TakeLab at SemEval-2017 Task 6: #RankingHumorIn4Pages

Marin Kukovačec, Juraj Malenica, Ivan Mršić, Antonio Šajatović,
Domagoj Alagić, Jan Šnajder
Text Analysis and Knowledge Engineering Lab
Faculty of Electrical Engineering and Computing, University of Zagreb
Unska 3, 10000 Zagreb, Croatia
{name.surname}@fer.hr

Abstract

This paper describes our system for humor ranking in tweets within the SemEval 2017 Task 6: #HashtagWars (6A and 6B). For both subtasks, we use an off-the-shelf gradient boosting model built on a rich set of features, handcrafted to provide the model with the external knowledge needed to better predict the humor in the text. The features capture various cultural references and specific humor patterns. Our system ranked 2nd (officially 7th) among 10 submissions on the Subtask A and 2nd among 9 submissions on the Subtask B.

1 Introduction

While extremely interesting, understanding humor expressed in text is a challenging natural language problem. Besides standard ambiguity of natural language, humor is also highly subjective and lacks an universal definition (Mihalcea and Strapparava, 2005). Moreover, humor should almost never be taken at face value, as its understanding often requires a broader context – external knowledge and common sense. On top of that, what is funny today might not be funny tomorrow, as humor goes hand in hand with ever-changing trends of popular culture.

Even though there has been some work on humor generation (Petrović and Matthews, 2013; Valitutti et al., 2013), most work has been concerned with humor detection, a task of classifying whether a given text snippet is humorous (Mihalcea and Strapparava, 2005; Kiddon and Brun, 2011; Yang et al., 2015; Chen and Lee, 2017). However, this research was mostly focused on a simple binary detection of humor.

In this paper, we describe a system for ranking humor in tweets, which we participated with in the SemEval-2017 Task 6 (Potash et al., 2017). It comprised two subtasks, one dealing with predicting which tweet out of two is more humorous, and other with ranking a set of tweets by their humorousness. Even though these tasks can be both posed and tackled differently, we straightforwardly used the obtained pairwise classifications from the first task in coming up with ranked lists for the second. Our system uses a standard gradient boosting classifier (GB) based on a rich set of features and collections of external knowledge. We ranked 2nd among 10 submissions (7th officially) at the Subtask 6A, and 2nd among 9 submissions at the Subtask 6B.

2 Task Description

The dataset provided by the task organizers comprises the tweets collected from many episodes of the Comedy Central show @midnight.[1] This game show is based around contestants and viewers providing humorous and witty tweets in response to a given topic (hashtag), which are then ranked by their humorousness. The compiled dataset consists of 11,685 tweets grouped into 106 hashtags. Each group is further split into three bins: the most humorous tweet (denoted 1), nine less humorous tweets (denoted 2), and a varying number of the least humorous tweets (denoted 0). The test set consists of 749 tweets grouped into 6 hashtags not present in the train set.

The task is divided into subtasks 6A and 6B. In the first subtask, participants must recognize the more humorous tweet of the two, whereas the second subtask asks for a complete tripartite ranking of all tweets under a given hashtag. This basically means that the order of the tweets is not important as long they are placed in the correct bin. For more details consult (Potash et al., 2017).

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

Proceedings of the 11th International Workshop on Semantic Evaluations (SemEval-2017), pages 396–400,
Vancouver, Canada, August 3 - 4, 2017.

3 Model

We tackle both subtasks with a single base model. For the subtask A, we trained a model that predicts which tweet of the given two is more humorous, and then used this information to rank the tweets for the subtask B. More specifically, we counted how many times a tweet was more humorous than other tweets, and ranked the tweets by that number. Note that this resulted in a complete ranking, which we reorganized into bins (which is possible as the cardinality of the two out of three bins is known). In the following sections, we describe our rich set of features and model optimization.

3.1 Features

We used Twitter-specialized preprocessing tools. More precisely, we first tokenize the dataset and then obtain the part-of-speech (POS) tags with Twokenizer.[2] This tool also accounts for the normalization of the elongated vowels (e.g., "heeeeello"→"hello"). For dependency parsing, we use TweeboParser.[3] We lemmatize the obtained words using the NLTK toolkit (Bird et al., 2009). In the end, we obtain a 140-dimensional feature vector of a tweet.[4] Below we describe the features grouped into categories (number of features per group given in parentheses).

