SemEval-2017 Task 6: #HashtagWars: Learning a Sense of Humor

Peter Potash, Alexey Romanov, Anna Rumshisky
University of Massachusetts Lowell
Department of Computer Science
{ppotash,aromanov,arum}@cs.uml.edu

Abstract

This paper describes a new shared task for humor understanding that attempts to eschew the ubiquitous binary approach to humor detection and focus on comparative humor ranking instead. The task is based on a new dataset of funny tweets posted in response to shared hashtags, collected from the 'Hashtag Wars' segment of the TV show @midnight. The results are evaluated in two subtasks that require the participants to generate either the correct pairwise comparisons of tweets (subtask A), or the correct ranking of the tweets (subtask B) in terms of how funny they are. 7 teams participated in subtask A, and 5 teams participated in subtask B. The best accuracy in subtask A was 0.675. The best (lowest) rank edit distance for subtask B was 0.872.

1 Introduction

Most work on humor detection approaches the problem as binary classification: humor or not humor. While this is a reasonable initial step, in practice humor is continuous, so we believe it is interesting to evaluate different degrees of humor, particularly as it relates to a given person's sense of humor. To further such research, we propose a dataset based on humorous responses submitted to a Comedy Central TV show, allowing for computational approaches to comparative humor ranking.

Debuting in Fall 2013, the Comedy Central show @midnight[1] is a late-night "game-show" that presents a modern outlook on current events by focusing on content from social media. The show's contestants (generally professional comedians or actors) are awarded points based on how funny their answers are. The segment of the show that best illustrates this attitude is the Hashtag Wars (HW). Every episode the show's host proposes a topic in the form of a hashtag, and the show's contestants must provide tweets that would have this hashtag. Viewers are encouraged to tweet their own responses. From the viewers' tweets, we are able to apply labels that determine how relatively humorous the show finds a given tweet.

Because of the contest's format, it provides an adequate method for addressing the selection bias (Heckman, 1979) often present in machine learning techniques (Zadrozny, 2004). Since each tweet is intended for the same hashtag, each tweet is effectively drawn from the same sample distribution. Consequently, tweets are seen not as humor/non-humor, but rather varying degrees of wit and cleverness. Moreover, given the subjective nature of humor, labels in the dataset are only "gold" with respect to the show's sense of humor. This concept becomes more grounded when considering the use of supervised systems for the dataset.

The idea of the dataset is to learn to characterize the sense of humor represented in this show. Given a set of hashtags, the goal is to predict which tweets the show will find funnier within each hashtag. The degree of humor in a given tweet is determined by the labels provided by the show. We propose two subtasks to evaluate systems on the dataset. The first subtask is pairwise comparison: given two tweets, select the funnier tweet, and the pairs will be derived from the labels assigned by the show to individual tweets. The second subtask is to rank the the tweets based on the comparative labels provided by the show. This is a semi-ranking task because most labels are applied to more than one tweet. Seen as a classification task, the labels are comparative, because there is a notion of distance. We introduce a new edit distance-

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

Proceedings of the 11th International Workshop on Semantic Evaluations (SemEval-2017), pages 49–57,
Vancouver, Canada, August 3 - 4, 2017.

inspired metric for this subtask.

A number of different computational approaches to humor have been proposed within the last decade (Yang et al., 2015; Mihalcea and Strapparava, 2005; Zhang and Liu, 2014; Radev et al., 2015; Raz, 2012; Reyes et al., 2013; Barbieri and Saggion, 2014; Shahaf et al., 2015; Purandare and Litman, 2006; Kiddon and Brun, 2011). In particular, Zhang and Liu (2014); Raz (2012); Reyes et al. (2013); Barbieri and Saggion (2014) focus on recognizing humor in Twitter. However, the majority of this work focuses on distinguishing humor from non-humor.

This representation has two shortcomings: (1) it ignores the continuous nature of humor, and (2) it does not take into account the subjectivity in humor perception. Regarding the first issue, we believe that shifting away from the binary approach to humor detection as done in the present task is a good pathway towards advancing this work. Regarding the second issue, consider a humour annotation task done by Shahaf et al. (2015), in which the annotators looked at pairs of captions from the New Yorker Caption Content[2], Shahaf et al. (2015) report that "Only 35% of the unique pairs that were ranked by at least five people achieved 80% agreement..." In contrast, the goal of the present task is to not to identify humour that is universal, but rather, to capture the specific sense of humour represented in the show.

