Umigon: sentiment analysis for tweets based on lexicons and heuristics

Clement Levallois
Department of Marketing Management, Rotterda School of Management
and Erasmus Studio, Erasmus University Rotterdam
The Netherlands.
clevallois@rsm.nl

Abstract

Umigon is developed since December 2012 as a web application providing a service of sentiment detection in tweets. It has been designed to be fast and scalable. Umigon also provides indications for additional semantic features present in the tweets, such as time indications or markers of subjectivity. Umigon is in continuous development, it can be tried freely at www.umigon.com. Its code is open sourced at: https://github.com/seinecle/Umigon

1. General principle of operation

Umigon belongs to the family of lexicon based sentiment classifiers (Davidov et al. 2010, Kouloumpis et al. 2011). It is specifically designed to detect sentiment (positive, negative or neutral) in tweets. The "sentiment detection engine" of Umigon consists of 4 main parts, which are detailed below:

- detection of semantic features in the entire tweet. Smileys and onomatopes are given special attention.
- evaluation of hashtags.
- decomposition of the tweet into a list of its n-grams (up to 4-grams), comparison of each n-gram with the terms in lexicons. In case of a match, a heuristic is applied.
- final series of heuristics at the level of the entire tweet, taking advantage of the semantic features detected in the previous steps. A final, unique sentiment (pos, neg or neut) is ascribed to the tweet.

2. The four steps of the classification engine

We refer in footnotes to the Java classes which implement the processes described here.

2.1 Global heuristics

Smileys and onomatopes carry strong indications of sentiment, but also come in a variety of orthographic forms which require methods devoted to their treatment[1].

Onomatopes and exclamations often include repeated vowels and consonants, as in yeaaaaahhhh (repeated "a" and "h"), but also yeaah (repeated "a"), or yeeeeaaaaah (repeated "e" and "a"). We list the most common exclamations and use regular expressions to capture the variety of forms they can assume. If such a form is found in the tweet, the related sentiment (positive or negative) is saved, and will be evaluated at a final stage for the global sentiment of the entire tweet.

Similarly, smileys are frequently spelled in multiple variations: :-) can also be found as :-)) or :-)))))))) . For this reason here also the flexibility of regular expressions is used to detect spelling variations. In addition, we consider that a smiley positioned at the very end of a tweet gives an unambiguous signal as to the sentiment of the tweet. For instance:

@mydearfriend You got to see Lady Gaga live, so lucky! Hate you :)))

Here, whatever the negative sentiments (Hate you) signaled in the tweet, the final smiley has an overriding effect and signals the strongest sentiment in the tweet. For this reason smileys located in final positions are recorded as such.

2.2 Evaluation of hashtags

Hashtags are of special interest as they single out a semantic unit of special significance in the tweet. Exploiting the semantics in a hashtag faces the issue that a hashtag can conflate several terms, as in #greatstuff or #notveryexciting. Umigon applies a series

[1] https://github.com/seinecle/Umigon/blob/master/src/java/Heuristics/SentenceLevelHeuristicsPre.java

of heuristics matching parts of the hashtag with lexicons[2]. In the case of #notveryexciting , the starting letters not will be identified as one of the terms in the lexicon for negative terms. Similarly, the letters very will be identified as one of the terms present in the lexicon for "strength of sentiment". exciting will be detected as one of the terms in the lexicon for positive sentiment. Taken together, not very exciting will lead to an evaluation of a negative sentiment for this hashtag. This evaluation is recorded and will be combined with the evaluation of other features of the tweet at a later stage.

