Predicting time-to-event from Twitter messages

Hannah Tops Antal van den Bosch Florian Kunneman

Centre for Language Studies, Radboud Universiteit Nijmegen,
P.O. Box 9103, NL-6500 HD Nijmegen, The Netherlands

Abstract

We describe a system that estimates when an event is going to happen from a stream of microtexts on Twitter referring to that event. Using a Twitter archive and 60 known football events, we train machine learning classifiers to map unseen tweets onto discrete time segments. The time period before the event is automatically segmented; the accuracy with which tweets can be classified into these segments determines the error (RMSE) of the time-to-event prediction. In a cross-validation experiment we observe that support vector machines with χ^2 feature selection attain the lowest prediction error of 52.3 hours off. In a comparison with human subjects, humans produce a larger error, but recognize more tweets as posted before the event; the machine-learning approach more often misclassifies a 'before' tweet as posted during or after the event.

1 Introduction

Twitter[1] is a microblogging service for posting messages, or tweets, with a maximum length of 140 characters. As one of the key social media it has gained immense popularity in recent years. Users of Twitter often communicate what they did in the past, are doing at the moment, or will be doing in the future. In our research we study a property of a specific subtype of tweets, anticipatory tweets that refer to a future event, in order to estimate the point in time at which the referred to event will take place. Such estimates, in combination with some aggregation or ranking, could be used to generate a forecast of future events. A system like this could assist journalists, the police, and the public in being prepared for events that are still several days or weeks in the future.

Our approach is to first collect tweets about events that already happened, and of which we know when. For each tweet we identify its time of posting relative to the event and then train a machine learning classifier to map unseen tweets onto discrete periods. Previous research [4] showed that tweets posted before event start time can accurately be distinguished from tweets during and after an event. However, a classification with a higher specificity than 'before' needs to be made in order to estimate the time-to-event. In [3] the time-to-event in hours was estimated, resulting in a performance that was at best 43 hours off on average. As it is hard to map tweets onto a specific time-to-event in hours, in the current research tweets within the 'before' category are divided into broader time categories spanning a multitude of hours. Given the imbalanced distribution of tweets referring to an event over time (the large majority of tweets is posted right before, during or right after an event), the segmentation of time categories is not a trivial matter. Therefore, two different segmentation methods were compared: left branching and k-means clustering.

[1] `http://twitter.com`

2 Related research

Social media has become a popular domain in text mining. Unique advantages of Twitter as a development platform are the actuality of tweets, their rich metadata (on user, time, and optionally location), their large quantity, and information-rich communicative conventions such as hashtags. On the other hand, when seen as a single medium, Twitter is highly fragmented into short, often noisy messages carrying a large amount of redundant information. Tweets also have the tendency to be self-contained, which means they usually only have a simple discourse and pragmatic structure [6].

In this research we aim to benefit from the information, quantity and especially temporal metadata of tweets in order to make accurate time-to-event predictions, preferably days ahead of the start of an event. The works of Ritter *et al.* [6] and Weerkamp and De Rijke [8] are most comparable to this aim. Ritter *et al.* [6] train on annotated open-domain event mentions in tweets in order to create a calendar of events based on explicit date mentions and words typical of the event, such as named entities. In comparison, we focus on tweets with the same designated hashtag for an event and use these to decide if and when the event will take place in the future. Furthermore, we differentiate between the time-to-event of tweets and apply this in a machine learning framework, taking into account any word indication of the time-to-event rather than merely date or entity mentions, and make estimations for each separate tweet. Weerkamp and De Rijke [8] study anticipation in tweets, focusing on personal activities in the very near future. While they also look at future mentions of events, they do not predict the time itself.

3 Experimental set-up

3.1 Data

For this study we focus on football matches played in the Eredivisie[2] as events. They usually generate a sufficient amount of tweets, occur frequently, and for each match there is a distinctive hashtag by convention ('#feyaja' for a match between Feyenoord and Ajax, Feyenoord being the home team). We harvested tweets by means of `twiqs.nl`, a database of Dutch tweets from December 2010 onwards. We selected the top 6 teams of the league (Ajax, AZ, FC Twente, FC Utrecht, Feyenoord and PSV) and queried tweets referring to all matches played between them in the years 2011 and 2012, which resulted in 60 matches. Querying was done based on the conventional hashtag for each match, with a restricted search space of three weeks before and after the match. This restriction was to make sure that the events mentioned in the tweets were referring to the same match, and not a match played between the same home and away team on a different time but having the same hashtag. The harvesting resulted in 703,767 tweets.

