Reputation Profiling with GATE

Mark A. Greenwood, Niraj Aswani, and Kalina Bontcheva

Univeristy of Sheffield, Sheffield, UK
initial.surname@dcs.shef.ac.uk

Abstract. Brand management has become increasingly difficult in the age of social media as the volume of opinions and discussions surrounding a companies products have swelled beyond the scale at which they can be reviewed manually. In this paper we report details of a system for monitoring Twitter[1] to find tweets relevant to a specific entity and the classification of such tweets based upon their reputational effect. The system was evaluated as part of the recent RepLab 2012 profiling task and the results from this evaluation show that our system out performs a number of naïve baseline approaches.

1 Introduction

Brand management has always been an important part of any companies public relations stratergy, but with the recent explosion in social media it has become more and more difficult to exhaustively monitor the vast amount of information in a timely fashion. The solution is to employ automatic methods to monitor social media for opinions. Such algorithms usually involve two stages. Firstly relevant documents (tweets, forum posts, etc.) have to be identified before a second phase can determine the opinions contained within them. The RepLab 2012 profilling task accurately mirrors this situation using Twitter; the task consists of filtering tweets to determine those which are related to a given entity, and a polarity task in which the tweets are to be labelled according to their effect on brand reputation.

In this paper we describe the GATE [2] based approaches to filtering and polarity we submitted to RepLab 2012 for evaluation. We treated filtering and polarity classification as two independent tasks and as such the majority of this paper is divided into two sections which detail the approaches we developed before concluding with a discussion of the the evaluation results.

2 Relevance

The first stage in any approach to reputation profiling invovles determining which "documents" (be they tweets, forum posts, etc.) are actually relevant. For example, a tweet containing the word *Apple* might refer to the fruit[2], the well

[1] http://www.twitter.com
[2] http://en.wikipedia.org/wiki/Apple

known manufacturer of computers[3], or the Beatles record company[4] among a whole host of possibilities[5].

Our approach to determining relevance is based upon our recent research into disambiguation [3]. In this work we have been interested in determining which of a given set of DBpedia entries which share a common lexalization is actually being referred to. Similar to state-of-the-art methods, our algorithm uses the textual context, in which the particular candidate entity appears, in order to calculate a number of similarity metrics. In the current case this textual content includes the tweet itself, the expanded form of any hashtags[6] and any pages the tweet explicitly links to. Then an overall score is produced for each candidate URI, based on a weighted sum of the following similarity metrics:

- *String similarity*: edit distance between the text string (such as *Paris*), and the lexicalisations of the entity URIs (e.g. *Paris* and *Paris, Texas*).
- *Structural similarity*: calculated based on the ontology and instance property values in the Linking Open Data[7] (LOD) resource.
- *Contextual similarity*: calculated based on the probability that two words have a similar meaning, based on random indexing [8].
- *Commonness*: number of mentions of a specific URI as anchor text in Wikipedia (based on the commonness metric for Wikipedia pages [5], also referred to as popularity [7]). This is the equivalent to assigning the most frequent sense in word sense disambiguation. However, unlike [7], for efficiency we do not use Google queries as additional evidence.

Tie-breaks, i.e. candidate URIs with the same overall score, are resolved based on which one has the highest commonness score. If nevertheless more than one candidate remains, the instance which is more specific according to the LOD ontology is preferred.

Unfortunately, because of the nature of tweets, it often isn't possible to distinguish a noun (the fruit) from a proper noun (either of the two companies) due to the lack of case information etc. Due to this we assume that any reference to the entity, using any case, might be relevant and then disambiguate. This has the unfortunate side effect of assigning a URI to every mention. Our approach needs to be improved to incorporate the null assignment to handle cases where it is clear that the mention is not related to any of the known options.

3 Polarity

We treated the polarity task as a standard text classification problem and classified every tweet regardless of whether or not we deemed the tweet to be relevant[8].

[3] `http://en.wikipedia.org/wiki/Apple_Inc.`
[4] `http://en.wikipedia.org/wiki/Apple_Corps`
[5] `http://en.wikipedia.org/wiki/Apple_(disambiguation)`
[6] Hashtags were expanded via `http://tagdef.com/`
[7] `http://linkeddata.org/`
[8] This also makes sense given that in RepLab 2012 the two tasks are evaluated separately as well as in combination.

We experimented with two different classification approaches; k-Nearest Neighbours and Naïve Bayes. Both approaches utilised the same GATE pipeline for pre-processing of the tweets. The rest of this section is split into three parts. Firstly we outline the pre-processing pipeline and then we detail the two classifiers we built for this task.

