QUB at SemEval-2017 Task 6: Cascaded Imbalanced Classification for Humor Analysis in Twitter

Xiwu Han and Gregory Toner
Queen's University Belfast
Belfast, BT7 1NN, UK
x.han@qub.ac.uk; g.toner@qub.ac.uk

Abstract

This paper presents our submission to SemEval-2017 Task 6: #HashtagWars: Learning a Sense of Humor. There are two subtasks: A. Pairwise Comparison, and B. Semi-Ranking. Our assumption is that the distribution of humorous and non-humorous texts in real life language is naturally imbalanced. Using Naïve Bayes Multinomial with standard text-representation features, we approached Subtask B as a sequence of imbalanced classification problems, and optimized our system per the macro-average recall. Subtask A was then solved via the Semi-Ranking results. On the final test, our system was ranked 10th for Subtask A, and 3rd for Subtask B.

1 Introduction

Humor is an essential trait of human intelligence that has not yet been addressed extensively in current AI research[1]. It's certainly one of the most interesting and puzzling research areas in the field of natural language understanding, and developing techniques that enable computers to understand humor in human languages deserves research attention (Yang et al., 2015).

Humor recognition or analysis by computers aims to determine whether a sentence in context expresses a certain degree of humor. This can be extremely challenging (Attardo, 1994) because no universal definition of humor has been achieved, humor is highly contextual, and there are many different types of humor with different characteristics (Raz, 2012). Previous studies (Mihalcea and Strapparava, 2005; Yang et al., 2015; Zhang and Liu, 2014; Purandare and Litman, 2006; Bertero and Fung, 2016) dealt with the humor recognition task as a binary classification task, which was to categorize a given text as humorous or non-humorous (Li et al., 2016). Textual data consisting of comparable amounts of humorous texts and non-humorous texts were collected, and a classification model was then built using textual features. Barbieri and Saggion (2014) examined cross-domain application of humor detection using Twitter data. Purandare and Litman (2006) used data from a famous TV series, *Friends*. Speakers' turns which occurred right before simulated laughter were defined as humorous ones and the other turns as non-humorous ones. They also used speakers' acoustic characteristics as features. Bertero and Fung (2016) pursued a similar hypothesis. Their target was to categorize an utterance in a sitcom, *The Big Bang Theory*, into those followed by laughter or not. They were the first to use a deep learning algorithm for humor classification. Besides, because genre bias can be problematic, Yang et al. (2015) tried to minimize genre differences between humorous and non-humorous texts.

SemEval-2017 Task 6 aims to encourage the development of methods that should take into account the continuous nature of humor, on the one hand, and to characterize the sense of humor of a particular source, on the other. The dataset was based on humorous responses submitted to a Comedy Central TV show *@midnight*[2]. There are two subtasks: A. Pairwise Comparison, where a successful system should be able to predict among a pair of tweets which is funnier; and B. Semi-Ranking, where, given a file of tweets for a hashtag, systems

[1] http://alt.qcri.org/semeval2017/task6/

[2] http://www.cc.com/shows/-midnight

Proceedings of the 11th International Workshop on Semantic Evaluations (SemEval-2017), pages 380–384,
Vancouver, Canada, August 3 - 4, 2017.

should produce a ranking of tweets from funniest to least funny.

Since automatic humor analysis is difficult, our goal is only to provide computer assistance to human experts. We approached Subtask B as a sequence of imbalanced classification problems, and optimized our system per the macro-average recall. Subtask A was then solved simply via the Semi-Ranking results of Subtask B.

2 Data and Our Features

The training and trial data consists of 106 files, and the test data consists of 6 files. Each file corresponds to a single hashtag, and is named accordingly. For example, for the hashtag #DogSongs, the file is called Dog_Songs. The tweets are labeled 0, 1, or 2. 0 corresponds to a tweet not in the top 10 (i.e. not considered funny). 1 corresponds to a tweet in the top 10, but not the funniest tweet. 2 corresponds to the funniest tweet. Figure 1 shows the distribution of these three classes on the training and trial data sets. This is unlike existing relevant research, which involved comparable amounts of humorous and non-humorous texts.

