Weakly Supervised Classification of Tweets for Disaster Management

Girish Keshav Palshikar, Manoj Apte, Deepak Pandita

TCS Research, Tata Consultancy Services Limited,
54B Hadapsar Industrial Estate, Pune 411013, India.
{gk.palshikar, manoj.apte, deepak.p7}@tcs.com

Abstract. Social media has quickly established itself as an important means that people, NGOs and governments use to spread information during natural or man-made disasters, mass emergencies and crisis situations. Given this important role, real-time analysis of social media contents to locate, organize and use valuable information for disaster management is crucial. In this paper, we propose self-learning algorithms that, with minimal supervision, construct a simple bag-of-words model of information expressed in the news about various natural disasters. Such a model is human-understandable, human-modifiable and usable in a real-time scenario. Since tweets are a diffferent category of documents than news, we next propose a model transfer algorithm, which essentially refines the model learned from news by analyzing a large unlabeled corpus of tweets. We show empirically that model transfer improves the predictive accuracy of the model. We demonstrate empirically that our model learning algorithm is better than several state of the art semi-supervised learning algorithms.

1 Introduction

With the ever widening spread of computers, communications and the Internet, social media is becoming increasingly important as a means of social interaction. In particular, social media has quickly established itself as an important means that people, NGOs and governments use to spread information during natural or man-made disasters, mass emergencies and crisis situations. In these scenarios, social media is used to report first-hand ("ground zero") experiences including photos and videos, near-hand observations, contact relatives and friends, request help, disseminate information about available help and other services, organize local search and rescue operations, monitor situation, report status, damages or losses etc. Given this important role, real-time analysis of socia media contents to locate, organize and use valuable information for disaster management is an active research area; see [5] for a comprehensive survey.

In this paper, we propose *a weakly supervised* learning algorithm to automatically detect documents containing information about disasters. We include natural disasters like earthquake, flood, hurricane, famine, forest fire, volcano eruption, tsunami, land slide, disease epidemic (e.g. H1N1, swine flu, ebola) and

Table 1. Some example news and tweets related to earthquakes and floods.

`A moderate earthquake with a magnitude of 5.5 struck Pakistan on Saturday, but no loss to life or property has been reported so far.`
`Record flooding from rain-swollen rivers has washed out hundreds of structures in Missouri, Illinois, Arkansas and eastern Oklahoma, forcing thousands to flee their homes, and 9.3 million Americans still face flood warnings. At least 28 people have died in the U.S.`
`Two tremors jolt central, southeastern Iran #Earthquake`
`#France - Seine River peaked on Saturday. Flood killed four. Thousands forced from homes`

man-made diasasters like nuclear power plant accidents. We exclude crimes, accidents, insurgencies, terrorist attacks and war.

Factual information about various kinds of disasters, as expressed in, say, news stories, is often similar at a broad level: they mostly include aspects like damage, injuries, deaths brought about by a disaster, search, rescue, relief, recovery etc. We generalize this informal observation to a natural principle, which might be called as the **principle of information correspondence**: *in a given type of documents, similar events are expressed similarly.* The similarity of expression of information is essentially at the level of semantics. Clearly, news and tweets are two distinct types of documents. Thus we may expect that different news items will broadly express information about disasters in a similar manner; and different tweets will broadly express information about disasters in a similar manner, although information expression across news and tweets need not be similar. Example text from news and tweets related to two types of disasters: earthquake and flood is shown in Table 1. Crimes, accidents, acts of terrorism, and even weather reports, may sometimes contain similar information as a disaster. Thus any technique for automatically detecting disaster-related information should reject such confounding pieces of information.

In this paper, our main goal is to build a model, with minimal supervision, that can be used to quickly classify tweets in an incoming stream as DISASTER-RELATED$(+1)$ or NOT-DISASTER-RELATED(-1). We require the model to be simple, human-understandable (and even human-modifiable) and usable in a real-time scenario. Towards this end, we propose self-learning algorithms that, with minimal supervision, construct models of information expressed in news about various natural disasters. The constructed model is extremely simple and basically consists of a set (bag) of characteristic words that are often present in information expressed about disasters. The algorithm constructs a single model for all disasters i.e., it does not differentiate among different classes of disasters, although our approach is easy to use for learning a model for a specific disaster type. Since tweets are a diffferent category of documents than news, we next propose a model transfer algorithm, which essentially refines the model learned from news by analyzing a large unlabeled corpus of tweets. We show empirically that model transfer improves the predictive accuracy of the model. We demonstrate empirically that our model learning algorithm is better than several state of the art semi-supervised learning algorithms. Section 2 contains related work, section 3 contains learning algorithms, section 4 contains baselines, section 5 contains experiments and section 6 outlines conclusions and further work.

