Learning Class-specific Word Representations for Early Detection of Hoaxes in Social Media

Arkaitz Zubiaga
University of Warwick, Coventry, UK
a.zubiaga@warwick.ac.uk

Abstract—As people increasingly use social media as a source for news consumption, its unmoderated nature enables the diffusion of hoaxes, which in turn jeopardises the credibility of information gathered from social media platforms. To mitigate this problem, we study the development of a hoax detection system that can distinguish true and false reports early on. We introduce a semi-automated approach that leverages the Wikidata knowledge base to build large-scale datasets for veracity classification, which enables us to create a dataset with 4,007 reports including over 13 million tweets, 15% of which are false. We describe a method for learning class-specific word representations using word embeddings, which we call multiw2v. Our approach achieves competitive results with F1 scores over 72% within 10 minutes of the first tweet being posted, outperforming other baselines. Our dataset represents a realistic scenario with a real distribution of true and false stories, which we release for further use as a benchmark in future research.

Index Terms—social media, hoax detection, breaking news.

I. INTRODUCTION

SOCIAL media platforms such as Twitter and Facebook are increasingly being used by the general public to follow the latest news [20], [11] and by journalists for newsgathering [6], [28]. The fact that anyone can post and share content in social media without moderation enables decentralised production of citizen journalism with an unprecedented detail of report [4]. However, the unmoderated nature of social media also leads to the production and diffusion of hoaxes [15], [1], which exacerbates the credibility of social media as a source for news consumption.

Research in automated detection of misinformation in social media has indeed increased in recent years [21], [27]. Researchers have assessed the capacity of average people to identify reports that are inaccurate, finding that their performance leaves much to be desired [10]. This reinforces the need to develop automated systems for hoax detection, however existing work has largely limited to post-hoc classification of reports as true or false, which means that reports can only be classified hours after they are first released. Research in performing early classification of reports by their truth value is very scarce. An important challenge that hinders the development of early hoax detection systems is the dearth of suitable datasets. Datasets are usually produced by first identifying lists of fake reports. These are then completed by including news reports from other sources to have balanced datasets with fake and real news reports. This, however, is not necessarily representative of a real scenario of incoming reports. This work aims to overcome this issue by introducing a novel approach for generating a representative dataset with accurate news and hoaxes.

To develop a representative data collection process, we look into death reports of celebrities circulating in social media. Death reports are known to be riddled with hoaxes [1] users frequently making up the death of celebrities, making them viral as if they were real reports. We match these death reports in social media with the person’s entry in the Wikidata knowledge base. Doing this offline for the dataset generation enables us to easily determine if the person had actually died when it was reported or not. This semi-automated and straightforward dataset generation process enables us to create a large-scale dataset with over 13 million tweets associated with 4,007 death reports over the course of three years.

In this paper we make the following key contributions:

• We propose a novel approach that leverages the Wikidata knowledge base to build a large-scale dataset for early detection of hoaxes in social media.

• We propose a classification approach that uses class-specific word representations using word embeddings for effective detection of hoaxes. This approach is possible thanks to the semi-automated approach for generation of large-scale datasets, which enables large sets of training data to be available for building the model of our classifier and for learning word representations.

• We look into the use of sliding windows which enables us to leverage the most recent tweets in the timeline associated with a report, instead of the entire timeline.

Our data collection approach enables us to produce a dataset with 4,007 death reports including over 13 million tweets, of which 15% are fake. Our experimentation shows the effectiveness of our proposed approach for building class-specific word representations, achieving F1 scores over 72% within just 10 minutes of the first report being posted, and outperforming other baselines. Our experiments also show that the use of sliding windows does not help improve the results. The release of our dataset and trained word embedding models will enable further research in veracity classification using a benchmark scenario.

II. RELATED WORK

A. Veracity Classification

Previous work on veracity classification has used different social media platforms including Twitter [23] and Sina Weibo

1http://www.snopes.com/tag/celebrity-death-hoaxes/
However, most of this work has performed post-hoc classification of reports as true or false [8], [14], [22], which means that they need to observe the entire development of a story before classifying it. This may imply hours or even days of delay by the time a story can be classified. Our objective here instead is to aim for early classification of stories, with the ultimate goal of detecting hoaxes early on.