During pre-processing all retweets were filtered, reducing the total amount to 406,517 tweets. A retweet is a reply to a tweet that includes the original message and a variant of the term 'RT'. We removed them primarily because they can be posted several days after the original tweet while containing the same words.

As features we extracted all uni-, bi- and trigrams of words from the tweets. Unigrams appearing 10 times or less, and bi- and trigrams appearing 40 times or less were filtered from the feature space; the remaining features are encoded as binary features (a bag-of-n-gram encoding). Additionally the length of the tweet, rounded at a step size of 5 words, is encoded as an extra numeric feature. This results in a vector space of 63,810 features.

3.2 Segmenting the 'before' category

At the most general level, tweets referring to an event can be divided into the categories 'before', 'during' and 'after'. As we are interested in estimating the time-to-event for each tweet at some resolution, the 'before' category needs to be divided into more specific time categories. The most straightforward way to do this would be to distinguish time categories by a fixed length of time. However, as can be seen in Figure 1, the vast majority of the 'before' tweets is posted during the hours right before the start of an event – note that

[2]The highest-level Dutch football league

the y-axis of the figure is logarithmic – with an average time between tweet and event of the before tweets of -32.6 and median of -4. In the final hour before the events we included in our study, 62,000 tweets are posted, compared to fewer than 7,500 during the sixth hour before the event. Thus, when splitting the data into fixed time segments longer than a few hours, almost all tweets would be residing in the category closest to event time. Another straightforward way would be to distinguish categories by a fixed number of tweets, but then a similar problem arises. For example, distinguishing seven categories in this way, the category that is furthest from event would contain all tweets from 72 hours or longer before event start time. Therefore we experimented with two alternative division schemes: left branching and k-means clustering.

Applying the left branching scheme, the data is split into progressively longer time categories, where each split breaks the remaining tail in two. Starting from the 'before' category as a whole and moving from the start of an event backwards, categories are split at the hour where the amount of tweets on both sides is approximately equal. After each split, the category on the right (closer to event start time) is fixed and another split is performed on the category on the left. This process is repeated until the most distant time category is more than 10 days before the start of an event, which results in seven 'before' categories. This division scheme takes into account the imbalanced temporal distribution of the data by making splits based on frequency, and restricts large variations of time segment lengths by only making splits in a left branching fashion.

For the k-means clustering division scheme we selected the time categories with the smallest Sum of the Squared Error (SSE) with respect to their centroids, based on the content-based bag-of-n-gram representation described in the previous subsection. In order to compare k-means clustering to the left branching approach, we fixed the number of time categories to 7. Starting with 7 arbitrary time categories, their centroid and the Euclidean distance of each instance to the centroids are computed. Instances are assigned to the category with the nearest centroid, and the SSE with respect to the centroids is recomputed for the new division. This process is repeated until the SSE stops decreasing. In comparison to the left-branching scheme, k-means clustering takes into account both the content and the distribution of tweets. This way, time segments that are more coherent (possibly in a contextually meaningful way) will be favoured.

The time categories resulting from the two categorization schemes are displayed in Table 1. As can be seen, k-means clustering leads to more compact time categories close to event start time and less compact time categories further away from an event, in comparison to the left-branching approach.

Table 1: Resulting segmentations (starting-ending hour and number of tweets, rounded to hundreds) of the two segmentation methods.

Category	0	1	2	3	4	5	6	7	8
Left	$504-298$ 2,300	$297-219$ 2,500	$218-151$ 4,800	$150-79$ 9,700	$78-32$ 19,400	$31-5$ 39,100	$4-1$ 79,700	during	after
k-means	$504-267$ 2,100	$266-132$ 7,200	$131-69$ 9,700	$68-34$ 12,400	$33-13$ 22,600	$12-4$ 32,300	$3-1$ 71,300	during	after

3.3 Classification

3.3.1 Classifiers

Three machine-learning classifiers were applied to the events split by the two segmentation methods: Naive Bayes (NB), k-nearest neighbor classification (k-NN), and Support Vector Machines (SVM).