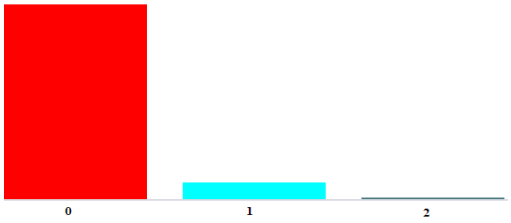


Figure 1: Imbalanced distribution of humor degrees on the training and trial data sets.

However, this distribution might be more pragmatic for analyzing humor in real life languages. Though humor is as important to our life as is spice to our food[3], it's just a small part of the whole matter. The distribution of humor vs. non-humor might be naturally imbalanced. The nature of Subtask A is also imbalanced. Although only tweet pairs with differing humor degrees are evaluated in the final test, those pairs with the same degree of humor will occupy by far the larger proportion of all tweet pairs for a given hashtag. We chose to solve Subtask A simply from ranking results of Subtask B.

A better semantic understanding of the hashtag will contribute to a better performance in the task. For example, named entities obviously form an important part of contextual knowledge. The task organizers allow participants to manually annotate the trial and training data, such as annotating the proper nouns referenced in a tweet. However, the automatic annotating performance could be unreliable and be detrimental to the hashtag understanding. Besides, the manual annotation of around ten thousand tweets is not a trivial task. Therefore, we only included the tweets and the relevant hashtags for classification features. They were regarded as two textual features, with the hashtag parsed into a sequence of words.

3 Our Approach

We first focused on Subtask B, solving it by cascaded imbalanced classification. In our daily life, there exists similar imbalanced distribution to that shown in Figure 1, such as the World Cup and beauty contests. In such cases, there can be n predefined levels or ranks, the number of participants or survivors allowed for each higher rank is usually exponentially smaller than its lower ranks. In a cascaded way, a such n-rank machine learning task could be solved by $n-1$ imbalanced classifiers.

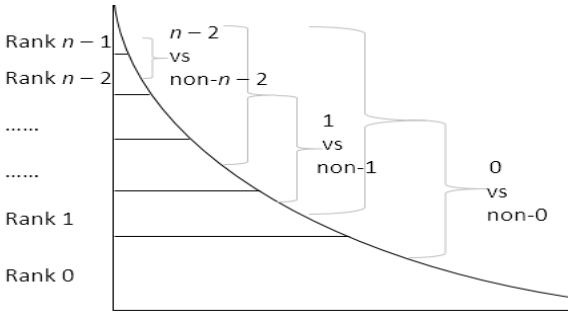


Figure 2: Cascaded imbalanced classification.

The cascaded method is illustrated in Figure 2, and a pseudo algorithm for training the classifiers, classifying a query, and semi-ranking, is detailed in Table 1. Each binary imbalanced classifier in the cascade is trained to distinguish the data points of one rank and those of its higher ranks, while data points in any lower ranks are not counted in. All the $n-1$ classifiers work one by one in the ranking order to complete the election or filtering process.

For imbalanced classification, there are many existing solutions (Kotsiantis et al., 2006; Sun et al., 2009; Galar et al., 2012). The solutions can be based on data resampling, algorithm adjustment, cost-sensitive tuning, boosting approaches, or hybrid methods. Since humor distribution might be naturally imbalanced, we chose to tune the classifying cost matrix and prediction confidence.

[3] http://www.aath.org/humor-the-spice-of-life.