2 Related Work

Mihalcea and Strapparava (2005) developed a humor dataset of puns and humorous one-liners intended for supervised learning. In order to generate negative examples for their experimental design, the authors used news titles from Reuters and the British National Corpus, as well as proverbs. Recently, Yang et al. (2015) used the same dataset for experimental purposes, taking text from AP News, New York Times, Yahoo! Answers, and proverbs as their negative examples. To further reduce the bias of their negative examples, the authors selected negative examples with a vocabulary that is in the dictionary created from the positive examples. Also, the authors forced the negative examples to have a similar text length compared to the positive examples.

Zhang and Liu (2014) constructed a dataset for recognizing humor in Twitter in two parts. First, the authors use the Twitter API with targeted user mentions and hashtags to produce a set of 1,500 humorous tweets. After manual inspections, 1,267 of the original 1,500 tweets were found to be humorous, of which 1,000 were randomly sampled as positive examples in the final dataset. Second, the authors collect negative examples by extracting 1,500 tweets from the Twitter Streaming API, manually checking for the presence of humor. Next, the authors combine these tweets with tweets from part one that were found to actually not contain humor. The authors argue this last step will partly assuage the selection bias of the negative examples.

In Reyes et al. (2013) the authors create a model to detect ironic tweets. To construct their dataset they collect tweets with the following hashtags: irony, humor, politics, and education. Therefore, a tweet is considered ironic solely because of the presence of the appropriate hashtag. Barbieri and Saggion (2014) also use this dataset for their work.

Finally, recently researchers have developed a dataset similar to our HW dataset based on the New Yorker Caption Contest (NYCC) (Radev et al., 2015; Shahaf et al., 2015). Whereas for the HW segment, viewers submit a tweet in response to a hashtag, for the NYCC readers submit humorous captions in response to a cartoon. It is important to note this key distinction between the two datasets, because we believe that the presence of the hashtag allows for further innovative NLP methodologies aside from solely analyzing the tweets themselves. In Radev et al. (2015), the authors developed more than 15 unsupervised methods for ranking submissions for the NYCC. The methods can be categorized into broader categories such as originality and content-based.

Alternatively, Shahaf et al.(2015) approach the NYCC dataset with supervised models, evaluating on a pairwise comparison task, upon which we base our evaluation methodology. The features to represent a given caption fall in the general areas of Unusual Language, Sentiment, and Taking Expert Advice. For a single data point (which represents two captions), the authors concatenate the features of each individual caption, as well as encoding the difference between each caption's vector. The authors' best-performing system records a 69% accuracy on the pairwise evaluation task. Note that for this evaluation task, random baseline is 50%.

[2] http://contest.newyorker.com/

3 #HashtagWars Dataset

3.1 Data collection

The following section describes our data collection process. First, when a new episode airs (which generally happens four nights a week), a new hashtag will be given. We wait until the following morning to use the public Twitter search API[3] to collect tweets that have been posted with the new hashtag. Generally, this returns 100-200 tweets. We wait until the following day to allow for as many tweets as possible to be submitted. The day of the ensuing episode (i.e. on a Monday for a hashtag that came out for a Thursday episode), @midnight creates a Tumblr post[4] that announces the top-10 tweets from the previous episode's hashtag (the tweets are listed as embedded images, as is often done for sharing public tweets on websites). If they're not already present, we add the tweets from the top-10 to our existing list of tweets for the hashtag. We also perform automated filtering to remove redundant tweets. Specifically, we see that the text of tweets (aside from hashtags and user mentions) are not the same. The need for this results from the fact that some viewers submit identical tweets.

Using both the @midnight official Tumblr account, as well as the show's official website where the winning tweet is posted, we annotate each tweet with labels `0`, `1` and `2`. Label `2` designates the winning tweet. Thus, the label `2` only occurs once for each hashtag. Label `1` indicates that the tweet was selected as a top-10 tweet (but *not* the winning tweet) and label `0` is assigned for all other tweets. It is important to note that every time we collect a tweet, we must also collect its tweet ID. While this was initially done to comply with Twitter's terms of use[5], which disallows the public distribution of users' tweets, The presence of tweet IDs allows us to easily handle the evaluation process when referencing tweets (see Section 4). The need to determine the tweet IDs for tweets that weren't found in the initial query (i.e. tweets added from the top 10) makes the data collection process slightly laborious, since the top-10 list doesn't contain the tweet ID. In fact, it doesn't even contain the text itself since it's actually an image.