2 Related Work

In this section, we summarize the work related to disaster detection, focusing primarily on semi-supervised learning and transfer learning. [5] surveyed the computational methods to process social media messages and map them to the problems like detection of events, creation of actionable and useful summaries etc. [14] has used textual, social and temporal characteristics to detect events on social streams. Social text streams are represented as multi-graphs with nodes as social actors and information flows as edges. Events are detected by combining text-based clustering, temporal segmentation, and information flow-based graph cuts of the dual graph of the social networks. [12] has built an earthquake reporting system based on event detection. They have used a classifier based on the keywords in tweet, the size of tweet based on number of words and the context to detect earthquake like event. Once an event is identified a probabilistic spatiotemporal model is built for finding the center and trajectory of the event. LITMUS [8] is a landslide detection system which integrates USGS seismic data, NASA TRMM Rainfall network with Twitter, Instagram and Youtube. Social media data is filtered using keyword-based filtering, geotagging, classification and relevance score is computed to detect landslides. [11] cast seed-based event extraction as a weakly supervised learning problem where only positive and unlabeled data is available. They regularize the label distribution over unlabeled examples towards user-specified expectation of the label distribution for the keyword. [16] presented a self-training algorithm that decreases the disagreement region of hypotheses. The algorithm supplements the training set with self-labeled instances. The instances that greatly reduce the disagreement region of hypotheses are labeled and added to the training set.

There is a large amount of work in semi-supervised classification which uses a large amount of unlabeled data and a small amount of labeled data to build better classifiers. [9] introduced an algorithm for learning from labeled and unlabeled text documents based on the combination of Expectation-Maximization (EM) and a naive Bayes classifier. A classifier is trained using the labeled documents and probabilistically labels are predicted for unlabeled documents. Then a new classifer is trained using the labels for all the documents and iterates to convergence. We use a simplified version of this approach as a baseline. [3] utilized the semi-supervised sarcasm identification algorithm of [13] and proposed SA SI algorithm that successfully captures sarcastic sentences in twitter and other domains. The algorithm employs two modules: semi supervised pattern acquisition for identifying sarcastic patterns that serve as features for a classifier, and a classification stage that classifies each sentence to a sarcastic class.

Transfer learning involves leveraging knowledge learnt from source domain/task to improve learning in target domain/task. [2] has proposed a transfer-learning algorithm for text classification based on an EM-based Naive Bayes classifier. First the initial probabilities under a distribution of labeled data set are estimated and then an EM algorithm is used to revise the model for a different distribution of the unlabeled test data. This approach has been used by [15] for crowd-selection on twitter. [1] analyzes sentiments by using opinion holder bias prediction. First, the bias of a social media user towards a specific topic is measured by solving the relational learning task over a network of users connected by endorsements. The sentiments are analyzed by transferring user biases to textual features. They show that even when the topic changes its profile as new terms arise and old terms change their meaning, the user bias helps in building more accurate classification models due to consistency over time.

3 Learning Algorithms

3.1 Weakly Supervised Model Learning from News

Our approach is two-fold. In the first *learning phase*, we learn a one-class classification model for identifying documents of class +1 (which are disaster reporting news), as against any other kind of document (class = -1). The learnt model is in the form of a word set W, consisting of words which characterize only the class +1. We are not interested in characterizing class -1, and in that sense this is a one-class classification problem. The model W is learnt in a weakly supervised manner - the only "help" given to the model learning algorithm is in the form of a small labeled *seed set* D, where each document in D is labeled with class = +1 (i.e., each document is a known to be related to disaster), and a small set W_0 of known seed words which partially "characterize" class +1 (i.e., are related to disasters). In addition, a large corpus U of unlabeled documents is given i.e., none of the documents in U are labeled with any class label (either +1 or -1). Since the text in news is significantly different than that in tweets, we need to transfer this model to learn to classify tweets. That part is discussed in section 3.2. Finally, in the *prediction (operation) phase*, we use the final model (i.e., the news domain model transferred to the tweets domain) to dynamically classify any incoming tweet as having class = +1 or not.

