute the hashcodes that index into the hashtable buckets, LSH fractures the input feature-space with a set of K randomly sampled hyperplanes with normal vectors $\left\{\mathbf{u}_k \in \Re^V\right\}_{k=1}^{K}$, where V is the vocabulary size. The K-bit binary hashcodes $\mathbf{b}_i$ for each tweet $\mathbf{x}_i$ can be computed simply by determining on which side of the hyperplanes the tweet feature vector falls. This operation reduces into K dot products, followed by sign thresholding, as illustrated in Equation (1):

$$\mathbf{b}_i = F(\mathbf{x}_i) = sgn\left[\mathbf{u}_1^\intercal \mathbf{x}_i \dots \mathbf{u}_K^\intercal \mathbf{x}_i\right] \quad (1)$$

where *sgn* denotes the sign function, $sgn(x) = 1$ if $x > 0$, and 0 otherwise. It has been shown that this procedure causes the Hamming distance between the binary hashcodes to correlate with the cosine similarity computed on the tweet TF-IDF feature vectors [8]. This means that tweets colliding in the same hashtable bucket are most likely to have a high cosine similarity, and therefore to be nearest neighbours. In our work we are interested in the modified cosine similarity given by Equation (2):

$$cos_Q(\mathbf{x}_i, \mathbf{x}_j) = \frac{\mathbf{x}_i \mathbf{Q} \mathbf{x}_j}{\sqrt{\mathbf{x}_i^\intercal \mathbf{Q} \mathbf{x}_i}\sqrt{\mathbf{x}_j^\intercal \mathbf{Q} \mathbf{x}_j}} \quad (2)$$

where $\mathbf{Q} \in \{0,1\}^{V \times V}$ is a binary paraphrase indicator matrix. In this matrix, $Q_{ij} = 1$ if words w_i, w_j are considered paraphrases (e.g. blast $\leftrightarrow$ explosion, source $\leftrightarrow$ informant), and $Q_{ij} = 0$, otherwise. In comparison to using unmodified cosine similarity, Equation (2) will assign higher similarity scores between tweets that share paraphrases from $\mathbf{Q}$. Petrović et al. [11] show that LSH can be adapted to preserve this modified cosine similarity in the resulting binary hashcodes simply by pre-multiplying the tweets with the square root of $\mathbf{Q}$ (Equation (3)):

$$\mathbf{b}_i = F(\mathbf{x}_i) = sgn\left[\mathbf{u}_1^\intercal (\mathbf{Q}^{1/2}) \mathbf{x}_i \dots \mathbf{u}_K^\intercal (\mathbf{Q}^{1/2}) \mathbf{x}_i\right] \quad (3)$$

The term $\mathbf{Q}^{1/2}\mathbf{x}_i$ can be interpreted as mapping the tweet $\mathbf{x}_i$ into a new inner product space defined by the paraphrase matrix $\mathbf{Q}$ [11]. The hashcodes resulting from Equation (3) can be used to index the tweets into hashtable buckets. Tweets colliding in the same bucket should have a high likelihood of being similar, i.e. discussing the same event.

3.2 Automatically mining lexical paraphrases

Our primary contribution in this paper is a new way of automatically computing the lexical paraphrase matrix $\mathbf{Q}$ using word embeddings. There has been an extensive amount of prior research that has shown that the cosine similarity between word embeddings is correlated with the semantic relatedness between the corresponding words [9]. In our work we make use of this property by deeming two words to be lexical paraphrases if the cosine similarity between their word embeddings is sufficiently high. More concretely, to automatically construct the lexical paraphrase matrix we follow a simple three-step procedure:

Learn Word Embeddings: Learn a set of word embedding vectors using Word2vec [9] on a background corpus containing the same type of documents that are to be expanded. In our case, we use a random sample of tweets crawled from a different time period to train our word embedding vectors.

Word Filtering: Note that the majority of the words within the background corpus will not be useful for expansion, since they are either too general to make effective expansion terms (e.g. words like 'about' or 'news') or are specific Twitter terminology from the period of that corpus (e.g. '#eusew15'). Hence, we filter the words considered within the word embedding corpus to only words that are likely to be informative based on a series of public word lists (see Section 4).