Cultural reference features (96). By inspecting the dataset, we noticed that, at least within the @midnight game show, most jokes are based solely on taking the names of famous people, movies, TV series, and other culture references, and modifying them in an unexpected, yet humorous way. To this end, we acquired a number of collections covering such references, so that our model could recognize them within the tweets. The collections comprise, among others, movie titles, song names, book titles, TV series titles, cartoon titles, people names and their professions, nationality, and birth year (Yu et al., 2016) (Table 1). To obtain the features, we first calculate the tf-idf-weighted bag-of-words (BoW) vectors of all tweets and all items in the acquired collections. Then, for each collection, we construct a single feature that denotes the maximum cosine similarity between a given tweet's vector and those of the items from the collection. We also construct a one-hot-encoded vector of professions of a person mentioned in the text (the resource covers 88 different occupations). In the case no person is mentioned in a tweet, this vector is set to a zero vector. Additionally, we specifically check whether there is a USA citizen mentioned in a tweet and fetch an average Google Trends[5] rank of all the named entities found within a tweet.

Collection	Number of items
Movie titles	6,609
Song names	3,820
Book titles	191
TV series titles	228
Cartoon titles	183
People information	10,951
One-line jokes	2,868
Curse words	165

Table 1: External knowledge collections we used in the model.

Binary and count-based features (15). Besides simple tweet length in characters, we also measure the common noun, proper noun, pronoun, adjective, and verb to token ratios. Analogously, we measure the punctuation count and punctuation to character ratio. Besides this, we detect whether time and place deixes occur in a tweet (Zhang and Liu, 2014) using a precompiled list of deixes. Furthermore, we used binary features that denote whether the tweet contains a exclamation mark, negation, hashtag, or a URL. On top of these features, we calculated how much times single words are repeated in a tweet.

Readability-based features (3). Our intuition tells us that good jokes "flow": they are catchy and easily pronounceable. To capture this, we used the features that gauge tweet readability. First, we employ the Flesch reading-ease test (Flesch, 1948). Secondly, we follow the work of Zhang and Liu (2014) and extract all the alliteration chains (sequences of at least two words that start with the same phone). We construct the alliteration feature as a total length of alliteration chains divided by the number of tweet tokens. Lastly, we measure the vowels to characters ratio.

[2] https://github.com/brendano/ark-tweet-nlp/

[3] https://github.com/ikekonglp/TweeboParser

[4] Note that this has nothing to do with the character limit on Twitter, which is coincidentally also 140.

[5] https://trends.google.com/trends/

Humor-specific features (25). As the simplest feature in this category, we measure the number of tokens between the root of a dependency-parsed sentence and furthest node pointing to it. With this feature we hope to capture punchlines, whose common characteristic is that they are usually found at the end of a sentence.

Additionally, by examining the dataset, we noticed that humor often arises from the use of a modifier that does not seem to fit with the word it modifies. For instance, in the case of "Conan the Apologizer" (orig. "Conan the Barbarian"), the tweet is humorous because Conan is never attributed with such a trait, as he never apologizes. To detect this disparity, we measure the cosine similarity of skip-gram (SG) embeddings (Mikolov et al., 2013) between certain parts of a tweet: the root and the subject, the root and the object, and the root and all of its modifiers. In the last case, we sum all the similarities. We use the freely-available pre-trained vectors.[6]

To detect puns, we use a simple heuristic – a tweet contains a pun if it matches in all but one word with any of the items from our collections. We also acquired a collection of one-line jokes and curse words (Table 1). For one-line jokes, we realized that it would be unreasonable to expect having a complete joke within a tweet. To account for this, we simply counted how many words from the collection of one-line jokes are present in a tweet. This way we hope to capture the characteristic words found in one-line jokes. Lastly, we map the tweet's hashtag to a predefined set of humor patterns, which we manually compiled out of all the hashtags from the dataset: *Movie*, *Song*, *Book*, *Cartoon*, *Show*, *Sci*Fi*, *Celeb*, *Food*, **Words*, *Add**, *Make**, *If**, *Before**, **Because*, *One*letter*, *Ruin**, *Sexy**, *y** (where * denotes a wildcard). We construct a one-hot-encoded vector of these patterns, which is set to zero if a new hashtag could not match to any of these patterns.