We represent each input document U_i as a *word tuples sequence (WTS)* σ_i, which is an ordered sequence of word tuples: $\sigma_i = \langle wt_1, wt_2, \ldots, wt_{N_i} \rangle$, where N_i is the number tokens in the document U_i and each wt_j, $1 \leq j \leq N_i$ is a word tuple. Each *word tuple* has the form $wt_j = (w_j, t_j, c_j, f_j)$, where w_j is a word token, t_j is its POS tag (using Stanford POS tagger), c_j is the *term frequency* i.e., the number of times this token occurs in the current input document irrespective of its POS tag, f_j is its *document frequency* i.e., the number of documents in the entire corpus U in which this token occurs irrespective of its POS tag. Thus each word tuple is essentially a feature vector for each word token in the document. Word tokens are considered after stopword removal and stemming. We insert a dummy word tuple to mark the sentence boundaries. Note that if a word w occurs multiple times in the same document, σ_i will contain multiple word tuples corresponding to w; these word tokens may differ in the POS tag component, but otherwise they would be identical. If N denotes the number of documents in the corpus U, then one way to to compute the TFIDF for a word w_j is: $\frac{c_j}{N_i} \cdot log \frac{N}{f_j}$.

The algorithm *learn_disaster_model* (Figure 1) initializes the model W with the given set W_0 of characteristic keywords, plus words that frequently occur "around" words in W_0 in the known disaster reporting documents D. Initially, all documents in U are marked as -1. The algorithm iteratively examines each document in U (among those which are still marked as -1), and checks if the label for this document can be changed to $+1$, as follows. If the set W_2 of frequently occurring words in this document do not have a significant overlap with the current model W, then this document is ignored currently i.e., its label continues to be -1. If this document does contain a significant overlap with the current model W, then this document is marked as $+1$ and is never considered again. But before going to the next document, the algorithm selects those words (if any) from W_2, which occur in WordNet but whose corpus count in WordNet is not "too high", and adds them to the current model. After finishing the examination of all documents in U, the algorithm continues to the next iteration, because the labels of some more documents may now change, if new words were added to W in the previous iteration. The algorithm stops after a user-specified number of iterations or if no words were added to W in the previous iteration.

The subroutine $GetContextWords(W_0, n, window, S)$ works as follows. For every word w in given seed list W_0, compute the set X of all words (nouns or verbs only) which occur before or after w in a window of given size in any sentence in the documents in the given set of documents S. For each word x in X, find the number of documents in S in which x occurs and remove x from X if this frequency is less than the given threshold n. So far, the output model s just a set of words. The algorithm can be modified to compute a weight for each word w_i; e.g., its TFIDF score, or the conditional probability $P(w_i|class = +1)$.

3.2 Model Transfer Algorithm

Suppose we have a model W_{news} learned from news. The simplest approach would use the model learned from the news corpus as it is on tweets, to predict which tweets are related to disasters. We will show in section 5.2 that this approach has less accuracy, because news and tweets are different types of documents in terms of the vocabulary and style of writing. We need to refine the model W_{news}, by removing and adding words, to construct a new model W_{tweet}. We propose a new transfer learning algorithm, *augment_model*, to augment the model from a source domain by examining the unlabeled corpus from the target domain. For this, we assume that we have an unlabeled corpus of tweets available (dataset $D4$). A new word is added to the model if it co-occurs with "sufficient" frequency with a word in W_{news}. *Pointwise mutual information (PMI)* between two words u and v is defined as $PMI(u, v) = log\left(\frac{p(u,v)}{p(u)p(v)}\right)$ where, $p(u, v)$ is the probability of co-occurrence of u and v, $p(u)$ and $p(v)$ is the probability of ocurrence of u and v respectively. Out of available alternatives, we use PMI as a measure of similarity between a word in the model and any other word in the

```
**algorithm** learn_disaster_model
**input** $D = \{D_1, \ldots, D_k\}$; // $k$ known disaster reporting news
**input** $W_0 = \{w_1, \ldots, w_n\}$; // $n$ seed words related to disasters
**input** $U = \{U_1, \ldots, U_m\}$; // $m$ documents with unknown class
**output** $W$; // output model (words related to disasters)
**input** $n_D, n_0, n_1, n_{inv}, c_{noun}, c_{verb}, \theta_0, window$; // hyper parameters of the algorithm
$W := W_0 \cup GetContextWords(W_0, n_D, window, D)$;
$flag :=$ **true**; $iter := 1$;
**while** ($flag$ **and** $iter \leq MaxIter$)
    $flag :=$ **false**; // reset $flag$
    **for** $(i = 0; i \leq m; i++)$ // do for each document in $U$
        **if** ($U_i$ is already marked as +1) **then continue**; **endif**
        $W_1 := GetFrequentWords(n_0, \{U_i\})$; // words in $U_i$ with frequency $\geq n_0$
        **if** $|W_1 \cap W_0| = \emptyset$ **then continue**; **endif**
        $sim := WordsetSim(W, W_1)$;
        **if** $(sim \leq \theta_0)$ **then continue**; **endif**
        $W_2 := \emptyset$;
        **foreach** (word $u \in W_1$) **do**
            **if** ($GetTokenFreq(u, \{U_i\}) \geq n_1$ **and** $WNHasWord(u)$ **and**
                $WNFreq(u, noun) \leq c_{noun}$ **and** $WNFreq(u, verb) \leq c_{verb}$ **and**
                $GetTokenFreq(u, U - \{U_i\}) \leq n_{inv}$) **then** $W_2 := W_2 \cup \{u\}$; **endif**
        **end foreach**
        $W := W \cup W_2$; Mark $U_i$ as +1; $flag :=$ **true**;
    **end for**
    $iter++$;
**end while**
```