3.1.1 A Semi-Automated System for Data Collection

Because the data collection process is continuously repeated and requires a non-trivial amount of human labor, we have built a helper system that can partially automate the process of data collection. This system is organized as a website with a convenient user interface.

On the start page the user enters the id of the Tumblr post with the tweets in the top 10. Next, we invoke Tesseract [6], an OCR command-line utility, to recognize the textual content of the tweet images. Using the recognized content, the system forms a webpage on which the user can simultaneously see the text of the tweets as well as the original images. On this page, the user can query the Twitter API to search by text, or click the button "Open twitter search" to open the Twitter Search page if the API returns zero results. We note that the process is not fully automated because a given text query can we return redundant results, and we primarily check to make sure we add the tweet that came from the appropriate user. With the help of this system, the process of collecting the top-10 tweets (along with their tweet IDs) takes roughly 2 minutes. Lastly, we note that the process for annotating the winning tweet (which is already included in the top-10 posted in the Tumblr list) is currently manual, because it requires going to the @midnight website. This is another aspect of the data collection system that could potentially be automated.

3.2 Dataset

Data collection occurred for roughly eight months, producing a total of 12,734 tweets for 112 hashtags. The resulting dataset is what we used for the task.

The distribution of the number of tweets per hashtag is represented in Figure 1. For 71% of hashtags, we have at least 90 tweets. The files of the individual hashtags are formatted so that the individual hashtag tokens are easily recoverable. Specifically, tokens are separated by the '_' character. For example, the hashtag *FastFoodBooks* has the file name "fast_food_books.tsv".

Figure 2 represents an example of the tweets collected for the hashtag *FastFoodBooks*. Ob-

[3] https://dev.twitter.com/rest/public/search

[4] http://atmidnightcc.tumblr.com/

[5] https://dev.twitter.com/overview/terms

[6] https://github.com/tesseract-ocr/tesseract

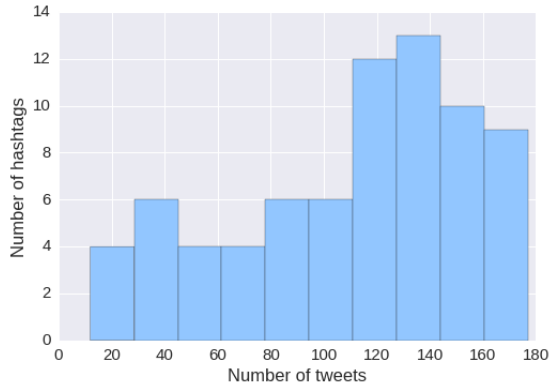

Figure 1: Distribution of the numbers of tweets per hashtag

serve that this hashtag requires external knowledge about fast food and books in order to understand the humor. Furthermore, this hashtag illustrates how prevalent puns are in the dataset, especially related to certain target hashtags. In contrast, the hashtag *IfIWerePresident* (see Figure 3) does not require external knowledge and the tweets are understandable without awareness of any specific concepts.

For the purpose of our task, we released 5 files/660 tweets as the trial data, 101 files/11,325 tweets (separate from the trial data) as the training data, and 6 files/749 tweets as the evaluation data. The 6 evaluation files were chosen based on the following logic: first, we examined the results of our own systems on individual hashtags using leave-one-out evaluation (Potash et al., 2016). We looked for a mixture of hashtags that had high, average, and low performance. Secondly, we wanted a mixture of hashtags that promote different types of humor, such as puns that use external knowledge (for example the hashtag *FastFoodBooks* in Figure 3.2), or hashtags that seek to express more general humor (for example the hashtag *IfIWerePresident* in Figure 3.2).

4 Subtasks

In this task, the results are evaluated in two subtasks. Subtask A requires the participants to generate the correct pairwise comparisons of tweets to determine which tweet is funnier according to the TV show @midnight. Subtask B asks for the correct ranking of tweets in terms of how funny they are (again, according to @midnight).