Input: Training set $\boldsymbol{X}$ of data point $\boldsymbol{x}$ labeled with n ranks {0, 1, …, n – 1}, and a test set $\boldsymbol{T}$ of data point $\boldsymbol{t}$ with no labels.
Output: Set $\boldsymbol{F}$ of n – 1 imbalanced classifiers, and semi-ranked $\boldsymbol{T}$ of data point $\boldsymbol{t}$ labeled with ranks {0, 1, …, n – 1}.
1. **for** rank r (0 <= r < n – 1), remove from $\boldsymbol{X}$ any $\boldsymbol{x}$ labeled with r' (r' < r). 2. Re-label any $\boldsymbol{x}$ within rank r'' (r''>r) as *non-r*. 3. Train an imbalanced binary classifier $f_r(\boldsymbol{x})$ on the re-labeled $\boldsymbol{X}$, and set $\boldsymbol{F}$ as $F \bigcup \{f_r(x)\}$. 4. **end for** 5. Given $\boldsymbol{t}$ in $\boldsymbol{T}$, **for** 0 <= r < n – 1, apply $f_r(\boldsymbol{t})$, **end for**. 6. Label $\boldsymbol{t}$ with r^*, the highest rank predicted. 7. Set the ranking score of $\boldsymbol{t}$ as $r^* + C_{fr^*(t)}$, where $C_{fr^*(t)}$ is prediction confidence of $f_{r^*}(\boldsymbol{t})$. 8. Sort $\boldsymbol{T}$ per the ranking scores, and return $\boldsymbol{T}$.

Table 1: Algorithm for solving Subtask B.

A cost matrix is used to represent the differing cost of each type of misclassification (Elkan, 2001). Typically, each row in the matrix is used to represent the predicted label and each column corresponds to the actual label of gold standard. The matrix entry $\boldsymbol{C}_{ij}$ is the cost of predicting the ith label when the jth label is actually correct. In general, $\boldsymbol{C}_{ij} > \boldsymbol{C}_{jj}$ when $i \neq j$, i.e. a correct prediction is less costly than an incorrect prediction. Usually the entries $\boldsymbol{C}_{jj}$ along the main diagonal will all be zero.

For a classifier that can output the full probability distribution over all class labels, prediction confidence is defined as the difference between the estimated probability of the true class and that of the most likely predicted class other than the true class. By tuning prediction confidence for one class, we can easily balance the weight distribution between this class and other classes. Tuning the cost matrix and prediction confidence could be done via optimizing a given performance measurement on a held-out development set or by cross-validation. Since our goal is to provide computer assistance to human experts in humor analysis, we chose macro-average recall as the performance measurement to be optimized. The parameters for imbalanced classification could be tuned in a pipeline way, i.e. for each classifier $f_r(\boldsymbol{x})$ we first tuned the cost sensitive matrix and then tuned the prediction confidence.

Though Subtask A aims to predict among a pair of tweets which is funnier, its evaluation requires a system to return all tweet pairs with different humor degrees for a given hashtag. More generally, a pairwise comparison problem with n predefined ranks of data points could be solved simply by the algorithm in Table 2, once the semi-ranking results have been obtained for Subtask B.

Input: Semi-ranked set $\boldsymbol{T}$ of data point $\boldsymbol{t}$ labeled with ranks {0, 1, …, n – 1}.
Output: Set of tweet pairs $\boldsymbol{P}$ = {($\boldsymbol{t}_i$, $\boldsymbol{t}_j$)| $i > j$, 0 < i < n, 0 <= j < n – 1}.
1. **for** $i = n - 1$; $i > 0$: 2. **for** $j = n - 2$; $j >= 0$: 3. $P = P \bigcup \{(t_i, t_j)\}$. 4. **end for** 5. **end for** 6. Return $\boldsymbol{P}$.

Table 2: Algorithm for solving Subtask A.

The algorithm in Table 2 depends on the predicted ranks, thus it will result in better recall of data pairs with different ranking degrees, and human experts will have more choices. However, better precision or F-measure could be achieved by exploiting the semi-ranking order and limiting the number of data points in each rank as required by the concrete task. For example, the number in rank 2 is 1, and in rank 1 it is 9 for a given hashtag in SemEval-2017 Task 6.