2.3 Decomposition in ngrams

The text of the tweet is decomposed in a list of unigrams, bigrams, trigrams and quadrigrams. For example, the tweet This service leaves to be desired will be decomposed in list of the following expressions:

> "This, service, leaves, to, be, desired, This service, service leaves, leaves to, to be, be desired, This service leaves, service leaves to, leaves to be, to be desired, This service leaves to, service leaves to be, leaves to be desired"

The reason for this decomposition is that some markers of sentiment are contained in expressions made of several terms. In the example above, to be desired is a marker of negative judgment recorded as such in the lexicon for negative sentiment, while desired is a marker of positive sentiment. Umigon loops through all the n-grams of the tweet and checks for their presence in several lexicons[3].

If an n-gram is indeed found to be listed in one of the lexicons, the heuristic attached to this term in this lexicon is executed, returning a classification (positive sentiment, negative sentiment, or another semantic feature). Heuristics attached to terms in the lexicons are described in detail in section 3.

2.4 Post-processing: a last look at the entire tweet .

At this stage, the methods described above may have returned a large number of (possibly conflicting) sentiment categories for a single tweet. For instance, in the example This service leaves to be desired, the examination of the n-grams has returned a positive sentiment classification (desired) and also negative (to be desired). A series of heuristics adjucates which of the conflicting indications for sentiments should be retained in the end. In the case above, the co-presence of negative and positive sentiments without any further indication is resolved as the tweet being of a negative sentiment. If the presence of a moderator is detected in the tweet (such as but, even if, though), rules of a more complex nature are applied[4].

3. A focus on lexicons and heuristics

Four lexicons are used for sentiment analysis (number of terms in the lexicons in brackets): "positive tone" (332), "negative tone" (630), "strength of sentiment" (59), "negations" (45). These lexicons have been created manually by the inspection of thousands of tweets, and continue to be expanded on a regular basis. Note that the same term can appear in different lexicons (if rarely in practice). For example, the term fucking appears in the lexicon for negative tone and in the lexicon for strong sentiments. Each term in a lexicon is accompanied by a heuristics and a decision rule.

3.1 Simple case from the "negative sentiments" lexicon:

Term	sadfaced
Heuristics	None
Decision Rule	012

If a tweet contains the term sadfaced, Umigon will directly add the code "012" (which stands for negative sentiment) to the tweet[5].

3.2 More complex case from the "positive sentiments" lexicon:

Term	Satisfied
Heuristics	!isImmediatelyPrecededBy ANegation
Decision Rule	011|012

If the term satisfied is present in a tweet, the heuristics !isImmediatelyPrecededByANegation is applied. This s a method checking whether the term immediately

[2] https://github.com/seinecle/Umigon/blob/master/src/java/Heuristics/HashtagLevelHeuristics.java

[3] https://github.com/seinecle/Umigon/blob/master/src/java/Classifier/ClassifierMachine.java

[4] https://github.com/seinecle/Umigon/blob/master/src/java/Heuristics/SentenceLevelHeuristicsPost.java

[5] See this class for the full list of possible classifications: https://github.com/seinecle/Umigon/blob/master/src/java/Classifier/Categories.java

preceding satisfied in the tweet is a negation or not[6]. This method returns a Boolean (true / false). The Boolean returned by this heuristics will determine the outcome of the decision rule. Here, the decision rule is a simple binary choice: codify as 011 (meaning, a positive sentiment) if satisfied is not preceded by a negation; codify it as 012 (negative sentiment) otherwise.

3.3 Complex case from the "negative sentiments" lexicon:

Term	hard
Heuristics	!isImmediatelyPrecededByANegation+++!isImmediatelyFollowedBySpecificTerm///work|disk
Decision Rule	A?(B?(012):011)

This example shows how several heuristics (separated by +++) can be combined, leading to complex rules of decision. In this example, whenever the term hard is detected in a tweet, 2 heuristics are evaluated: is the term preceded by a negation? Is the term followed by specific terms – work or disk, in this case? Each of these heuristics returns a Boolean. The Booleans are fed into the interpreter of the decision rule, where A and B represent the 2 Booleans[7]. Depending on their value, the decision tree takes a different branch, leading to the selection of one codification. In the example:
If A is false, return 011: a positive sentiment.
Example: not hard
If A is true and B is true, return 012: a negative sentiment. Example: it is hard
If A is true and B is false, returns null: nothing (a neutral sentiment).
Example: this is a hard disk

While in practice it is rarely needed to write up rules of such complexity, they offer an extra flexibility to exploit the semantic features of terms in varying contexts.