Research looking into either real-time or early detection of hoaxes is scarce. [13] use a set of features including user metadata and propagation structure to verify stories within hours of being posted for the first time. They show competitive performance with the use of both feature sets 72 hours after the story was first posted. Another approach is presented by [19], combining hashtags and links as features to determine the veracity of reports. They report results between 1 and 10 hours, with results increasingly improving over time. While both of these are clever approaches that are worthwhile considering, neither of their systems was publicly released and the features they use are hardly reproducible for the reader. Others have taken a different approach by using stance classifiers [17]. Instead of using a classifier that directly outputs one of true or false given a report as input, they try to determine the stance that each social media post expresses with respect to a report, such as supporting, denying, querying or commenting. They then propose to aggregate the different stances to determine the likely veracity of a report. While this is a sensible approach, it also requires a significant amount of posts to be observed in order to aggregate the different stances, which may impede early determination of report veracity.

Research in veracity classification has been largely limited by the dearth of proper datasets. As [21] stated, development of a dataset annotated for veracity is very challenging, as judgments from professionals are needed to carefully verify and subsequently annotated stories. As shown by previous research [10], average users struggle to distinguish true and false stories, and it is therefore not a suitable task to be performed through crowdsourcing, requiring professional input instead. As a result, few datasets have been produced, and most of these datasets are created by first collecting false stories, and then completing the datasets with randomly picked true stories [12], [13]. The use of different methodologies for collecting false and true stories is however not ideal as it will inevitably differ from a real scenario.

In this work, we describe a novel approach for semi-automated dataset generation, which removes the sampling bias as verification of larger sets of instances is possible through the use of Wikidata as an external source. Likewise, our approach enables collections of both true and false stories by following the same methodology, leading to a representative dataset.

### B. Learning Class-specific Word Representations

Class-specific word representations have been found to be useful for different classification tasks, as is the case with the use of Brown clusters to build class-specific language models [3]. Brown clusters have been successfully used by researchers for training word representations [24], natural language processing tasks such as dependency parsing [9] or for building class-specific language models [2], among others. As a state-of-the-art approach for semantic word representation, here we make use of word embeddings [16]. We propose to train and leverage class-specific word embeddings to learn the patterns of each class in the training data. The difficulty to achieve this generally lies in the necessity for large-scale annotated datasets that have large numbers of instances for each class. Our semi-automated approach for building large-scale annotated datasets enables to have large collections of data to train class-specific word embeddings.

### III. Dataset

Our data collection methodology is semi-automated, involving little and easy human input, which enabled us to collect a large-scale dataset. The dataset generation process consists of three steps: (1) data collection, (2) linking to Wikidata, and (3) data annotation.

#### A. Data collection

We first perform keyword-based collection of tweets from Twitter. We use ‘RIP’ as a keyword that is largely associated with death reports. Twitter’s results are not case sensitive, so we collect all tweets including the keyword and remove those that are not upper-cased in a later stage. We perform the collection of tweets containing the keyword ‘RIP’ for a period of three years between January 1, 2012 and December 31, 2014. This longitudinal data collection led to a total of over 94.2 million tweets.

#### B. Linking to Wikidata

As we completed the collection of tweets at the end of 2014, we downloaded a dump of Wikidata [25] in January 2015, which is a structured knowledge base that includes, among others, an extensive database of notable people, in part extracted from Wikipedia but also completed by volunteer contributors. We used its API to download all entries corresponding to people leading to a collection of 1,136,543 different people. Each of these entries includes the fields shown in the following example:

```
{"id":"8023", "name":"Nelson Mandela", "birth":{"date":"1918-07-18","precision":11}, "death":{"date":"2013-12-05","precision":11}, "description":"former President of South Africa, anti-apartheid activist", "aliases\["Nelson Rolihlahla Mandela","Mandela","Madiba"]}
```

We are interested in most of these features for our research, but especially in the name and aliases, which we use to identify mentions of people in our ‘RIP’ tweets, and also the death date, which indicates if a person is still alive or has died on a particular date. Note that birth and death dates have a precision value associated, which refers to the granularity of the date. A value of 11 implies the date is accurate at the day

To identify entries that are about people, we looked for entries with the property “P569”, which refers to “date of birth” and is therefore indicative of an entry belonging to a person: https://www.wikidata.org/wiki/Property:P569
level. The standard for contemporary people is for this value to be 11. Year and month-level precisions are occasionally given for people in earlier centuries. We use the Wikidata knowledge base to look for mentions of contemporary people in our Twitter dataset, and so the lack of precision for ancient people does not have an effect in our case.