As I Lay Dying of congestive heart failure @midnight #FastFoodBooks
Harry Potter and the Order of the Big Mac #FastFoodBooks @midnight
The Girl With The Jared Tattoo #FastFoodBooks @midnight
A Room With a Drive-thru @midnight #FastFoodBooks

Figure 2: An example of the items in the dataset for the hashtag *FastFoodBooks* that requires external knowledge in order to understand the humor. Furthermore, the tweets for this hashtag are puns connecting book titles and fast food-related language

#IfIWerePresident my Cabinet would just be cats. @midnight
Historically, I'd oversleep and eventually get fired. @midnight #IfIWerePresident
#IfIWerePresident I'd pardon Dad so we could be together again... @midnight
#IfIWerePresident my estranged children would finally know where I was @midnight

Figure 3: An example of the items in the dataset for the hashtag *IfIWerePresident* that does not require external knowledge in order to understand the humor

4.1 Subtask A: Pairwise Comparison

For the first subtask, we follow the approach taken by Shahaf et al. (2015) and make predictions on pairs of tweets with the goal of determining which tweet is funnier. Using the tweets for each hashtag, we construct pairs of tweets in which one tweet is judged by the show to be funnier than the other. The pairs used for evaluation are constructed as follows:

(1) The tweets that are the top-10 funniest tweets are paired with the tweets not in the top-10.

(2) The winning tweet is paired with the other tweets in the top-10.

If we have n tweets for a given hashtag, (1) will produce $10(n-10)$ pairs, and (2) will produce 9 pairs, giving us $10n-91$ data points for a single

hashtag. Constructing the pairs for evaluation in this way ensures that one of the tweets in each pair has been judged to be funnier than the other. We follow Shahaf et al. and use the label 1 to denote that the first tweet is funnier, and 0 to denote that the second tweet is funnier. However, this labeling is counter-intuitive to zero-indexing, and could be changed to avoid confusion in labeling (see Section 5).

Since we only provide teams with files containing tweet ID, tweet text, and tweet label (gold label: `0`, `1`, or `2`), it is up to the teams to form the appropriate pairs with the correct labels. In order to produce balanced training data, we recommend that the ordering of tweets in a pair be determined by a coin-flip. At evaluation time, we provide the teams with hashtag files with tweet id and tweet text. We then ask the teams to provide predictions for every possible tweet combination. Our evaluation script then chooses only the tweet pairs where two different labels are present. The pairs can be listed in either ordering of the tweets because the scorer accounts for the two possible orderings for each pair. We decided against the idea of providing the appropriate pairs themselves for evaluation because it is very easy to use frequencies of tweet IDs in the pairs to determine overall tweet label.

The evaluation measure for subtask A is the micro average of accuracy across the individual evaluation hashtags. For a given hashtag, the accuracy is the number of correctly predicted pairs divided by the total number of pairs. Therefore, random guessing will produce 50% accuracy on this task.

4.2 Subtask B: Ranking

The second subtask asks teams to use the same input data for training and evaluation as subtask A. However, whereas subtask A creates pairs of tweets based on the labeling, subtask B asks teams to predict the labels directly. For this dataset, the number of tweets per class is known. Moreover, since the labels describe a partial ordering, predicting the labels is akin to providing a ranking of tweets in order of how funny they are. Therefore, for subtask B, we ask the teams to provide prediction files where the tweets are ranking by how funny they are. From the provided ranking we infer the labeling: the first tweet is labeled `2`, the next nine labeled `1`, and the rest labeled `0`.

The metric for evaluating subtask B is inspired by a notion of edit distance, because standard classification metrics do not take into account class' comparative rankings. Treating labels as buckets, the metric determines, for a predicted label, how many 'moves' are needed to place it in the correct bucket. For example, if the correct label is `1` and the predicted label is `0`, the edit distance is 1. Similarly, if the correct label is `0` and the predicted label is `2`, the edit distance is 2. For a given hashtag file, the maximum edit distance for all tweets is 22. As a result, the edit distance for a given hashtag file is the total number of moves for all tweets divided by 22. This gives a normalized metric between 0 and 1 where a lower value is better. For the final distance metric, we micro-average across all evaluation files.

5 Results

Three teams participated only in subtask A, one team participated only in subtask B, and four teams participated in both subtasks. The official results for participating teams are shown in Tables 1 and 2 for subtasks A and B, respectively. Note that due to space constraints we use short versions of hashtag names in the tables. Namely, "Christmas" corresponds to the hashtag *RuinAChristmasMovie*, "Shakespeare" corresponds to *ModernShakespeare*, "Bad Job" to *BadJobIn5Words*, "Break Up" to *BreakUpIn5Words*, "Broadway" to *BroadwayACeleb*, and "Cereal" to *CerealSongs*.