Fig. 1. Algorithm to learn the disaster word model.

unlabeled corpus. We select top N_0 (we used $N_0 = 25$) having the highest PMI with any word in the model W_{news} and remove words not present in WordNet (unless they begin with #), or are named entities person, location, organization etc. We add the remaining words from this list to W_{news} to get W_{tweet}.

3.3 Model-based Classification

Let a given document (a news or a tweet) d contain words (nouns or verbs) $W_d = \{v_1, v_2, \ldots, v_k\}$. We have a simple and efficient real-time algorithm *identify_disaster_tweet* that can use the given model W to predict the class label for any given document d. Basically, if the similarity between the set W_d and the given model W is more than a user-specified threshold θ_1 then the algorithm predicts $class = +1$ (disaster related) else it predicts $class = -1$ (not disaster related). We use the Jaccard similarity between W and W_d.

We have found that disaster related documents sometimes look similar to those related to crime, accidents, war, terrorism or weather forecasts. To reduce this confusion, we can use the corpus to create separate models for each of these class of documents. We then modify our model-based classification algorithms to use these *negative models* as follows. If the similarity between the set W_d and any given negative model W_{neg} is more than a user-specified threshold θ_1 then predict $class = -1$ for d. If d is not similar to any of the negative models, only then we use the previous rule to predict whether d is disaster related or not.

4 Baseline Methods

We have created some baseline methods to compare our approach with. Starting with a given set W_0 of "seed" keywords characterizing disasters, the algorithm *wordset_expansion* detects and adds other words (only nouns or verbs) in a given unlabeled corpus, which are very similar to those in W_0 i.e., it creates a single "cluster" of words, starting with a set of cluster prototype or representative words. The algorithm does not use the set of known disaster documents, nor does it impose any restrictions on the frequencies of the words to be added. The cosine similarity uses the word embeddings produced by GloVe [10]. We designed a second baseline method which uses topic modeling (*topic_based*). We extracted 300 topics using the mallet toolkit [7] on dataset $D1$ and manually labelled the topics as disaster-related or not. We found 7 topics (out of 300) to be related to disasters. For example, following are some words in one of the topics: `earthquake, ocean, warning, quake, tsunami, tremor, magnitude, strike, damage`. For any given document, mallet gives a topic distribution within that document. We used a simple classification rule that if the most frequent topic within a document D is one of the diasaster related topics, then label D as $+1$ else as -1. We use this topic-based classification scheme as a baseline because, it is also weakly supervised, like our approach. We used the semi-supervised classification method of Transductive SVM [6]($transductive\_SVM$) as the third baseline; we used SVM-Lite tool to train a Transductive SVM using dataset $D1$. Our fourth baseline algorithm is a self-training algorithm $NB\_iterative$, which takes the same set of unlabeled and positive examples as given to *learn_disaster_model*, along with a small set N of known negative examples. In each iteration, it trains a simple Naive Bayes classifier on the current sets of positive and negative examples, and predicts a class label L_u for each document $u \in U$ with confidence c_i. If $L_u = +1$ and c_i is sufficiently high then it adds u to D and removes it from U; else if $L_u = -1$ and c_i is sufficiently high then it adds u to N and removes it from U; otherwise, u remains in U without any class label. After a specified number of iterations, the final Naive Bayes model is tested. This algorithm does not use any user-specified set of seed keywords.