4 Experiment and Results

This task is a 3-partite problem that could be solved via the algorithms given in Table 1 and 2. Using Java and Naïve Bayes Multinomial (NBM) classification of Weka 3.7[4] (Witten et al, 2011), we did the experiment with the training and trial data as training set. As for classification features, our present research simply chose word n-grams with n = 1, 2, and 3. By optimizing the macro-average recall of an NBM classifier on the training set with all original class labels, 3200 word types were kept before vectorization. Figure 3 gives some results of a part of the optimizing process. The star denotes the optimized point. For tuning the cost matrix and prediction confidence, we used 10-fold cross validation. The parameter values of the largest macro-average recall and the least standard deviation were returned for training final NBM classifiers on the whole training set and predicting for the final test set.

Table 3 lists the results of tuning cost matrix and prediction confidence. We first tuned the cost matrices, and the best macro-average recalls are marked with [1] in Table 3.

[4] http://www.cs.waikato.ac.nz/ml/weka/downloading.html

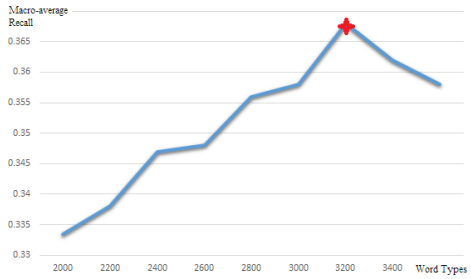
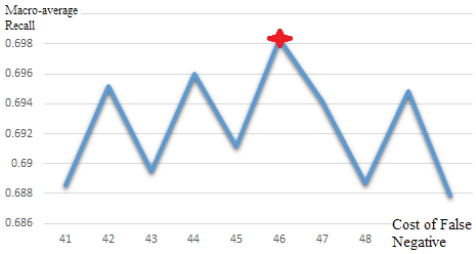


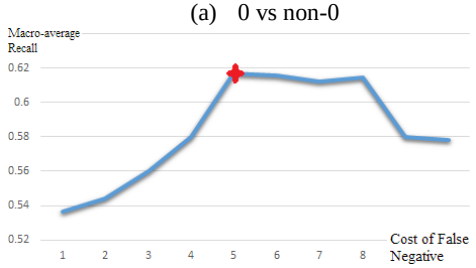
Figure 3: Selecting word n-gram types before vectorization.

NBM Classifiers	0 vs non-0	1 vs non-1
Positive Class	0	1
Negative Class	1	2
Cost of False Positive	1	1
Cost of False Negative	46	5
Macro-average Recall[1]	$0.698_{\pm 0.11}$	$0.617_{\pm 0.14}$
Negative Confidence	0.01	0.96
Macro-average Recall[2]	$0.713_{\pm 0.09}$	$0.623_{\pm 0.12}$

Table 3: Tuning results for cost matrix and prediction confidence



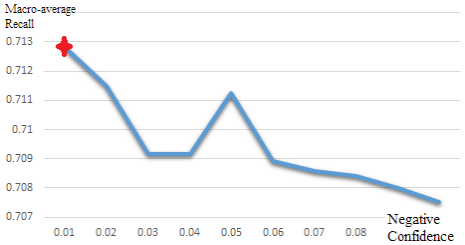
(a) 0 vs non-0



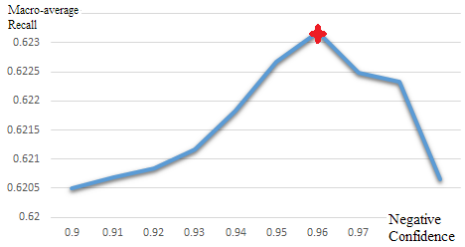
(b) 1 vs non-1

Figure 4: Tuning the cost of false negative.