We report the results broken down by hashtag, as well as the overall micro-average. This table records results that were submitted to the CodaLab competition pages[7]. TakeLab (Kukovačec et al., 2017) submitted predictions with the labels flipped, which causes each run to appear in the table twice. The corrected files are not given an official ranking. After the release of the labeled evaluation data, many teams reported improved results. We have accrued these new results and combined them with the official submission rankings to produce Tables 3 and 4. The goal of these tables is to report the most up-to-date results on the evaluation set. Moreover, all results that do not have an official ranking in these tables are results that are reported individually by the teams in their system papers (except for TakeLab's results) after the gold evaluation labels were released.

[7] https://competitions.codalab.org/competitions/15682, https://competitions.codalab.org/competitions/15689

Rank	Team	Run	Hashatag						Average
			Christmas	Shakespeare	Bad Job	Break Up	Broadway	Cereal	
1	HumorHawk	2	0.673	**0.789**	0.704	0.723	0.643	0.492	**0.675** (±0.101)
	TakeLab	2	**0.683**	0.543	0.641	0.576	**0.716**	**0.704**	0.641 (±0.071)
2	HumorHawk	1	0.650	0.726	0.603	0.620	0.627	0.588	0.637 (±0.049)
3	DataStories	1	0.641	0.714	**0.828**	0.686	0.496	0.479	0.632 (±0.134)
4	Duluth	2	0.485	0.585	0.557	**0.913**	0.527	0.589	0.627 (±0.154)
	TakeLab	1	0.575	0.550	0.620	0.563	0.603	0.689	0.597 (±0.051)
5	SRHR	1	0.520	0.451	0.606	0.505	0.550	0.524	0.523 (±0.051)
6	SVNIT	1	0.455	0.353	0.395	0.654	0.542	0.563	0.506 (±0.113)
7	TakeLab	1	0.425	0.450	0.380	0.437	0.397	0.311	0.403 (±0.051)
8	Duluth	1	0.441	0.445	0.417	0.240	0.470	0.402	0.397 (±0.083)
9	TakeLab	2	0.317	0.457	0.359	0.424	0.284	0.296	0.359 (±0.071)
10	QUB	1	0.165	0.343	0.229	0.165	0.091	0.154	0.187 (±0.086)
Average			0.529 (±0.157)	0.550 (±0.156)	0.560 (±0.171)	0.565 (±0.221)	0.527 (±0.170)	0.518 (±0.158)	0.542 (±0.150)

Table 1: The official results for the subtask A broken down by hashtag. Bold indicates the best run for the given hashtag. "Christmas" corresponds to the hashtag *RuinAChristmasMovie*, "Shakespeare" corresponds to *ModernShakespeare*, "Bad Job" to *BadJobIn5Words*, "Break Up" to *BreakUpIn5Words*, "Broadway" to *BroadwayACeleb*, and "Cereal" to *CerealSongs*.

Rank	Team	Run	Hashatag						Average
			Christmas	Shakespeare	Bad Job	Break Up	Broadway	Cereal	
1	Duluth	2	**0.818**	0.909	1.000	**0.636**	1.000	0.909	**0.872** (±0.137)
2	TakeLab	1	0.909	0.909	1.000	0.818	1.000	**0.818**	0.908 (±0.081)
3	QUB	1	**0.818**	0.909	**0.818**	1.000	1.000	0.909	0.924 (±0.081)
3	QUB	2	**0.818**	0.909	**0.818**	1.000	1.000	0.909	0.924 (±0.081)
5	SVNIT	2	**0.818**	1.000	0.909	1.000	1.000	**0.818**	0.938 (±0.089)
6	TakeLab	2	**0.818**	1.000	1.000	0.909	1.000	0.909	0.944 (±0.074)
7	SVNIT	1	1.000	**0.818**	1.000	0.909	1.000	1.000	0.949 (±0.076)
8	Duluth	1	1.000	1.000	1.000	1.000	**0.909**	0.909	0.967 (±0.047)
9	#WarTeam	1	1.000	1.000	1.000	1.000	1.000	1.000	1.000 (±0.000)
Average			0.889 (±0.088)	0.939 (±0.064)	0.949 (±0.08)	0.919 (±0.124)	0.990 (±0.030)	0.909 (±0.064)	0.936 (±0.036)

Table 2: The official results for the subtask B broken down by hashtag. Bold indicates the best run for the given hashtag. "Christmas" corresponds to the hashtag *RuinAChristmasMovie*, "Shakespeare" corresponds to *ModernShakespeare*, "Bad Job" to *BadJobIn5Words*, "Break Up" to *BreakUpIn5Words*, "Broadway" to *BroadwayACeleb*, and "Cereal" to *CerealSongs*.