Having the collection of ‘RIP’ tweets and the entries for people on Wikidata, we look within the tweets for mentions of names (and aliases) of people in the Wikidata knowledge base, e.g. tweets containing ‘RIP Nelson Mandela’. To do so, as a first step, since the keyword search on Twitter is case insensitive, we removed all occurrences where the keyword ‘RIP’ was not completely upper-cased. We then looked for tweets where the keyword ‘RIP’ was followed by one of the person names (or aliases) in Wikidata. We do this for all the tweets and keep the instances in which the name of a person is mentioned at least 50 times in a day. Removing instances with fewer than 50 tweets reduces noise from spam tweets that did not go viral, and makes the manual annotation (which we explain below) more manageable. Note that this process can also identify numerous instances of mentions of the same person, i.e., being reported dead in social media more than once within the time frame of our study between 2012 and 2014. Consecutive days mentioning the same person are considered part of the same death instance, while we only consider a new instance when there is at least one day gap between mentions. This process led to a dataset with 4,007 death reports pertaining to 3,066 different people. The total number of tweets associated with these reports amounts to 13,302,600.

C. Annotation

At this stage we have 4,007 death reports linked to Wikidata pages. An automated comparison of the date of these reports with respect to the death date in the Wikidata pages is largely indicative of the story being true or false and facilitates the annotation work, but still, some manual work is needed to validate it. Another issue is that some names are ambiguous, and they match different Wikidata pages; we manually annotate which Wikidata page the death reports belong to when these are ambiguous.

To perform this annotation easily, we developed an annotation tool that visualises the stream of tweets associated with a report, along with a form that enables the annotation (see Figure 1). After reading through the tweets in the timeline on the left, the annotator can then use the form on the right to perform the annotation. The annotation consists of two tasks: (1) selecting the Wikidata entry that the death report is about, and (2) selecting the appropriate category for the death report, i.e. real death, fake death or commemoration. The annotation is straightforward as the death date (or lack thereof) makes the manual categorisation very easy. The example in Figure 1 shows a hoax reporting the death of Justin Bieber, which can be identified from the Wikidata entry having a death date of “0,” which indicates that the person is alive according to Wikidata. The annotation is even easier for real deaths, as the annotation tool automatically detects a match between the date of the tweets and the date of the death of one of the Wikidata entries.

The annotation of the 4,007 death reports in our dataset led to the following distribution: 2,301 real deaths, 1,092 commemorations and 614 fake deaths. Table I shows the statistics of the dataset. While the categories are imbalanced, this still shows that fake deaths represent a significant proportion of all reports (15.3%) and need to be tackled to avoid their diffusion. The skewed distribution of categories presents in turn an additional challenge for the classification task.

| Veracity       | Instances | Tweets    |
|----------------|-----------|-----------|
| Real           | 2,301     | 9,131,976 |
| Commemoration  | 1,092     | 526,588   |
| Fake           | 614       | 643,432   |
| Total          | 4,007     | 10,301,996|

TABLE I

DISTRIBUTION OF LABELS AND TWEETS IN THE DATASET.

It is worthwhile emphasising that the manual annotation is fairly easy thanks to the linking to Wikidata that provides context to determine the correct label. However, the automated classification of reports we performed in this work is much more challenging as it deals with early detection of hoaxes, i.e. when the Wikidata page is not yet necessarily updated.

IV. HOAX DETECTION

In this section we describe the objective of the hoax detection task, and we provide details of the features and experiment settings that we use for our work.