Figure 4 gives parts of the cost matrix tuning process nearby the optimization points (denoted with stars). To make the tuning less expensive, we fixed the cost for false positive as 1, and only tuned the cost for false positive. Then, with the optimized cost matrices for NBM classifiers, we tuned the confidence for predicting negative items, and the best macro-average recalls are marked with [2] in Table 3. Figure 5 gives parts of the prediction confidence tuning process near the optimization points (denoted with stars). We finally trained our system on the whole training set with the tuned parameters, and applied this system on the evaluation set. For Subtask A, our submission is ranked 10th, with a micro-averaged accuracy of 0.187. For Subtask B, our submission is ranked 3rd, with an edit distance of 0.924.



(c) 0 vs non-0



(d) 1 vs non-1

Figure 5: Tuning prediction confidence

5 Conclusion

For detecting humor, we assume that the distribution of humorous and non-humorous texts in a language is naturally imbalanced. Instead of aiming at an automatic humor analysis system, our goal for solving SemEval-2017 Task 6 is to provide computer assistance to human experts. Therefore, macro-average recall was employed as the major measurement for training. We approached Subtask B as a sequence of imbalanced classification problems, and optimized our system per the macro-average recall. Subtask A was then solved via the Semi-Ranking results. In future research, we plan to employ more classification features and other imbalanced machine learning techniques.

Acknowledgments

This research is sponsored by the Leverhulme Trust project with grant number of RPG-2015-089.

References

Salvatore Attardo. 1994. *Linguistic theories of humor*, volume 1. Walter de Gruyter.

Francesco Barbieri and Horacio Saggion. 2014. Automatic detection of irony and humour in Twitter. *Proceedings of the Fifth International Conference on Computational Creativity*, Ljubljana, Slovenia, jun. Josef Stefan Institute.

Dario Bertero and Pascale Fung. 2016. A long short-term memory framework for predicting humor in dialogues. *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 130-135, SanDiego, California, June.

C. Elkan. 2001. The Foundations of Cost-sensitive Learning. *International Joint Conference on Artificial Intelligence*, LAWRENCE ERLBAUM ASSOCIATES LTD, 17.1: 973-978.

M. Galar, A. Fernandez, E. Barrenechea, H. Bustince, and F. Herrera. 2012. A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42(4), 463-484.

Sotiris Kotsiantis, Dimitris Kanellopoulos, and Panayiotis Pintelas. 2006. Handling imbalanced datasets: A review. *GESTS International Transactions on Computer Science Engineering*, 30.1: 25-36.

Jiwei Li, Xinlei Chen, Eduard H. Hovy, and Dan Jurafsky. 2016. Visualizing and understanding neural models in NLP. *The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, San Diego California, USA, 681-691.

Rada Mihalcea and Carlo Strapparava. 2005. Making computerslaugh: Investigationsinautomatichumor recognition. *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing*, 531- 538. Association for Computational Linguistics.

Amruta Purandare and Diane Litman. 2006. Humor: Prosody analysis and automatic recognition for f*r*i*e*n*d*s*. *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing, EMNLP '06*, 208-215, Stroudsburg, PA. Association for Computational Linguistics.

Yishay Raz. 2012. Automatic humor classification on twitter. *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Student Research Workshop*, 66-70. Association for Computational Linguistics.

Yanmin Sun, Andrew KC Wong, and Mohamed S. Kamel. 2009. Classification of imbalanced data: A review. *International Journal of Pattern Recognition and Artificial Intelligence*, 23.04: 687-719.

Ian H Witten, Eibe Frank, and Mark A Hall. 2011. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, Third edition, ISBN: 978-0-12-374856-0.

Diyi Yang, Alon Lavie, Chris Dyer and Eduard Hovy. 2015. Humor Recognition and Humor Anchor Extraction. *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 2367-2376, Lisbon, Portugal.

Renxian Zhang and Naishi Liu. 2014. Recognizing humor on twitter. *Proceedings of the 23rd ACM International Conference on Information and Knowledge Management*, 889-898. ACM.