6 Discussion

6.1 Task Analysis

The last row of Table 1 shows the average accuracy of each hashtag across all systems (the official results of the TakeLab systems are not included in this average since we also include in the average the unofficial, corrected results). The two easiest hashtags are ones that require less external knowledge compared to the other four. These four hashtags specifically riff on a particular Christmas movie, Shakespeare quote, celebrity/Broadway play, or cereal/song. Consequently, one single system did best in three out of four of these hashtags (TakeLab). It is not coincidence, since this system made extensive use of external knowledge bases. Furthermore, the three hashtags where it did best required knowledge of specific entities, whereas the knowledge required in the hashtag *ModernShakespeare* is the actual lines from Shakespeare plays.

As we mentioned in Section 3.2, the evaluation hashtags were chosen partly because of our own system performance on the hashtags (Potash et al., 2016). One of the most difficult hashtags from our initial experiments was the hashtag *CerealSongs*, which was the hashtag systems performed the worse on in this task. We believe this is because the humor in this hashtag is based on *two* sources of external knowledge: cereals and songs. Correspondingly, the hashtag with the second worse performance also requires two sources of external knowledge: Broadway plays and celebrities (this hashtag was originally chosen as a representative of the hashtags our systems recorded average performance). The hashtag *BadJobIn5Words* was one that had high performance by our own systems, and that continued in this task. This hashtag had the second highest accuracy, and would have had the highest if the Duluth team (Yan and Pedersen, 2017) did not have such remarkable success on the highest accuracy hashtag, *BreakUpIn5Words*.

The poor performance for the hashtags *CerealSongs* and *BroadwayACeleb* is also interesting

Official Ranking	Team	Accuracy	Notes
	SVNIT	0.751	An SVM classifier with incongruity, ambiguity, and stylistic features
	DataStories	0.711	Siamese biderectional LSTM with attention
	HumorHawk	0.683	Embedding/Character Joint Humor Model
1	HumorHawk	0.675	XGBoost ensemble of feature-based and emedding models
	TakeLab	0.641	Gradient boosting classifier with a rich set of features, including cultural references
2	HumorHawk	0.637	Embedding/Character Joint Humor Model
3	DataStories	0.632	Siamese biderectional LSTM with attention
4	Duluth	0.627	Trigram language model (news dataset)
	SRHR	0.564	Random Forest classifier with word association and sematic relatedness features
5	SRHR	0.523	Random Forest classifier with word association and sematic relatedness features
6	SVNIT	0.506	Multilayer perceptron with incongruity, ambiguity, and stylistic features
7	TakeLab	0.403	Gradient boosting classifier with a rich set of features, including cultural references (reversed labels)
8	Duluth	0.397	Trigram language model (tweets dataset)
9	TakeLab	0.359	Gradient boosting classifier with a rich set of features, including cultural references (reversed labels)
10	QUB	0.187	A set of imblanaced classifiers with n-gram features

Table 3: Unofficial results for the subtask A on the released evaluation set reported by the participating teams

Official Ranking	Team	Score	Notes
	Duluth	0.853	Bigram language model (news dataset)
1	Duluth	0.872	Trigram language model (news dataset)
2	TakeLab	0.908	Gradient boosting classifier with a rich set of features, including cultural references
3	QUB	0.924	A set of imblanaced classifiers with n-gram features
3	QUB	0.924	A set of imblanaced classifiers with n-gram features
5	SVNIT	0.938	Multilayer perceptron with incongruity, ambiguity, and stylistic features
6	TakeLab	0.944	Gradient boosting classifier with a rich set of features, including cultural references
7	SVNIT	0.949	A Naive Bayes classifier with incongruity, ambiguity, and stylistic features
8	Duluth	0.967	Trigram language model (tweets dataset)
9	#WarTeam	1.000	A word-based voting algorithm of a Naive Bayes and neural network word scorers

Table 4: Unofficial results for the subtask B on the released evaluation set reported by the participating teams

to note since they were chosen because the hashtag names had strong similarity to hashtags in the training data. For example, 12 hashtags in the training data had the word 'Song'. Likewise, five hashtags had the word 'Celeb', and there was one more hashtag with the word 'Broadway'. Alternatively, The two hashtags with the best performance followed the 'X in X words' format, for which there were 16 such hashtags in the training data. Regarding the hashtag *BadJobIn5Words*, there are six hashtags in the training data beginning with the word 'Bad'.