3.1 Pre-Processing

Both approaches to polarity classification are based (primarily) upon simple tokenization of the individual tweets. We have implemented a simple GATE pipeline which performs tokenization, taking in to account a number of token types specific to the way in which tweets are often written. This pipeline consists of the following processing resources (PR):

- **Document Reset:** a standard GATE PR which simply deletes existing annotations, allowing an application to be run multiple times for development purposes.
- **Tokenizer:** the standard GATE UNICODE tokenizer.
- **Sentence Splitter:** As each document contains a single tweet, which we assume to be a single sentence, simply creates a single annotation spanning the entire tweet. When processing the training data features are created on each annotation recording the polarity and language of the tweet.
- **Hashtag Processor:** a simple JAPE grammar which recognises hashtags, ensuring that a single hashtag is represented as a single token and a HashTag annotation.
- **Emoticon Processor:** a combination of a gazetteer and a JAPE grammar which recognises emoticons, normalises them (see below) and ensures that they are represented as a single token and an Emoticon annotation.
- **Part-of-Speech Tagger:** a standard GATE PR which assigns part-of-speech (POS) tags to token annotations.

The only non-standard GATE component in this pipeline is the emoticon processor. The motivation behind this component is that with limited training data it is vital that we reduce, or eliminate, variation in expression of emoticons which may well be pivitol in conveying polarity. This normalization was performed by building a gazetteer of known emoticons[9]. Each entry in the gazetteer was paired with a normalized form of the emoticon. For example :-), :), and :] are all normalised to :). The JAPE grammar then ensures a single token annotation spans the emoticon using the normalised form rather than the underlying characters.

The result of running this pre-processing pipeline is a document annotated with a sequence of tokens which can be used by a machine learning algorithm to learn a polarity classifier. The same pipeline is also used to pre-process the unseen tweets before the classifiers are applied.

[9] `http://en.wikipedia.org/wiki/List_of_emoticons` was used as a starting point

3.2 *k*-Nearest Neighbours Classification

The standard GATE distribution provides a number of machine learning tools which can be used to perform text classification[10]. For these experiments into polarity classification we used the Batch Learning PR configured to perform *k*-Nearest Neighbours (*k*-NN) classification using an implementation from Weka[4]. Whilst space constraints preclude full details of the implementation (which can be found in [2]) the algorithm is configured to use the following features for learning: POS tags both 1-gram and 2-gram, emoticons, hashtags and the language of the tweet. Whilst it may seem strange that the words themselves (or at least their root forms) were not used to train the classifier, experimentation showed that including them led to a drop in performance of up to 5%. The reason behind this rather odd result is as yet unclear but may be related to both the small amount of training data (and hence only a small number of words occuring frequently) and the mixture of both English and Spanish text.

3.3 Naïve Bayes Classification

Whilst the machine leanring PRs provided with GATE are easy to use and highly configurable we decided to implement a second polarity classifier to allow for more fine grained control over the entire process. We chose to build a Naïve Bayes classifier following the example in [6]. This approach essentially reduces down to using uni-grams to classify text as the algorithm assumes that the probability of a word occuring is independent of it's position within the document.

Our specific implementation used lowercased versions of tokens as well as emoticons and hashtags as the input to the learning algorithm. In an attempt to improve classification due to the small amount of text present in the training tweets we also made use of the supporting documents (i.e. pages linked to from tweets). Rather than blindly including these pages in to the training set, the algorithm was tweaked to include the word counts but to ignore the document lengths. This simple change to the Naïve Bayes algorith was made in an attempt to not skew the distributions, especially given that we classified each tweet in the test set using the tweet alone, and these are of a common length (i.e. usually short and never more than 140 characters).

4 Results and Discussion

We submitted two runs for the profiling task; each run paired one of the polarity approaches with our disambiguation based approach to relevance filtering; `GATE_1` used the *k*-NN classifier for polarity classification while `GATE_2` used the Naïve Bayes classifier. Unfortunately due to an error in the script used to combine the approaches for `GATE_1` all tweets were classified as positive with respect to polarity. Whilst we list the results for this run below, we also detail a third submission, `GATE_3`, which we have evaluated (using the supplied gold standard)

[10] see `http://gate.ac.uk/userguide/chap:ml` for details

	Accuracy	R	S	F(R,S)
GATE	0.52	0.12	0.13	0.09
all relevant	0.71	0	0	0

Table 1. Relevance Filtering Results

	Accuracy	R	S	F(R,S)
GATE_1	0.44	0	0	0
GATE_2	0.33	0.27	0.28	0.26
GATE_3	0.41	0.25	0.21	0.22
All positive	0.44	0	0	0
All neutral	0.33	0	0	0
All negative	0.23	0	0	0

Table 2. Polarity Classification Results

independently. For comparison we have also included the baseline results. For full details of other submissions and the evaluation metrics used see the RepLab 2012 overview paper [1].