NB is a probabilistic classifier based on the assumption that all features are conditionally independent of each other, given the class. During training, a model is generated by calculating the prior probability for each class and the probability of each feature given the different classes. During testing, the probability from the occurrence of each feature with regards to the class of the instance is calculated. The predicted class of the instance is the class with the highest probability [5].

k-NN [2] is based on the degree of similarity between instances. Given a test instance, it finds a group of k instances in the training set that are closest. The majority class of those k instances is assigned to the

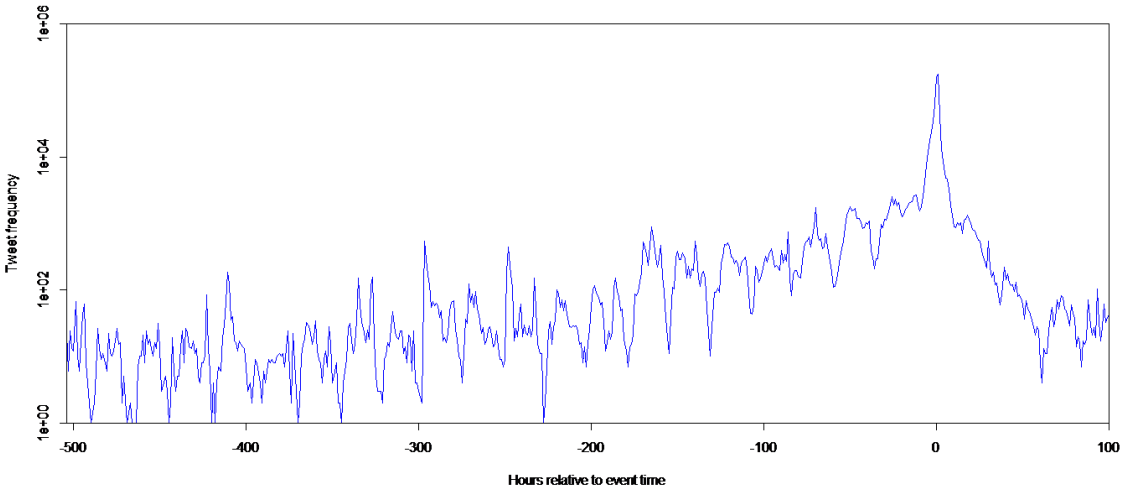

Figure 1: Average number of tweets per hour before and after event starting time. Y-axis is logarithmic.

test instance.

SVM is a robust and generally accurate classification method [9, 1]. In a two-class learning task, the aim of SVM is to find a decision surface with a maximal margin that best separates the positive from the negative instances. The instances in the training set closest to the decision surface are called support vectors.

3.3.2 Feature selection

To reduce the feature space and noise represented in the data, we ranked features based on their indicativeness for separating categories. From these we selected the 10,000 highest-ranked features to be used for classification. Both Information Gain and χ^2 were applied to rank features, resulting in 6 classification systems (3 classifiers $\times$ 2 features selection algorithms). Information Gain estimates how much information the presence of a feature contributes to making the correct classification. χ^2 uses the χ^2 statistic to evaluate the individual relevance of a feature with respect to each class, compared to its expected relevance [5].

3.4 Evaluation

To test the performance of our systems, we performed 'leave-six-events-out cross-validation', approximating 10-fold cross-validation, and involving repeated splits between 6 events as test data and 54 as training data. This is repeated 10 times such that each event is used once as test data, and splits were made before the feature selection step. Folds were split at the event level because it is unrealistic to both train and test on tweets referring to the same event.

In order to score the amount of time in hours that a system is off in its estimations, we calculated the Root Mean Squared Error (RMSE), a common metric for evaluating numeric predictions. The sum of the squared differences between the actual value v_i and the predicted value e_i of all predictions is calculated, their mean is computed, and then the square root is taken to produce the RMSE (see equation 1). We calculated the RMSE for the last element (end point) of the predicted class. For example, when a tweet is posted 78 hours before an event and the system classifies the tweet with a category that ranges from 266 to 132 hours before event time, the squared error of this estimation would be $(78 - 132)^2$.

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(v_i - e_i)^2} \tag{1}$$

As only the seven 'before' categories can be expressed as numeric values, as opposed to the nominal 'during' and 'after' categories, the RMSE can only be calculated when both the classification and the true label are one of these seven categories. Therefore, in addition to their RMSE we measured the *responsiveness* of each system: the relative amount of occasions where the classifier generated a 'before' classification (classifying with one of the categories 0-6) when it should have. A system with a low RMSE and a high responsiveness is to be favored over a system with a low RMSE and a low responsiveness.