Our current task analysis has focused on subtask A. The primary reason for this is that the performance on subtask B was relatively poor. To put the results in perspective, we created random guesses for subtask B, and these random guesses recorded an average distance of 0.880. From the results, only one team was able to beat this score. We can see that two of the three highest performing teams in subtask A did not participate in subtask B, and the other team that did participate approached subtask B as a secondary task (see Section 6.2).

6.2 System Analysis

For the teams that participated in both subtasks, they used the output of a single system to predict for both subtasks. Two teams, SVNIT (Mahajan and Zaveri, 2017) and QUB (Han and Toner, 2017) , initially predicted the labels of each tweet based on the output of a supervised classifier, and then used these labels to both rank the tweets and make pairwise predictions for the subtasks. Duluth took a similar approach, but used the output of a language model to rank the tweets, as opposed to labels provided by a classifier. Conversely, TakeLab sought to solve subtask A first, then used the frequencies of a tweet being chosen as funnier in a pair to provide a single, ordered metric to make predictions for subtask B. The team that only participated in subtask B, #WarTeam, also used the output of a supervised classifier to label the tweets, which in turn provided the ranking. One of interesting results from having the two subtasks (which are effectively two different ways of evaluating the same overall task) is to see how it distinguishes the unified approaches to solving both subtasks. We can see that, in fact, the top team is not con-

sistent between the two subtasks. It is not a surprise to see that the best performing team (out of the four that participated in both subtasks) in subtask A was TakeLab, who focused primarily on this task. Conversely, TakeLab finished second in subtask B to Duluth, who focused on creating an ordered metric for ranking via language models.

In terms of overall system approach, we can analyze how heavily systems rely on feature-engineering, verse using learned representations from neural networks. Three of the top four systems for subtask A leveraged neural network architectures. Two of these systems used only pretrained word representations as external knowledge for the neural network systems. This is in opposition to other systems that relied on the output of separate tools, or looking up terms in corpora. Some teams, such as HumorHawk[8] (Donahue et al., 2017) and #WarTeam, used a combination of these two types of systems, and notably, the system that was ranked first in Subtask A (HumorHawk) was an ensemble system that utilized prediction from both feature-based and neural networks-based models.

As for the feature-based systems, one trend we observed is that many teams tried to capture the incongruity aspect of humor (Cattle and Ma, 2017) , often present in the dataset. The approaches used by teams varied from n-gram language models, word association, to semantic relatedness features. In addition, the TakeLab team used cultural reference features, such as movie and song references, and Google Trends features for named entities. During the performed analysis, the team found these features most useful for the model.

Considering neural network-based systems, LSTMs were used the most, which is expected given the sequential nature of text data. Plain LSTM models alone, using pretrained word embeddings, achieved competitive results, and DataStories (Baziotis et al., 2017) ranked third using a siamese bidirectional LSTM model with attention.

One key difference between the dataset used in this task and the datasets based on the NYCC (Radev et al., 2015; Shahaf et al., 2015) is the presence of the hashtag. Some teams used additional hashtag-based features in their systems. For example, humor patterns, defined by the hashtag, were one of the most important features for the TakeLab team. Other teams used semantic distances between the hashtag and tweets as features.

Table 1 also includes the standard deviation of system scores across the hashtags. Looking at the numbers there appears to be little in the way of a pattern regarding the standard deviation numbers. When correlated with system accuracy, the results is 0.11, which supports the idea that consistency across the hashtags has no relation to overall system performance. Even between the two purest neural network-based systems, DataStories and HumorHawk run 1, the standard deviations vary greatly: 0.134 (DataStories) and 0.049 (HumorHawk run 1). In fact, 0.049 was the lowest standard deviation across all systems. Duluth recorded the highest standard deviation across the datasets, primarily due to the fact that it had the single highest accuracy on any hashtag (0.913 for the hashtag *BreakUpIn5Words*), as well as the lowest single hashtag score for any system with an overall accuracy greater than 0.600 (0.485 for the hashtag *RuinAChristmasMovie*). One possibility for this high standard deviation is that this is the only unsupervised system. However, the other run submitted by Duluth (whose primary difference is that its language model was trained on a dataset of tweets as opposed to news articles) has a both a significantly lower accuracy and standard deviation.