A. Task Description

The hoax detection task consists in identifying emerging reports that are false. In our experiments, we aim to identify the death reports that have been fabricated, i.e. reporting cases of deaths that have not actually happened. We formally define the death hoax detection task as that in which a supervised classifier has to determine which of the following three categories a new incoming reporting belongs to: \( Y = \{\text{real, commemoration, fake}\} \). We use three categories as we distinguish cases of fake reports, where a death has been fabricated, real reports, where a death report has indeed recently happened, and commemorations, where a past death is being remembered.
B. Classification Features

We use three different types of features, including two features that are widely used in previous work (social features and textual features), as well as our proposed class-specific word representations. Additionally, we propose two different combinations of those features. To simulate the task of early detection of hoaxes, we perform experiments at different points in time. Experiments performed in time \( t \) will generate the features only from tweets posted before that time. The feature sets we use for the experiments are as follows:

- **Social features (social):** We use a set of 16 features that refer to the reputation of the users participating in a report and to diffusion patterns. Please see the appendix for more details of these features.

- **Textual features using word embeddings (w2v):** As a state-of-the-art word representation approach, we use Word2Vec embeddings [16] to represent the content of the tweets associated with a report. The model we use for the embeddings was trained from the entire collection of tweets in the training set, i.e. all the 2012 and 2013 tweets. We represent each tweet as the average of the embeddings for each word, and finally get the average of all tweets.

- **Class-specific word representations (multiw2v):** The same word can have different meanings depending on the category in which it is used. For instance, ‘RIP’ usually refers to ‘Rest In Peace’ or ‘Requiescat In Pace’ when it is used along with a real death, but it can mean ‘Really Inspiring Person’ when used as a hoax. This can be hard to distinguish even for humans as the word is exactly the same, but it can be statistically modelled using word embeddings. Provided that we have large-scale training data, we propose to train different word embedding models for each class, so that each model learns the vocabulary of that class. We build three different collections from our training set, each belonging to tweets from one of the categories, and train a separate word embedding model from each of the three collection, so that we have a word embedding for real reports, another one for fake reports and a third one for commemorating reports. Having three different word embedding models (real, fake, commemoration), we then create three different vectors, each of which is created as above, however using a different word embedding model. Finally, we combine all three vectors by concatenating them into a single vector. Our proposed model, which we call multiw2v, enables characterisation of reports with respect to each class in the dataset.

- **Social and textual combined (social+w2v):** We combine social and word embedding features by concatenating vectors.

- **Social and class-specific representations combined (social+multiw2v):** We combine social features and class-specific word representations by concatenating vectors.

C. Experiment Settings

Given that the objective of our experimentation is to find out what features perform best for early detection of hoaxes, assessing the performance of our proposed class-specific word representations, we first tested different classifiers: Support Vector Machines, Random Forests, Logistic Regression and Naive Bayes. We found the Logistic Regression classifier [18] to perform significantly better than the rest of the classifiers, and so for the sake of clarity and space we show results for this classifier in the rest of this article. Specifically, we use a multinomial logistic regression classifier [19] which relies on the Principle of Maximum Entropy [20] to determine the category for each item in the test set. It is a supervised classifier that first builds a model using a training set.

To split the training and test sets, we simulate a realistic scenario where a model is trained from past reports to then classify future reports. As our dataset includes data for 2012, 2013 and 2014, we use the first two years for training and the last year for testing. Given that both the training and test sets are large, we experiment with different subsets of each. On the one hand, we train the model from different subsets of the training set to assess the impact of the size of the training data. On the other hand, we split the test set into 10 randomly generated subsets to experiment using a 10-fold cross-validation setting, i.e. the trained model is tested on each of the 10 folds, ultimately averaging the performance across all of them.

We report performance scores of different classifiers using macroaveraged F1 scores, i.e. averaged F1 scores for the three categories, where the F1 score for a category equates to the harmonic mean between the precision and the recall.

V. Classification Results

We first present a comparison of the different features under study, delving into results by category. Then, we explore the use of sliding windows for the classification.