Word Similarity Computation: Finally, we compute the cosine similarity $s_{ij} \in \Re$ between the embeddings of every word $\mathbf{w}_i \in \Re^D, \mathbf{w}_j \in \Re^D$, where D is the word embedding dimensionality, and threshold the resulting similarities using a threshold $\theta \in \Re$. Similarities are only computed between words in the same word list. Note that the cosine similarity we use here is the standard cosine similarity, not that given in Equation (2).

Words pairs with a similarity above θ are retained and used to construct the paraphrase matrix $\mathbf{Q}$. We use a binary matrix and set element Q_{ij} to 1 if the corresponding similarity $s_{ij} \geq \theta$, and $Q_{ij} = 0$ otherwise. We note that our method only relies on word embeddings and the availability of word lists to construct the paraphrase matrix. Given the wide availability of standard word embedding software and word lists for most languages, both resources are significantly easier to obtain than manually curating lexical paraphrases, for example by creating WordNet synsets. WordNet is an expensive resource that was relied upon by the LSH-FSD system of [11] to obtain high FSD effectiveness.

4. EXPERIMENTAL SETUP

Datasets: Our experimental testbed is the collection of over 50 million tweets introduced by [11]. The tweets in the dataset were sampled from July to September 2011, with a subset of the tweets manually labeled as being on-topic for one of 27 events (e.g. 'Amy Winehouse dies' or 'Earthquake in Virginia'). To learn the word embeddings that we use to calculate the semantic distance between terms (which we then use to select terms with which to expand each tweet), we use a background set of tweets from a different time period. In particular, we use a random sample of 451 million tweets crawled using Twitter Streaming API from the period of the 1 January 2015 to 30 June 2015.

Tweet text pre-processing: When working with Twitter data, pre-processing applied to the terms can have a marked impact on overall effectiveness. To this end, we explore the effect of the term processing techniques used by [10]. Specifically we explore Porter and Krovetz stemming in addition to *Twitter specific text pre-processing* that was reported to be effective in [11], namely ignoring links, @-mentions and treating hashtags as normal words (i.e. removing the leading # character).

Word Dictionaries: Given that our evaluation dataset is largely US-centric, we experiment with a set of freely available English term word lists. These word lists can be downloaded from http://icon.shef.ac.uk/Moby/. In particular, we experiment with five word lists representing different types of information, namely: common male names (denoted by M) (3,897 words), common female names (F) (4,946 words); place names in the United States (P) (10,196 words); commonly misspelt words (S) (366 words); and common dictionary words (W) (74,550 words).

Metrics: We use the widely accepted normalised Topic Weighted Minimum Cost (C_{min}) [2, 5, 10, 11]. C_{min} is a linear combination of miss and false alarm probabilities and is computed across all possible threshold values on the first story confidence score. This allows a comparison of different methods based on a single value metric. For C_{min} a smaller number is better. We compare the significance of the results by performing a paired t-test over the 27 per topic C_{min} scores.

Training and Parameters: We use the popular Word2Vec tool[1] to train our word embeddings with dimensionality D= 200. The threshold θ on the cosine similarity between Word2-vec embeddings is a tunable parameter of our model. In practice we use a different value θ_1 for the four smaller word dictionaries (M,F,P,S) and a separate value of θ_2 for the much larger common dictionary words list (W). We optimise thresholds $\{\theta_1, \theta_2\}$ jointly by conducting a parameter sweep over $\theta \in \{0.0, 0.1, \ldots, 0.9, 1.0\}$ on a *held-out training dataset*, minimising for C_{min}. The training dataset for parameter tuning was entirely independent of our test dataset, and consisted of 806,342 tweets in total. 7,512 of these tweets were manually labelled as on-topic for one of 10 events. The training dataset events occurred in 2011 and include, amongst others, the death of Steve Jobs, the Seoul floods, and the Indiana State Fair stage collapse[2]. We use the same LSH-FSD system parameters as [10, 11], namely K=13 hashcode bits and L=70 hashtables, the hashing trick is used with a pool of size 2^{18} and we select 2000 tweets and a back-off threshold of b_t=0.6 for the variance reduction step.