7 Conclusion

We have presented the results of the SemEval 2017 shared task: #HashtagWars: Learning a Sense of Humor. It was the first year this task was presented, attracting 8 teams and 19 systems across two substasks. The top performing systems achieved 0.675 accuracy in subtask A and 0.872 score on subtask B, advancing the difficult task of humor understanding. Interestingly, the top-ranked system used an ensemble of both feature-based and neural network-based systems, suggesting that despite the overwhelming success of neural networks in the past few years, human intuition is still important for systems that seek to automatically understand humor.

[8]Two of the organizers were members of this team. They were not involved in the data selection process. They had no knowledge of which files were selected for evaluation, nor how these files were chosen.

References

Francesco Barbieri and Horacio Saggion. 2014. Automatic detection of irony and humour in twitter.

In *Proceedings of the International Conference on Computational Creativity*.

Christos Baziotis, Nikos Pelekis, and Christos Doulkeridis. 2017. Datastories at semeval-2017 task 6: Siamese lstm with attention for humorous text comparison. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*. Association for Computational Linguistics, Vancouver, Canada, pages 389–394. http://www.aclweb.org/anthology/S17-2065.

Andrew Cattle and Xiaojuan Ma. 2017. Srhr at semeval-2017 task 6: Word associations for humour recognition. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*. Association for Computational Linguistics, Vancouver, Canada, pages 400–405. http://www.aclweb.org/anthology/S17-2067.

David Donahue, Alexey Romanov, and Anna Rumshisky. 2017. Humorhawk at semeval-2017 task 6: Mixing meaning and sound for humor recognition. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*. Association for Computational Linguistics, Vancouver, Canada, pages 98–102. http://www.aclweb.org/anthology/S17-2010.

Xiwu Han and Gregory Toner. 2017. Qub at semeval-2017 task 6: Cascaded imbalanced classification for humor analysis in twitter. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*. Association for Computational Linguistics, Vancouver, Canada, pages 379–383. http://www.aclweb.org/anthology/S17-2063.

James J Heckman. 1979. Sample selection bias as a specification error. *Econometrica: Journal of the econometric society* pages 153–161.

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: short papers-Volume 2*. Association for Computational Linguistics, pages 89–94.

Marin Kukovačec, Juraj Malenica, Ivan Mršić, Antonio Šajatović, Domagoj Alagić, and Jan Šnajder. 2017. Takelab at semeval-2017 task 6: #rankinghumorin4pages. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*. Association for Computational Linguistics, Vancouver, Canada, pages 395–399. http://www.aclweb.org/anthology/S17-2066.

Rutal Mahajan and Mukesh Zaveri. 2017. Svnit @ semeval 2017 task-6: Learning a sense of humor using supervised approach. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*. Association for Computational Linguistics, Vancouver, Canada, pages 410–414. http://www.aclweb.org/anthology/S17-2069.

Rada Mihalcea and Carlo Strapparava. 2005. Making computers laugh: Investigations in automatic humor recognition. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pages 531–538.

Peter Potash, Alexey Romanov, and Anna Rumshisky. 2016. #hashtagwars: Learning a sense of humor. *arXiv preprint arXiv:1612.03216* .

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

Dragomir Radev, Amanda Stent, Joel Tetreault, Aasish Pappu, Aikaterini Iliakopoulou, Agustin Chanfreau, Paloma de Juan, Jordi Vallmitjana, Alejandro Jaimes, Rahul Jha, et al. 2015. Humor in collective discourse: Unsupervised funniness detection in the new yorker cartoon caption contest. *arXiv preprint arXiv:1506.08126* .

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

Antonio Reyes, Paolo Rosso, and Tony Veale. 2013. A multidimensional approach for detecting irony in twitter. *Language resources and evaluation* 47(1):239–268.

Dafna Shahaf, Eric Horvitz, and Robert Mankoff. 2015. Inside jokes: Identifying humorous cartoon captions. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, pages 1065–1074.

Xinru Yan and Ted Pedersen. 2017. Duluth at semeval-2017 task 6: Language models in humor detection. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*. Association for Computational Linguistics, Vancouver, Canada, pages 384–388. http://www.aclweb.org/anthology/S17-2064.

Diyi Yang, Alon Lavie, Chris Dyer, and Eduard Hovy. 2015. Humor recognition and humor anchor extraction pages 2367–2376.

Bianca Zadrozny. 2004. Learning and evaluating classifiers under sample selection bias. In *Proceedings of the twenty-first international conference on Machine learning*. ACM, page 114.

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