A. Comparison of Features

We first compare the five sets of features and combinations of features we described above. We show results for classification experiments in different points in time including 0 (only the first tweet posted), 5, 10, 15, 30, 60, 120, 180 and 300 minutes. This allows us to explore the ability to perform accurate classification early on in the first few minutes, as well as to analyse how much the classifier’s performance can improve as time goes on up to 5 hours.

Table I shows the results comparing performance of different features. We observe that the approaches using our proposed method for class-specific word representations (multiw2v) perform significantly better than the rest, including the use of standard word embeddings (w2v). While social features alone perform poorly, they are actually beneficial when they are combined with the multiw2v features. We see that the combination of social+multiw2v consistently outperforms the sole use of multiw2v features, however this improvement is

\[ \text{http://scikit-learn.org/} \]
between the current time and the time of the first tweet was posted, and (2) the current time.

Table IV shows the results of using different time windows: 0.1, 0.25, 0.5, 0.75 and 1.0. We use the social+multiw2v as the best performing features here for the analysis. With these results we observe that the use of sliding windows is not useful, and that it is much better to use all the tweets associated with a report than the last few. While we do observe that it is better to keep including new tweets as time goes on, which leads to performance gains, we also see that it is important to include all tweets from the very beginning. Results for $t = 0$ are the same in all cases as the use of a window does not have an effect in this case.

VI. DISCUSSION

We have introduced a novel approach for semi-automated generation of annotated social media datasets for veracity classification. Different from previous work, our approach does not need to collect true and false stories using different approaches, and consequently enables experimentation in a realistic scenario with a realistic ratio of false stories. Our semi-automated approach consists in leveraging the Wikidata knowledge base, with which we can easily verify if celebrity death reports circulating in social media refer to people who have actually died or are instead made up reports. Following this process, we have produced a dataset comprising 4,007 different death reports, which include over 13 million tweets, and have a ratio of 15% false stories.

The generation of this dataset has also enabled us to run experiments for early hoax detection from social media, which we have experimented for very early detection within minutes of the first report. Taking advantage of the large-scale of our dataset, we have proposed a novel approach that learns class-specific word representations using word embeddings. This approach has proven to clearly outperform the use of a single model of word embeddings for the entire dataset. Our approach achieves competitive results for detection of hoaxes within the first 5 or 10 minutes, with F1 scores above 72% within 10 minutes, and leading up to F1 scores of 77% within 5 hours. With further experimentation, we have observed that the use of sliding windows, where the most recent tweets are considered, leads to performance gains, we also see that it is important to include all the tweets associated with a report than the last few. While we do observe that it is better to keep including new tweets as time goes on, which leads to performance gains, we also see that it is important to include all tweets from the very beginning. Results for $t = 0$ are the same in all cases as the use of a window does not have an effect in this case.

Table IV shows the results of using different time windows: 0.1, 0.25, 0.5, 0.75 and 1.0. We use the social+multiw2v as the best performing features here for the analysis. With these results we observe that the use of sliding windows is not useful, and that it is much better to use all the tweets associated with a report than the last few. While we do observe that it is better to keep including new tweets as time goes on, which leads to performance gains, we also see that it is important to include all tweets from the very beginning. Results for $t = 0$ are the same in all cases as the use of a window does not have an effect in this case.

Table IV shows the results of using different time windows: 0.1, 0.25, 0.5, 0.75 and 1.0. We use the social+multiw2v as the best performing features here for the analysis. With these results we observe that the use of sliding windows is not useful, and that it is much better to use all the tweets associated with a report than the last few. While we do observe that it is better to keep including new tweets as time goes on, which leads to performance gains, we also see that it is important to include all tweets from the very beginning. Results for $t = 0$ are the same in all cases as the use of a window does not have an effect in this case.