4.1 Relevance Filtering

As we submitted the same relevance judgements for each of our runs Table 1 shows just a single GATE run instead of duplicating the values. Note that the all relevant baseline has an accuracy of 0.71 which shows the large bias within the evaluation set towards relevant tweets. A similar bias can also be found within the training data which results in only a small amount of non-relevant examples. This bias may be a general occurance or may be due to the specific entities chosen for this evaluation, i.e. many of the entities do not actually require disambiguation.

4.2 Polarity Classification

Table 2 shows the results of our two attempts at polarity classification. Note that as previously mentioned we also report the results of the `GATE_3` run, which while not an officially evaluated run, shows how our `GATE_1` run should have performed.

We believe that relatively low performance of our classifiers is due to two things: the small amount of training data used and the difference in language use when expressing opinions across entities of different types. This second problem is probably more relevant than the lack of training data. During development we tested the algorithms using k-fold cross validation in two ways. In both cases we used six folds. One approach used the training data from one entity as a fold, and in the other data from all six entities were randomly split equally between the six folds. The average accuracy of the two approaches showed a difference of around 25%, with better performance being achieved when the folds were

	Accuracy
GATE_1	0.35
GATE_2	0.33
GATE_3	0.34
all relevant and positive	0.27
all relevant and neutral	0.26
all relevant and negative	0.18

Table 3. Combined Profilling Results

generated randomly. We belive that this is due to the fact that none of the six entities in the training data overlap in the products or services they provide and as such the language used to talk about them differs greatly. The six entities were:

- **Alcatel**: a provider of backend communications equipment, usually sold to governments or telecommunication companies rather than end-users
- **Apple**: a seller of consumer electronic goods including computers, phones and MP3 players
- **Armani**: a high-end fashion label
- **Barclays**: a British multinational banking and financial services company
- **Lufthansa**: the largest airline in Europe
- **Marriott**: a large chain of hotels and leisure resorts

As you can imagine complaining about a late flight (Lufthansa) would use very different language to complaints about short battery life in a consumer electronics product (Apple). This suggests that when building a classifier the training data should contain tweets about a variety of different entities.

4.3 Combined Profiling Performance

The results of combining our approaches to relevance filtering and polarity classification can be seen in Table 3. The combined evaluation was carried out by assuming a four class classification: not relevant, relevant and positive, relevant and neutral, and relevant and negative. As you can see from the results tables our combined approach to profilling out performed any of the three baseline systems included for comparison. While a system with accuracy of 0.35 (our best result obtained by the `GATE_1` run) can not be considered a strong performer we feel that it provides an ideal basis for our ongoing work in this area. As noted above we have already highlighted a number of areas where the approach could be improved and work is already going on to move the algorithms forward.

Acknowledgements

This research is partially supported by the EU-funded FP7 TrendMiner[11] project. Kalina Bontcheva is supported by the Engineering and Physical Sciences Research Council (grant EP/I004327/1).

References

1. Amigó, E., Corujo, A., Gonzalo, J., Meij, E., Rijke, M.d.: Overview of RepLab 2012: Evaluating Online Reputation Management Systems. In: CLEF 2012 Labs and Workshop Notebook Papers (2012)
2. Cunningham, H., Maynard, D., Bontcheva, K., Tablan, V., Aswani, N., Roberts, I., Gorrell, G., Funk, A., Roberts, A., Damljanovic, D., Heitz, T., Greenwood, M., Saggion, H., Petrak, J., Li, Y., Peters, W.: Text Processing with GATE (Version 6) (2011)
3. Damljanovic, D., Bontcheva, K.: Named Entity Disambiguation using Linked Data. In: Proceedings of the 9th Extended Semantic Web Conference (2012)
4. Frank, E., Hall, M., Holmes, G., Kirkby, R., Pfahringer, B., Witten, I.: Weka: A machine learning workbench for data mining. Data Mining and Knowledge Discovery Handbook: A Complete Guide for Practitioners and Researchers pp. 1305–1314 (2005)
5. Milne, D., Witten, I.H.: Learning to link with wikipedia. In: Proc. of the 17th Conf. on Information and Knowledge Management (CIKM). pp. 509–518 (2008)
6. Mitchell, T.: Machine Learning. McGraw-Hill, New York (1997)
7. Rao, D., McNamee, P., Dredze, M.: Entity linking: Finding extracted entities in a knowledge base. In: Multi-source, Multi-lingual Information Extraction and Summarization (2011)
8. Sahlgren, M.: An introduction to random indexing. In: Proc. of the Methods and Applications of Semantic Indexing Workshop. Copenhagen, Denmark (2005)

[11] `http://www.trendminer-project.eu/`