#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

In this work, we present a new dataset for computational humor, specifically comparative humor ranking, which attempts to eschew the ubiquitous binary approach to humor detection. The dataset consists of tweets that are humorous responses to a given hashtag. We describe the motivation for this new dataset, as well as the collection process, which includes a description of our semi-automated system for data collection. We also present initial experiments for this dataset using both unsupervised and supervised approaches. Our best supervised system achieved 63.7% accuracy, suggesting that this task is much more difficult than comparable humor detection tasks. Initial experiments indicate that a character-level model is more suitable for this task than a token-level model, likely due to a large amount of puns that can be captured by a character-level model.

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 subjective, 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 goal 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 evaluate po-

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

tential predictive models based on a pairwise comparison task in an initial effort to leverage the HW dataset. The pairwise comparison task will be to select the funnier tweet, and the pairs will be derived from the labels assigned by the show to individual tweets. Initial experiments on the HW dataset will involve both unsupervised and supervised approaches.

There have been numerous computational approaches to humor 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 decomposes the notion of humor into two groups: humor and non-humor. This representation ignores the continuous nature of humor, while also not accounting for the subjectivity in perceiving humor. Humor is an essential trait of human intelligence that has not been addressed extensively in the current AI research, and we feel that shifting from the binary approach of humor detection is a good pathway towards advancing this work.

The broader impact of our dataset will be in the field of human-computer interaction. As evidence we highlight two systems that use humor in a human-computer dynamic. First, in (Wen et al., 2015) a computer chat agent attempts to suggest humorous memes/images in response to questions, creating an enjoyable experience for users. Dybala et al. (2013) offer a system that is better applicable to pure text. The system attempts to detect if the user is in a negative emotional state. If so, the computer offers humor in an effort to improve the user's mood. In terms of personalized interaction, it is not adequate to treat humor as binary, but rather as a continuous spectrum, seeking to understand the sense of humor unique to a given user.

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 title from Reuters news, proverbs and British National Corpus. Recently, Yang et al. (2015) used the same 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 target 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 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 or their work.

Finally, within the last year 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). While for the HW 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 a supervised model, 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%. Therefore, the modest improvement above random guessing dictates the difficulty of predicting degrees of humor.

3 #HashtagWars Dataset

3.1 Data collection

The following is our data collection process. First, when a new episode airs (which generally happens four nights a week, unless the show is on break) a new hashtag will be given. We wait until the following morning to use the Twitter search API[2] 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[3] that announces the top-10 tweets from the previous episode's hashtag. 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 web-site 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. A public release of the dataset must comply with Twitter's terms of use[4], which disallows the public distribution of users' tweets. 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 text. 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 web-site with a convenient user interface.

On the start page the user enters the id of the Tumblr post with the tweets in top 10. After that, we invoke Tesseract [5], an OCR command-line utility, to recognize the textual content of the tweets' images. Using the recognized content, the system forms a web-page 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's 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 re-

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

[3]`http://atmidnightcc.tumblr.com/`

[4]`https://dev.twitter.com/overview/terms`

[5]`https://github.com/tesseract-ocr/tesseract`

quiries going to the @midnight website. This is another aspect of the data collection system that could potentially be automated.

3.2 Dataset

Data collection has been in process for roughly seven months, producing a total of 9,658 tweets for 86 hashtags. The resulting data set is currently being used in a SemEval-2017 task on humor detection.

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.

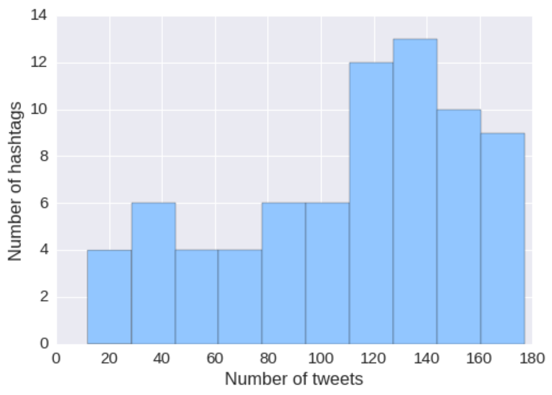

Figure 1: Distribution of the numbers of tweets per hashtag

Figure 2 represents an example of the tweets collected for the hashtag *FastFoodBooks*. Note that this hashtag requires an external knowledge about fast food and books in order to understand the humor. Furthermore, this hashtag illustrates how prevelant puns are in the dataset, especially related to certain hashtags. In contrast, the hashtag *IfIWerePresident* (see the Figure 3) does not require an external knowledge and the tweets are understandable without awareness about any specific concepts.