### Table II

| Hoaxes | 0  | 5' | 10' | 15' | 60' | 120' | 300' |
|--------|----|----|-----|-----|-----|------|------|
| social | .684 | .728 | .723 | .734 | .771 | .797 | .811 |
| w2v   | .756 | .791 | .806 | .805 | .830 | .847 | .855 |
| multiw2v | .783 | .815 | .830 | .837 | .832 | .849 | .861 |
| social+w2v | .772 | .797 | .816 | .823 | .833 | .852 | .866 |
| social+multiw2v | .790 | .816 | .835 | .848 | .852 | .864 | .874 |

### Table III

| Commemorations | 0  | 5' | 10' | 15' | 60' | 120' | 300' |
|----------------|----|----|-----|-----|-----|------|------|
| social          | .250 | .362 | .413 | .445 | .574 | .584 | .586 |
| w2v            | .648 | .654 | .662 | .664 | .681 | .700 | .714 |
| multiw2v       | .691 | .706 | .721 | .715 | .713 | .730 | .739 |
| social+w2v     | .648 | .668 | .688 | .689 | .712 | .726 | .735 |
| social+multiw2v | .687 | .710 | .721 | .730 | .762 | .770 | .772 |

### Table IV

| window | 0  | 5' | 10' | 15' | 60' | 120' | 300' |
|--------|----|----|-----|-----|-----|------|------|
| 0.1    | .649 | .409 | .424 | .428 | .602 | .594 | .615 |
| 0.25   | .649 | .469 | .511 | .555 | .671 | .692 | .682 |
| 0.5    | .649 | .500 | .576 | .611 | .712 | .719 | .707 |
| 0.75   | .649 | .548 | .609 | .635 | .737 | .748 | .755 |
| 1.0    | .649 | .696 | .726 | .743 | .757 | .767 | .774 |
Our plans for future work include experimentation with other events that can be linked to Wikidata or other knowledge bases, beyond death reports, such as resignation of public figures, numbers of casualties reported for emergency events, or other factual claims.

APPENDIX A
LIST OF SOCIAL FEATURES

With the social features we create vectors with 16 values:

- **User ratio**: Number of unique users divided by the number of tweets.
- **Retweeting user ratio**: Number of unique retweeting users divided by the number of tweets.
- **Tweet length**: Average length of tweets in characters.
- **Retweets per tweet**: Average number of retweets per tweet.
- **Reply ratio**: Number of tweets that are replying to another tweet divided by the number of all tweets.
- **Tweeting rate**: Number of tweets per second.
- **Link ratio**: Number of links found in all tweets divided by the number of tweets.
- **Question ratio**: Number of question marks found in all tweets divided by the number of tweets.
- **Exclamation ratio**: Number of exclamation marks found in all tweets divided by the number of tweets.
- **Picture ratio**: Number of pictures found in all tweets divided by the number of tweets.
- **Tokens per tweet**: Number of (space-separated) tokens found in all tweets divided by the number of tweets.
- **Hashtags per tweets**: Number of unique hashtags found in all tweets divided by the number of tweets.
- **Mentions per tweet**: Number of unique user mentions found in all tweets divided by the number of tweets.
- **Language count**: Number of unique languages used in the tweets.
- **Average follow ratio of users**: We compute the average of the follow ratios of all users. The follow ratio of a user is computed as $\frac{\log_{10}(\text{following})}{\log_{10}(\text{followers})}$.
- **Average follow ratio of retweeting users**: We compute the average of the follow ratios of all the retweeting users.