Apart from RMSE rates, we calculated the F1-score [7] and distance error for each separate category. While the F1-score is indicative of the correct classification of each category taken individually, the distance error takes into account the sequential ordering of the categories by executing higher penalties for estimations further off from the actual category. When a tweet should be classified as class 0 but is classified as class 2, it receives 2 penalty points. This is calculated for each class, and then averaged by the total amount of tweets in that class. The distance error is an approximation of the RMSE error but still has an intuitive meaning as indicating how 'off' a classifier is in identifying the correct segment in the array of segments.

4 Results

The RMSE and responsiveness are scored for each classification system in the context of the two segmentation methods (see Table 2). While the responsiveness is somewhat higher for systems applied to the left-branching segmentation scheme, segmenting the 'before' class by k-means clustering leads to a better RMSE for every system, achieving improvements of up to 2.6 hours. The differences are highest for the Naive Bayes classifier. However, because of the high standard deviations, the difference in performance of the two categorization schemes is almost negligible.

The SVM classifier outperforms k-NN and NB by several hours, with the lowest RMSE at 52.3. The responsiveness for SVM is the worst, however, indicating that this classifier has a higher tendency to overpredict the 'during' and 'after' categories. Naive Bayes (with χ^2) has the best responsiveness of 82%.

Alternating the Information Gain and χ^2 feature selection methods is most influential in the case of Naive Bayes, for which Information Gain is favourable. On the other hand, SVM achieves a slightly better performance with χ^2 feature selection.

Table 2: RMSE (hours) for the left branching and k-means segmentations. The responsiveness is given in brackets. For each RMSE score the Standard Deviation (σ) is calculated.

Alg.	Feature Selection	Left branching	σ	k-means	σ	Difference (left branching - k-means)
k-NN	IG	62.7 (.78)	7.1	60.5 (.77)	6.6	2.2
k-NN	χ^2	62.1 (.78)	6.8	60.6 (.77)	6.6	1.5
NB	IG	58.9 (.81)	5.2	56.6 (.81)	5.3	2.3
NB	χ^2	61.0 (**.82**)	4.8	58.4 (**.82**)	4.8	2.6
SVM	IG	54.8 (.75)	5.3	53.3 (.74)	5.3	1.5
SVM	χ^2	54.2 (.74)	6.0	**52.3** (.73)	6.4	0.9

4.1 Classification of time categories

4.1.1 Left branching

We calculated the per-category F1-score of the different time segments identified by the two segmentation methods. The scores for the left-branching categories are shown in Table 3. These show that making the right classification is harder if the event is further away, although categories 0 to 4 do not show a linear pattern, with the absolute worst performance for category 1. The 'during' category has the highest F1-score (0.72) which is likely to be at least partly due to the fact that the majority of tweets is posted during the event. The performance of the different systems shows that SVM consistently achieves the highest F1-score for every category, while Naive Bayes outperforms k-NN for every category but 'during'.

Table 3: F1-scores of the left-branching categorization method. The small numbers represent the standard deviations.

Alg.	Feature Selection	before 0	1	2	3	4	5	6	during 7	after 8	MICRO-F1
k-nn	IG	.10 .03	.04 .02	.23 .08	.17 .02	.22 .02	.38 .02	.53 .02	.63 .03	.41 .02	.48 .02
k-nn	χ^2	.10 .04	.04 .03	.23 .08	.17 .03	.22 .02	.38 .02	.52 .02	.63 .03	.41 .03	.48 .02
NB	IG	.17 .04	.05 .02	.28 .10	.21 .02	.27 .04	.48 .02	.58 .01	.52 .02	.54 .03	.51 .02
NB	χ^2	.16 .04	.04 .03	.27 .11	.20 .02	.26 .04	.48 .02	.54 .01	.58 .02	.53 .03	.49 .02
SVM	IG	.21 .07	.05 .03	.36 .11	.25 .05	.38 .06	.50 .03	.60 .03	.72 .05	.56 .04	.56 .03
SVM	χ	.23 .07	.07 .03	.36 .10	.26 .05	.38 .03	.50 .03	.60 .02	.72 .03	.56 .03	.55 .02

4.1.2 k-means clustering

The per-category F1-scores for k-means clustering are given in Table 4. It may seem odd that the MICRO-F1 scores in this table are all worse than the MICRO-F1 scores for left branching, given that the RMSE rates are consistently better in the case of k-means clustering. A closer look reveals that the worse overall performance is due to the lower scores for the 'before' categories closest to event time, that contain most instances. The better RMSE is reflected in the higher F1-scores for the categories far before event time, with improvements of up to .30 for both category 1 and 4. As these early categories are most interesting given our goal to make close estimations early before event time, k-means is highly favorable over left branching.