Sentiment-based features (1). We used the output of model for sentiment classification of tweets (Lozić et al., 2017) as one of our features.

3.2 Model Optimization

Considering that we tackle both subtasks using a binary classification model, we construct the instances by concatenating feature vectors of both tweets in a pair. The pairs are constructed as a Cartesian product between all the tweets in all different bin pairings. Note that this results in an extremely large number of instances, as there are $1 \cdot 9 + 1 \cdot (n - 10) + 9 \cdot (n - 10)$ different pairs, where n denotes the number of tweets under a given hashtag. Additionally, to help the model learn the symmetric predictions, we include these pairs' symmetric counterparts as well, which doubles the total number.

Due to resource constraints, we decided to start off with a variety of readily-available models and rule out those that perform badly in our rough preliminary evaluation. Specifically, we trained a selection of models with their default hyperparameters on the train set and evaluated them on the trial set. Surprisingly, a single model, gradient boosting (GB) with variance loss, performed the best, so we decided to use it in as our base model. We used a GB implementation of the scikit-learn package (Pedregosa et al., 2011). We ran a fine-grained 5-fold cross-validation over two GB hyperparameters: number of estimators and maximum tree depth. As we are working with tweets grouped into hashtags, the folds actually contained whole hashtags. Additionally, note that we used a random sample of 80% of pairs for training in order to reduce the computation costs.

4 Evaluation

The subtask 6A was evaluated in terms of accuracy (higher is better), whereas the subtask 6B was evaluated in terms of a metric inspired by edit distance. The metric captures how many moves the tweet must make to fall into a correct bin (lower is better). This metric was normalized by the maximum possible edit distance. Both metrics were micro-averaged across hashtags.

4.1 Feature Analysis

A gradient boosting model allowed us to effortlessly acquire the list of feature importances. We report the top ten most relevant features, according to the model, in Figure 1. Most notably, five cultural reference features found their spot within this list. This confirmed our earlier intuition that, at least within the @midnight game show, most jokes are based on culture references, which are slightly transformed to induce a comical feel.

[6] https://code.google.com/archive/p/word2vec/

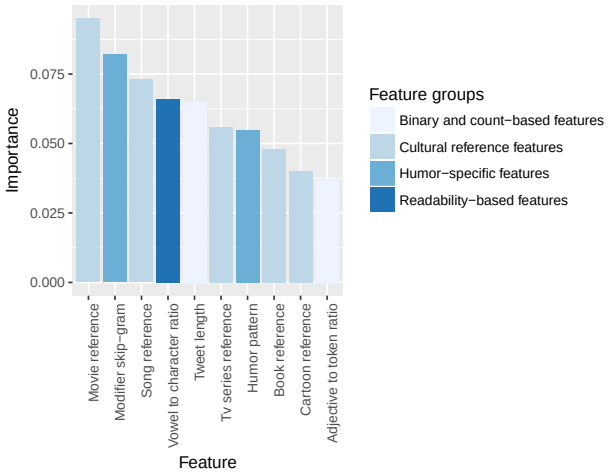

Figure 1: Top ten most important features, according to the trained GB model.

4.2 Model Variants

The two models we submitted for the subtask 6A are effectively identical. The only thing that makes them different is the size of the hyperparameter search space used model optimization: the second model used a more fine-grained grid of values and thus expectedly performed better. Our best (unofficial) model, denoted *TakeLab-2*, ranked 2nd (officially 7th) among 10 submissions with the accuracy of 0.641. Other participants' scores can be found in Table 2. We also included our official submissions (*TakeLab-official-1* and *TakeLab-official-2* along their non-official counterparts.[7]

As mentioned earlier, to obtain the tripartite ranking for the subtask 6B, we used the pairwise classifications obtained by the model used in the subtask 6A. This brought us to the distance metric value of 0.908, placing us at the 2nd place out of 9 submissions. Additionally, we experimented with LambdaMART algorithm to see how a full-fledged learning-to-rank algorithm would perform at this subtask. To that end, we explored two different variants of the model: one using all the described features (denoted *LambdaMART-all*) and one only the top ten features according to the model we used in subtask 6A (denoted *LambdaMART-10*). In comparison to the models we submitted, it is intriguing to see that both of these models perform only slightly worse. What is more, model variant trained using only the top ten features would rank third among all submissions.