Baselines: We compare our method to two state-of-the-art FSD models as follows. First, UMass [2], is a system that produced state-of-the-art performance in the TDT2 and TDT3 competitions [1] and which has since formed the de-facto baseline for comparison in subsequent research on first story detection [5, 10, 11]. UMass uses k-nearest neighbour clustering and an inverted index to identify first stories in the tweet stream. Second, LSH-FSD [10] is a streaming FSD model that was shown to be both more effective and more efficient than UMass. LSH-FSD employs Locality Sensitive Hashing (LSH) to hash tweets into buckets. Tweets colliding in the same bucket are tested for similarity using the cosine similarity. If the similarity is low enough a tweet is deemed to report a first story (Section 3.1).

5. RESULTS

In this section we present a series of results designed to answer **RQ-1**. Table 1 reports C_{min} FSD effectiveness for the baseline FSD systems and our proposed approach that uses paraphrases obtained from word embeddings. In particular, the first column highlights the FSD strategy - from inverted indexing (UMass) to hashing (LSH). The second column indicates the pre-processing applied to each tweet in terms of stemming or Twitter specific text processing [11]. The third column denotes any tweet expansion with various paraphrase sources. Importantly, it is not possible to reproduce all approaches from the literature due to efficiency constraints (UMass) and lack of access to the paraphrase resources used in [11]. Hence, we provide C_{min} scores that have been reported in the literature (column 4) and, where possible, our re-implementation of the approach (column 5). The bottom row of Table 1 reports the performance of our approach using the five word lists to filter the word embeddings.

From Table 1, we observe the following: firstly, comparing the reported performance of LSH using Twitter specific pre-processing to our implementation of this approach we see that performances are very similar (reported: 0.694 vs. implemented: 0.705). Furthermore, we see almost identical performance between the LSH-FSD systems using Porter

[1]https://code.google.com/archive/p/word2vec/

[2]Annotations provided by [5, 10] and available on request.

Approach	Pre-Processing	Paraphrase Expansion	Reported C_{min}	Implemented C_{min}
UMass	Twitter (Krovetz)	None	0.798	-
LSH	None	None	-	0.890
LSH	Twitter	None	0.694	0.705
LSH	Twitter (Porter)	None	0.756	0.745
LSH	Twitter	MSR	0.739	-
LSH	Twitter	Syntactic	0.729	-
LSH	Twitter	WordNet	0.679	-
LSH	Twitter	Word Embeddings	-	**0.638▲**

Table 1: C_{min} results over the Twitter testing dataset that have either been reported in the literature (Reported) or implemented by the authors (Implemented). Lower C_{min} scores are better. Statistically significant improvements (paired t-test signed rank test $p < 0.05$) over the (Implemented) LSH baseline (with Twitter pre-processing) are denoted ▲.

stemming (reported: 0.756 vs. implemented: 0.745). We therefore confirm the finding of [11] that stemming hurts FSD performance in Twitter. Overall, since our implemented results are similar to the reported results it is reasonable to compare results between the reported column and implemented column of Table 1.

Next, we present the FSD results arising from using our proposed data-driven method for generating lexical paraphrases. As explained in Section 4, we first tune the model parameters (i.e. thresholds on the cosine similarity scores) θ_1, θ_2 on the training dataset, obtaining a minimum C_{min}= 0.458 with θ_1=0.2, θ_2=0.8 (Figure 1). We then fix these parameters at the optimal values found on the training dataset and run our model once on the testing dataset, reporting the result. Comparing our proposed approach with the LSH-FSD baseline we observe that FSD performance can be improved by a statistically significant margin of 9.5% (baseline: 0.705, WE: 0.638). To answer our research question (**RQ-1**), we can conclude that using word embeddings to obtain lexical paraphrases can improve FSD effectiveness on Twitter data.