5 Experimental Studies

5.1 Datasets

Training News Dataset D1. This dataset contains 9983 documents. Out of these, 10 earthquake related documents are explicitly labeled as +1 (DISASTER-

RELATED), and are used as seeds. Among the 9973 remainining unlabeled documents, we know that there are 50 documents related to other disasters (e.g., hurricanes and floods), 60 each for crime, accidents and weather, although this information is *not* passed to the disaster model learning algorithm. The remaining 9743 unlabeled documents are randomly selected news items from the FIRE corpus, some of which may be related to disasters, crime, accidents or weather, but we do not know which ones. The FIRE (Forum for Information Retrieval Evaluation) corpus[1] contains 392,577 English news items from Indian newspapers such as The Telegraph and BDNews. **Test News Dataset D3**. This fully labeled dataset consists of 2537 news items, out of which 162 are labeled $+1$ (they are related to several natural disasters) and 2375 are labeled -1. Among the latter, 2225 news items (labeled -1) are from the BBC news website [4] corresponding to stories in five topical areas (business, entertainment, politics, sports and technology) from 2004-2005, and 50 each are news related to crime, accidents and weather. **Labeled Tweets Dataset D2**. This dataset contains 1220 disaster related tweets (labeled $+1$) and 1105 non-disasters tweets (labeled -1), among the latter 100 tweets related to crime, 100 tweets related to accidents and 100 tweets related to weather. The tweets were labeled manually by us. The tweets labeled $+1$ contains tweets related to a variety of natural disasters, such as avalanche, cyclone, drought, flood, forest fire, landslide, tsunami, volcano as well as nuclear accidents and biological disasters. **Unlabeled Tweets Dataset D4**. Using the Twitter Streaming API, we downloaded a corpus of 7,555,000 English language tweets from 12 to 19 September 2016. All tweets are unlabeled.

5.2 Results

We trained the algorithm *learn_disaster_model* on dataset D1. We created different models, by changing the value of θ_0. We used these word models to predict the class labels on dataset D3, again, for different values for the parameter θ_1. The best results were obtained for the word model created on $D1$ using $\theta_0 = 0.085$ and applied on $D3$ with $\theta_1 = 0.025$. This specific model contained 40 words. We used it to predict disaster related tweets on dataset $D2$, which gave $F = 0.71$ (entry M1 in table 2). So far, these results *do not* use our transfer learning algorithm. Next, we started with the model $M1$ (as created by *learn_disaster_model* on $D1$) and used our transfer learning algorithm *augment_model* on $D4$, which led to the addition of these words to the model `erupt, lava, #prayforkorea`. We tested this augmented (transferred) model on $D2$ (with $\theta_1 = 0.025$), which gave a higher F-measure of 0.749, indicating the advantage provided by the transfer learning even in the unsupervised setting (entry M2 in table 2).

We also used the model $M2$ to predict disaster related tweets on $D4$. The model labelled 9527 tweets (out of 7.555M) as DISASTER-RELATED. We selected and manually labeled top 100 tweets (dataset $D5$) based on their similarity with $M2$ (79 tweets as $+1$, 21 as -1). We then used *negative models* for Accident and Crime to modify the predictions of $M2$ as mentioned earlier. The modified

[1] http://fire.irsi.res.in/fire/data

Table 2. Experimental results on dataset D2.

Algorithm	P	R	F
wordset_expansion	0.642	0.098	0.171
NB_iterative	0.827	0.419	0.556
transductive_SVM	0.628	**0.585**	0.606
topic_based	**0.961**	0.44	0.604
M1	0.934	0.573	**0.71**
M2	**0.939**	**0.622**	**0.749**

algorithm correctly predicted 79 tweets as +1 and modified the predictions of 5 tweets (out of 21) which were incorrectly labeled +1 by $M2$ to −1 (16 tweets labeled −1 are still labeled as −1). This validates our proposition that *negative models* can improve the accuracy of our algorithm. Next, we used our baselines to predict disaster related tweets on $D5$. Among the baselines, *topic_based* performed better than other baselines by correctly labeling 70 tweets as +1. *transductive_SVM* and *NB_iterative* labeled 65 and 44 tweets correctly as +1 respectively. *wordset_expansion* peformed the worst by correctly labeling only 2 tweets as +1. This shows the robustness of our learning algorithm to handle unseen tweets.