[7]Unfortunately, we accidentally swapped the labels in the submission file so we had to unofficially submit a fixed file.

Team name	Accuracy
HumorHawk-2	0.675
TakeLab-2	**0.641**
HumorHawk-1	0.637
DataStories-1	0.632
Duluth-2	0.627
TakeLab-official-1	**0.597**
SRHR	0.523
SVNIT@SemEval	0.506
TakeLab-1	**0.403**
Duluth-1	0.397
TakeLab-official-2	**0.359**
QUB	0.187

Table 2: Final rankings on the subtask 6A. Our submissions are bolded.

Team name	Distance
Duluth-2	0.872
TakeLab-1	**0.908**
LambdaMART-10	**0.912**
QUB-1	0.924
QUB-2	0.924
SVNIT@SemEval-2	0.938
TakeLab-2	**0.944**
LambdaMART-all	**0.946**
SVNIT@SemEval-1	0.949
Duluth-1	0.967
# WarTeam-1	1.000

Table 3: Final rankings on the subtask 6B. Our submissions are bolded.

5 Conclusion

We described the system for humor detection which we participated with in the SemEval-2017 Task 6 (subtasks A and B). The gist of our system lies in an off-the-shelf gradient boosting model built on a rich set of handcrafted features. Knowing that humor understanding requires a broader context that also asks for external knowledge, we manually compiled a series of features that can capture the cultural references in a tweet – celebrities, movies, TV series, books, and so on. Besides that, we also included pronounciation-based, as well as humor-specific features that can recognize one-line jokes and puns, hoping to capture the humor patterns used throughout the @midnight game show. Future work includes experiments with full-fledged learning-to-rank models and a more detailed investigation of linguistically-motivated humor features, both backed up by exhaustive cross-dataset analyses.

References

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit*. O'Reilly Media, Inc.

Lei Chen and Chong Min Lee. 2017. Convolutional neural network for humor recognition. *arXiv preprint arXiv:1702.02584* .

Rudolph Flesch. 1948. A new readability yardstick. *Journal of applied psychology* 32(3):221.

Chloe Kiddon and Yuriy Brun. 2011. That's what she said: double entendre identification. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL HLT 2011)*. Portland, Oregon, USA, pages 89–94.

David Lozić, Doria Šarić, Ivan Tokić, Zoran Medić, and Jan Šnajder. 2017. TakeLab at SemEval-2017 Task 4: Recent deaths and the power of nostalgia in sentiment analysis in Twitter. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*. Vancouver, Canada, pages 782–787.

Rada Mihalcea and Carlo Strapparava. 2005. Making computers laugh: Investigations in automatic humor recognition. In *Proceedings of the 2005 Conference on Human Language Technology and Empirical Methods in Natural Language Processing (HIT EMNLP 2005)*. Vancouver, B.C., Canada, pages 531–538.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of the Neural Information Processing Systems Conference (NIPS 2013)*. Lake Tahoe, USA, pages 3111–3119.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research* 12:2825–2830.

Sasa Petrović and David Matthews. 2013. Unsupervised joke generation from big data. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL 2013)*. Sofia, Bulgaria, pages 228–232.

Peter Potash, Alexey Romanov, and Anna Rumshisky. 2017. SemEval-2017 Task 6: #HashtagWars: Learning a sense of humor. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*. Vancouver, Canada, pages 49–57.

Alessandro Valitutti, Hannu Toivonen, Antoine Doucet, and Jukka M. Toivanen. 2013. "Let everything turn well in your wife": Generation of adult humor using lexical constraints. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL 2013)*. Sofia, Bulgaria, pages 243–248.

Diyi Yang, Alon Lavie, Chris Dyer, and Eduard H. Hovy. 2015. Humor recognition and humor anchor extraction. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP 2015)*. Lisbon, Portugal, pages 2367–2376.

Amy Zhao Yu, Shahar Ronen, Kevin Hu, Tiffany Lu, and César A Hidalgo. 2016. Pantheon 1.0, a manually verified dataset of globally famous biographies. *Scientific data* 3.

Renxian Zhang and Naishi Liu. 2014. Recognizing humor on Twitter. In *Proceedings of the 23rd ACM International Conference on Information and Knowledge Management (CIKM 2014)*. ACM, Shanghai, China, pages 889–898.