With respect to the performance of the different systems we observe a similar pattern to the performance on left branching categories, with SVM ranking first, followed by Naive Bayes and k-NN.

To give an overview of the extent to which the different classifiers overshoot in their classification, by predicting a category highly distant in time from the actual distance, we plotted the distance errors on the k-means categories in Figure 2. The plot shows that the classification is skewed to the 'during' class. Naive Bayes achieves the most balanced performance, having the lowest distance errors for early classes while showing higher errors for categories 6 and 7, right before and during event time.

Table 4: F1-scores of the k-means categorization method. The small numbers represent the standard deviations.

Alg.	Feature Selection	before 0	1	2	3	4	5	6	during 7	after 8	MICRO-F1
k-NN	IG	.11 .04	.23 .06	.20 .02	.18 .03	.34 .03	.33 .02	.48 .01	.63 .03	.41 .02	.44 .02
k-NN	χ^2	.11 .04	.23 .06	.20 .02	.18 .03	.34 .03	.33 .02	.48 .01	.63 .03	.41 .03	.44 .02
NB	IG	.21 .05	.27 .08	.24 .03	.21 .04	.45 .05	.45 .03	.49 .01	.59 .02	.54 .04	.47 .02
NB	χ^2	.21 .06	.27 .09	.24 .03	.21 .04	.44 .05	.45 .03	.48 .01	.58 .02	.53 .04	.46 .02
SVM	IG	.25 .06	.35 .07	.30 .03	.29 .03	.56 .05	.43 .03	.55 .02	.72 .03	.56 .03	.52 .02
SVM	χ^2	.25 .06	.35 .07	.30 .03	.30 .05	.57 .06	.43 .04	.55 .02	.72 .03	.56 .03	.52 .02

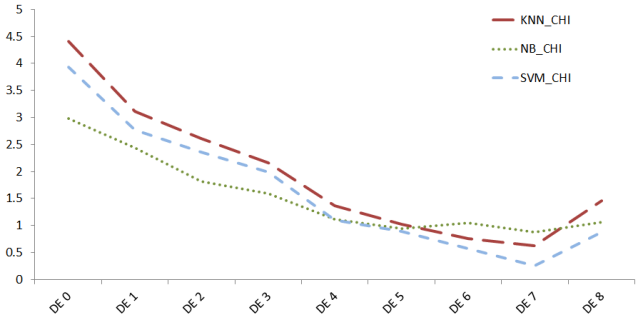

Figure 2: Distance errors for k-NN, NB, and SVM, using χ^2 and k-means

4.2 Human judgement

While our training and test data is based on 'perfect' hindsight information, one question is how well our systems performs in comparison to human judgements on estimating the time-to-event of single tweets. To have an indication, we carried out a small-scale experiment. We extracted 500 tweets randomly of the whole data set, and from those we picked 80 tweets spread more or less realistically in time, with at least 9 instances in each of 6 selectively chosen categories: 7 days or more before event time, 4 to 7 days before event time, 3 to 1 days before event time, the day of the event, during the event and after the event. We chose these day-based categories because they would better align with human intuition. The 8 participants (4 men and 4 women, with an average age of 28 and mean age of 24) had to decide for each tweet to which of the 6 categories it belonged. We additionally asked the participants how certain they were about their answer on a scale from 0 (certain) to 2 (uncertain).

The human performance was scored by calculating the average percentage correct for each category, the F1-score (based on the majority category given to each tweet), the distance error and average certainty of the participants for each category. Results are listed in Table 5.

Table 5: Comparing the prediction accuracy of humans vs. the best system

		> 7	4-7	1-3	0	during	after	average
Humans	Percentage correct	0.56	0.30	0.61	0.74	0.80	0.65	0.64
	F1-score	0.56	0.38	0.64	0.56	0.77	0.67	0.69
	Distance error	0.93	1.27	0.53	0.33	0.22	0.61	0.58
	Certainty	1.22	1.14	0.76	0.37	0.22	0.43	0.62
SVM,χ^2	F1-score	0.40	0.32	0.56	0.62	0.68	0.56	0.58
	Distance error	2.27	1.90	0.73	0.56	0.04	0.56	0.84

The results show that humans have more difficulty and are less certain in predicting the time-to-event when the event is still far away, especially for the penultimate time category. The distance errors are all below 1.3, which means that on average the predictions are maximally one category away from the correct category. Comparing the results to SVM with χ^2 feature selection (the best system in the main experiment) applied to these broad categories, we see that humans almost always outperform this system in terms of F1-scores and distance error. Especially the category > 7 days before event time is better predicted by humans. This can be explained by the fact that computers have to distinguish this category based on very few training data, whereas humans can rely on years of experience.