6 Conclusions and Further Work

In this paper, we proposed a weakly supervised algorithm to learn a bag of words model for various disasters from news corpus and then proposed a simple model transfer algorithm that augments the news-based model from a corpus of unlabeled tweets. The proposed algorithm performs better then several baselines based on semi-supervised learning approaches. The learned model is easy for humans to understand and modify. We also demonstrated the effectiveness of this approach on completely unseen stream of tweets. We are going to use this approach to detect other kinds of information, such as HIV/AIDS related tweets. A limitation of this approach is that the model structure is very simple; we are investigating more complex model structures to improve the performance. We have developed techniques (based on χ^2-test and test of proportions) to detect any decay in the importance of words in the model (e.g., hashtags) over time. We are exploring other model modification mechanisms as well as online learning algorithms to modify the model on the fly during operational deployment.

References

1. Pedro Henrique Calais Guerra, Adriano Veloso, Wagner Meira Jr, and Virgílio Almeida. From bias to opinion: a transfer-learning approach to real-time sentiment analysis. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 150–158. ACM, 2011.

2. Wenyuan Dai, Gui-Rong Xue, Qiang Yang, and Yong Yu. Transferring naive bayes classifiers for text classification. In *Proceedings of the national conference on artificial intelligence*, volume 22, page 540. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2007.
3. Dmitry Davidov, Oren Tsur, and Ari Rappoport. Semi-supervised recognition of sarcastic sentences in twitter and amazon. In *Proceedings of the fourteenth conference on computational natural language learning*, pages 107–116. Association for Computational Linguistics, 2010.
4. Derek Greene and Pádraig Cunningham. Practical solutions to the problem of diagonal dominance in kernel document clustering. In *Proc. 23rd International Conference on Machine learning (ICML'06)*, pages 377–384. ACM Press, 2006.
5. Muhammad Imran, Carlos Castillo, Fernando Diaz, and Sarah Vieweg. Processing social media messages in mass emergency: A survey. *ACM Computing Surveys*, 47(4):67:1–67:38, 2015.
6. Thorsten Joachims. Transductive inference for text classification using support vector machines. In *Proceedings of the Sixteenth International Conference on Machine Learning (ICML 99)*, pages 200–209, 1999.
7. Andrew Kachites McCallum. Mallet: A machine learning for language toolkit. http://mallet.cs.umass.edu, 2002.
8. Aibek Musaev, De Wang, and Calton Pu. Litmus: Landslide detection by integrating multiple sources. In *11th International Conference Information Systems for Crisis Response and Management (ISCRAM)*, 2014.
9. Kamal Nigam, Andrew Kachites McCallum, Sebastian Thrun, and Tom Mitchell. Text classification from labeled and unlabeled documents using em. *Machine learning*, 39(2-3):103–134, 2000.
10. Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014.
11. Alan Ritter, Evan Wright, William Casey, and Tom Mitchell. Weakly supervised extraction of computer security events from twitter. In *Proceedings of the 24th International Conference on World Wide Web*, pages 896–905. ACM, 2015.
12. Takeshi Sakaki, Makoto Okazaki, and Yutaka Matsuo. Earthquake shakes twitter users: real-time event detection by social sensors. In *Proceedings of the 19th international conference on World wide web*, pages 851–860. ACM, 2010.
13. Oren Tsur, Dmitry Davidov, and Ari Rappoport. Icwsm-a great catchy name: Semi-supervised recognition of sarcastic sentences in online product reviews. In *ICWSM*, 2010.
14. Qiankun Zhao, Prasenjit Mitra, and Bi Chen. Temporal and information flow based event detection from social text streams. In *AAAI*, volume 7, pages 1501–1506, 2007.
15. Zhou Zhao, Da Yan, Wilfred Ng, and Shi Gao. A transfer learning based framework of crowd-selection on twitter. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1514–1517. ACM, 2013.
16. Yan Zhou, Murat Kantarcioglu, and Bhavani Thuraisingham. Self-training with selection-by-rejection. In *2012 IEEE 12th International Conference on Data Mining*, pages 795–803. IEEE, 2012.