Finally, comparing the performance of our proposed approach to the LSH-FSD system of [11] using the MSR, syntactic and WordNet curated paraphrases, we make two findings. Firstly, our automatic means of computing paraphrases using word embeddings markedly improves FSD performance, whereas the automatically curated MSR and syntactic paraphrases of [11] actually were found to hurt FSD performance (e.g. MSR: 0.739 vs. WE: 0.638). Secondly, we observe a relative 6.0% gain in performance over tweet expansion using WordNet (WordNet: 0.679 vs. WE: 0.638), a particularly encouraging finding given that WordNet is manually curated whereas our paraphrases are automatically generated.

6. CONCLUSIONS

In this paper we showed how the similarity between dense real-valued word embeddings could be used to mine effective lexical paraphrases in an entirely automatic manner. We used the obtained paraphrase pairs to mitigate the problem of lexical variation on the task of first story detection (FSD) over Twitter data. Tweets were expanded with related terms, allowing the FSD system to ignore related tweets that were seen earlier in the stream but used different, related words. Through evaluation on a standard Twitter FSD dataset, we showed that FSD effectiveness can be improved by a statistically significant margin over a state-of-the-art FSD system. Furthermore, this approach is more effective than an approach that expands tweets using paraphrases obtained from WordNet, a linguistic resource that is very expensive to construct. For future work, we are interested in extending our approach to harvest paraphrases in other languages, such as Arabic, for use in a multilingual streaming FSD system.

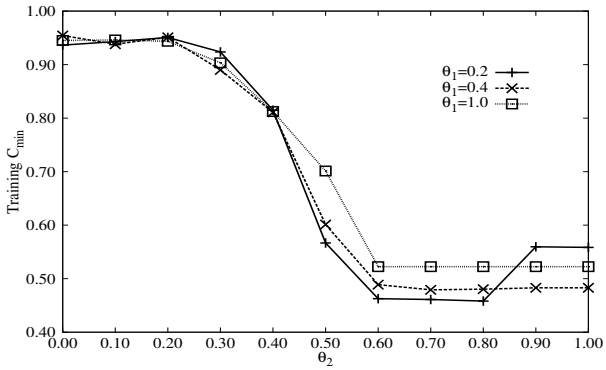

Figure 1: Tuning the θ_1, θ_2 threshold parameters on the training dataset. Filtering embeddings using all five word lists leads to the lowest C_{min}=0.458 at $\theta_1 = 0.2, \theta_2 = 0.8$.

7. ACKNOWLEDGMENTS

We acknowledge support from both the integrated Multimedia City Data (iMCD) project within the ESRC-funded Urban Big Data Centre (ES/L011921/1) and the EC co-funded SUPER (FP7-606853) project.

8. REFERENCES

[1] J. Fiscus Overview of of the TDT 2001 evaluation and results. In *Proc. TDT*, 2001.
[2] J. Allan. Introduction to topic detection and tracking. In *Proc. TDT*, 2002.
[3] C. Callison-Burch Syntactic constraints on paraphrases extracted from parallel corpora. In *Proc. EMNLP*, 2008.
[4] P. Indyk and R. Motwani. Approximate nearest neighbors: Towards removing the curse of dimensionality. In *Proc. STOC*, 1998.
[5] D. Wurzer, V. Lavrenko, M. Osborne Twitter-scale New Event Detection via K-term Hashing. In *Proc. EMNLP*, 2015.
[6] G. Miller. WordNet: a lexical database for English. Communications of the ACM, 38 (11), 1995.
[7] R. Lebret, and R. Collobert. N-gram-Based Low-Dimensional Representation for Document Classification. In *Proc. ICLP*, 2015.
[8] M. Charikar. Similarity Estimation Techniques from Rounding Algorithms. In *Proc. STOC*, 2002.
[9] T. Mikolov, K. Chen, G. Corrado and J. Dean. Efficient estimation of word representations in vector space. In *Proc. ICLR*, 2013.
[10] S. Petrović, M. Osborne, and V. Lavrenko. Streaming first story detection with application to Twitter. In *Proc. NAACL*, 2010.
[11] S. Petrović, M. Osborne and V. Lavrenko. Using paraphrases for improving first story detection in news and Twitter. In *Proc. NAACL*, 2012.
[12] C. Quirk, C. Brockett, and W. Dola. Monolingual machine translation for paraphrase generation. In *Proc. EMNLP*, 2004.