Looking at Table 6 and comparing human predictions to the predictions of our best system we can see that the RMSE of the human performance is considerably worse than the performance of this system, but that the responsiveness is much higher. The worse RMSE does not automatically mean that humans really did much worse. They classified the category > 7 days before event time more accurately, but this class has a very broad range (all tweets before 7 days or 168 hours) so the RMSE remains high.

Table 6: RMSE and responsiveness of the human experiment

	RMSE	responsiveness
SVM, χ^2	**62.1**	0.58
Humans	95.6	**0.92**

5 Discussion and conclusion

The scores for the different classes indicate that it is hard to predict an event that is still far away. There is a tendency to classify tweets with the most frequent 'during' class, which is reflected in the increasing

distance error with tweets posted longer before the event. There may be several factors at play. First, the fact that there is considerably less training data for these categories hinders their learnability and may bias the classifier to opt for the majority class in case of doubt. Second, earlier tweets may be more diverse in content. Tweets posted closer to the match often focus on the match, whereas tweets posted earlier deal with various other aspects of the event, such as buying tickets, logistics, inviting friends, etc. Third, temporal words are less precise when they refer to longer time spans. *Next week* can be 6 days away, or 10, whereas *tomorrow* is more precise. Finally people will use exact dates (*4th of November*) instead of phrases like *next week*. These exact dates are currently not recognized by the classifier. A relatively straightforward way to improve our system would be to have a first processing module that extracts these dates with language-specific regular expressions, and computes the time-to-event using these dates.

The method of time segmentation is important. We saw that the categorization using the k-means method produces better results than the left branching method. This can be explained by the fact that k-means clustering takes into account content-based characteristics of event-related tweets over time, instead of assuming some mathematical distribution of occurrences. In contrast, the results of the two feature selection methods were similar, although the two methods are certainly different. Apparently, our approach to the classification problem is robust against these feature selection methods.

In the future we plan to improve the 'before' segmentation by optimizing k in k-means clustering. Furthermore, as the influence of the 'during' category on classification performance was rather high in our experiment, we will apply new systems to lower the influence of this bias. Finally, while the current research was applied to tweets referring to football matches only, we will extend our approach to a wider variety of event types.

References

[1] Baharum Baharudin, Lam Hong Lee, and Khairullah Khan. A review of machine learning algorithms for text-documents classification. *Journal of Advances in Information Technology*, 1(1), 2010.

[2] T. M. Cover and P. E. Hart. Nearest neighbor pattern classification. *Institute of Electrical and Electronics Engineers Transactions on Information Theory*, 13:21–27, 1967.

[3] A. Hürriyetoglu, F. Kunneman, and A. van den Bosch. Estimating the time between twitter messages and future events. In *Proceedings of the 13th Dutch-Belgian Workshop on Information Retrieval (DIR 2013)*, number 986 in CEUR Workshop Proceedings, pages 20–23, 2013.

[4] F. Kunneman and A. Van den Bosch. Leveraging unscheduled event prediction through mining scheduled event tweets. In N. Roos, M. Winands, and J. Uiterwijk, editors, *Proceedings of the 24th Benelux Conference on Artficial Intelligence*, pages 147–154, Maastricht, The Netherlands, 2012.

[5] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA, 2008.

[6] Alan Ritter, Mausam, Oren Etzioni, and Sam Clark. Open domain event extraction from twitter. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '12, pages 1104–1112, New York, NY, USA, 2012. ACM.

[7] C.J. Van Rijsbergen. *Information Retrieval*. Buttersworth, London, 1979.

[8] W. Weerkamp and M. de Rijke. Activity prediction: A twitter-based exploration. In *SIGIR 2012 Workshop on Time-aware Information Access*, 08/2012 2012.

[9] Xindong Wu, Vipin Kumar, J. Ross Quinlan, Joydeep Ghosh, Qiang Yang, Hiroshi Motoda, GeoffreyJ. McLachlan, Angus Ng, Bing Liu, PhilipS. Yu, Zhi-Hua Zhou, Michael Steinbach, DavidJ. Hand, and Dan Steinberg. Top 10 algorithms in data mining. *Knowledge and Information Systems*, 14(1):1–37, 2008.