REFERENCES

[1] H. Allcott and M. Gentzkow. Social media and fake news in the 2016 election. Technical report, National Bureau of Economic Research, 2017.
[2] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin. A neural probabilistic language model. Journal of machine learning research, 3(Feb):1137–1155, 2003.
[3] P. F. Brown, P. V. Desouza, R. L. Mercer, V. J. D. Pietra, and J. C. Lai. Class-based n-gram models of natural language. Computational linguistics, 18(4):467–479, 1992.
[4] A. Bruns, T. Highfield, and R. A. Lind. Blogs, twitter, and breaking news: The prodagage of citizen journalism. Producing theory in a digital world: The intersection of audiences and production in contemporary theory, 80(2012):15–32, 2012.
[5] M. Datar, A. Gionis, P. Indyk, and R. Motwani. Maintaining stream statistics over sliding windows. SIAM journal on computing, 31(6):1794–1813, 2002.
[6] N. Diakopoulos, M. De Choudhury, and M. Naaman. Finding and assessing social media information sources in the context of journalism. In Proceedings of CHI, pages 2451–2460. ACM, 2012.
[7] J. L. Good et al. Maximum entropy for hypothesis formulation, especially for multidimensional contingency tables. The Annals of Mathematical Statistics, 34(3):911–934, 1963.
[8] Z. Jin, J. Cao, Y. Zhang, and J. Luo. News verification by exploiting conflicting social viewpoints in microblogs. In AAAI, pages 2972–2978, 2016.
[9] T. Koo, X. Carreras Pérez, and M. Collins. Simple semi-supervised dependency parsing. In Proceedings of ACL, pages 595–603, 2008.
[10] S. Kumar, R. West, and J. Leskovec. Disinformation on the web: Impact, characteristics, and detection of wikipedia hoaxes. In Proceedings of WWW, pages 591–602, 2016.
[11] H. Kwak, C. Lee, H. Park, and S. Moon. What is twitter, a social network or a news media? In Proceedings of WWW, pages 591–600. ACM, 2010.
[12] S. Kwon, M. Cha, and K. Jung. Rumor detection over varying time windows. PloS one, 12(1):e0168344, 2017.
[13] X. Liu, A. Nourbakhsh, Q. Li, R. Fang, and S. Shah. Real-time rumor debunking on twitter. In Proceedings of CIKM, pages 1867–1870. ACM, 2015.
[14] J. Ma, W. Gao, P. Mitra, S. Kwon, B. J. Jansen, K.-F. Wong, and M. Cha. Detecting rumors from microblogs with recurrent neural networks. In IJCAI, pages 3818–3824, 2016.
[15] F. Menczer. The spread of misinformation in social media. In Proceedings of WWW, pages 717–717, 2016.
[16] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In Advances in neural information processing systems, pages 3111–3119, 2013.
[17] V. Qazvinian, E. Rosengren, D. R. Radev, and Q. Mei. Rumor has it: Identifying misinformation in microblogs. In Proceedings of EMNLP, pages 1589–1599, 2011.
[18] A. Ratnaparkhi. A simple introduction to maximum entropy models for natural language processing. IRCS Tech. Reports Series, page 81, 1997.
[19] J. Sampson, F. Morstatter, L. Wu, and H. Liu. Leveraging the implicit structure within social media for emergent rumor detection. In Proceedings of CIKM, pages 2377–2382. ACM, 2016.
[20] J. Sankaranarayanan, H. Samet, B. E. TenTler, M. D. Lieberman, and J. Sperling. Twittersstand: news in tweets. In Proceedings of SIGSPATIAL, pages 42–51. ACM, 2009.
[21] K. Shu, A. Sliva, S. Wang, J. Tang, and H. Liu. Fake news detection on social media: A data mining perspective. ACM SIGKDD Explorations Newsletter, 19(1):22–36, 2017.
[22] E. Tacchini, G. Ballarin, M. L. Della Vedova, S. Moret, and L. de Alfaro. Some like it hoax: Automated fake news detection in social networks. arXiv preprint arXiv:1704.07506, 2017.
[23] T. Takahashi and N. Igata. Rumor detection on twitter. In Proceedings of SCIS, pages 452–457. IEEE, 2012.
[24] J. Turian, L. Ratinov, and Y. Bengio. Word representations: a simple and general method for semi-supervised learning. In Proceedings of ACL, pages 384–394, 2010.
[25] D. Vrandečić and M. Krötzsch. Wikidata: a free collaborative knowledge base. Communications of the ACM, 57(10):78–85, 2014.
[26] F. Yang, Y. Liu, X. Yu, and M. Yang. Automatic detection of rumor on sina weibo. In Proceedings of the ACM SIGKDD Workshop on Mining Data Semantics, page 13. ACM, 2012.
[27] A. Zubiaga, A. Aker, K. Bontcheva, M. Liakata, and R. Procter. Detection and resolution of rumours in social media: A survey. ACM Computing Surveys, 2017.
[28] A. Zubiaga, H. Ji, and K. Knight. Curating and contextualizing twitter stories to assist with social newsgathering. In Proceedings of IUI, pages 213–224. ACM, 2013.