[6] The method actually checks the two terms before, in order to capture cases such as "not very satisfied", where a negative term is present but not immediately preceding the term under review. See the details of all heuristics here: https://github.com/seinecle/Umigon/blob/master/src/java/Heuristics/Heuristic.java

[7] The class for the interpreter is: https://github.com/seinecle/Umigon/blob/master/src/java/RuleInterpreter/Interpreter.java

4. Performance

4.1 Accuracy

Umigon was formally evaluated in a semantic evaluation task proposed by SemEval-2013, the International Workshop on Semantic Evaluation (Wilson et al., 2013). The task consisted in classifying 3,813 tweets as positive, negative or neutral in polarity (task B). The results:

class	Pos	neg	neut
prec	0.7721	0.4407	0.6471
rec	0.5604	0.5507	0.7579
fscore	0.6495	0.4896	0.6981
average(pos and neg)	0.5696		

For reference, the best performing participant in this task obtained the following results (Mohammad et al., 2013):

class	pos	neg	neut
prec	0.8138	0.6967	0.6765
rec	0.6673	0.604	0.8262
fscore	0.7333	0.6471	0.7439
average(pos and neg)	0.6902		

We see that Umigon had an especially poor precision for tweets of a negative sentiment (results greyed in the table). This means that Umigon failed to identify many negative tweets as such. One reason accounting for this poor performance is the definition we adopt for what a negative sentiment is. For example, the SemEval task included this negative tweet:
"Renewed fighting rocks Syria: An early morning explosion rocked the flashpoint city of Deir Ezzor on Saturday in..."
By design, Umigon has not been conceived to classify such a tweet as negative because if it contains negative elements of a *factual* nature (explosion, fighting), but contains no marker of a negative *attitude*. This question aside, the accuracy of Umigon should be improved by increasing the number of terms and heuristics in the lexicons, which is an ongoing process.

4.2 Speed

Tested on a dataset provided by sentiment140.com[8], Umigon performs the classification of 1.6 million tweets in less than 15 minutes. We believe that not relying on Part of Speech tagging makes it a specially

[8] http://help.sentiment140.com/for-students

fast solution for lexicon-based sentiment classifiers. The classifier engine is implemented in such a way that the presence of absence of n-grams in the terms lists is checked through look-ups on hashsets (is this n-gram contained in a set?), not loops through these sets. Since look-ups in hashsets is typically of O(1) compexity[9], this insures that the performance of Umigon will not degrade even with expanded lexicons.

References

Davidov, D., Tsur, O., and Rappoport, A. 2010. Enhanced sentiment learning using twitter hashtags and smileys. Proceedings of Coling.

Kouloumpis, E., Wilson, T., and Moore, J. 2011. Twitter Sentiment Analysis: The Good the Bad and the OMG! Proceedings of ICWSM.

Mohammad, S., Kiritchenko, S. and Zhu, X. 2013. NRC-Canada: Building the State-of-the-Art in Sentiment Analysis of Tweets. In *Proceedings of the International Workshop on Semantic Evaluation*, SemEval '13, June 2013, Atlanta, Georgia.

Wilson, T., Kozareva, Z., Nakov, P., Rosenthal, S. Stoyanov, V. and Alan Ritter. 2013. SemEval-2013 Task 2: Sentiment Analysis in Twitter. In *Proceedings of the International Workshop on Semantic Evaluation*, SemEval '13, June 2013, Atlanta, Georgia.

[9] http://stackoverflow.com/questions/6574916/hashset-look-up-complexity