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 an external knowledge in order to understand the humor

4 Experiments

4.1 Evaluation Methodology

Both supervised and unsupervised approaches to this task can be evaluated using the same consistent methodology as follows. Using the tweets submitted for each hashtag, we construct pairs of tweets in which one tweet is judged by the show to be funnier than the other. The accuracy of prediction of the funnier tweet is then used as the evaluation measure. The pairs used for evaluation are constructed as follows:

(1) The tweets that are judged to be in the top-10 funniest tweets are paired with the tweets 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 another. The first and the second tweets in a pair are shuffled based on a coin flip.

The main evaluation measure is the micro average of accuracy on the individual test 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. We also include the following metrics: percentage of individual hashtags for which accuracy is above 50%, as well as the highest/lowest accuracy across all hashtags.

4.2 Unsupervised experiments

The first experiments we conduct are based on unsupervised methodology. The experiments are conducted on a total of 88,494 tweet pairs from 86 different hashtags.

4.2.1 Metrics

Following the methodology proposed by Radev et al. (2015), we apply the authors' three top-performing comparison metrics, namely LexRank (Erkan and Radev, 2004), as well as the positive and negative sentiment of the text (tweet in our case). In order to determine the sentiment of a tweet we use the TwitterHawk system (Boag et al., 2015), which placed first in topic-based tweet sentiment in SemEval 2015. We used the LexRank implementation available from the sumy library[6]. For a given hashtag, we calculate the individual LexRank scores of the tweets. The unsupervised methodology classifies the tweet with the greater value of a metric (feature) as the funnier tweet of the pair.

4.2.2 Results

The results of the unsupervised experiments are presented in Table 1. Despite the fact that the models achieved a good accuracy on several hashtags, the micro and macro averages are barely better than random guessing and even worse in the case of LexRank. We would expect that for hashtags where negative sentiment performed the best, the hashtags themselves would encapsulate some notion of negativity. In Table 2 we list five hashtags with the highest accuracy using the negative sentiment metric. Clearly the top-performing hashtag *MakeTVShowsEvil* has a strong sense of negativity. Unfortunately, this argument is weak for the four remaining hashtags, whose accuracy doesn't vary dramatically from the top-performing hashtag. Note Shahaf et al. (2015) achieved an accuracy of 61% using sentiment as an unsupervised metric for the NYCC dataset. This fact leads us to believe that the humor in the HW dataset is harder to recognize. Furthermore, their data set was much smaller and had only 754 pairs, whereas our dataset has 88k pairs.

[6]https://github.com/miso-belica/sumy

Hashtag	**Accuracy**
Make_TV_Shows_Evil	0.71
Hungry_Games	0.70
Twitter_In_5_Words	0.69
Sexy_Snacks	0.66
First_Draft_Cartoons	0.65

Table 2: The hashtags with best performance with negative sentiment metric.

4.3 Supervised Experiments

The supervised approach truly fulfills the notion of 'learning' a sense of humor, because we attempt to predict previously unseen hashtags based on a model trained on labeled tweet pairs. Unlike the unsupervised approach, a supervised system has the benefit of seeing what tweets are funnier based on the provided training data, with the hope it can generalize to hashtags not provided in the training data.

The experimental design for our supervised experiments is based on leave-one-out (LOO) evaluation. We withhold a single hashtag file for testing, and train on data generated from the remaining hashtag files. We create data points according to the methodology from Section 4.1.

On average, there are 112 tweets per file. Therefore, on average we train on 87,465 data points and test on 1,029 data points. Through the course of an entire LOO experiment, we test on a total of 88,494 data points, which is the result of 86 LOO experiments. We experiment with two different supervised methods. First, we train a feed-forward neural network (FFNN) based on hand-engineered features. Second, we experiment with a model that connects recurrent neural networks to a FFNN, with the goal of learning optimal tweet representations for

Features	**Acc Micro Avg**	**Acc >0.5**	**Max Acc**	**Min Acc**
LexRank	0.47	36%	0.65	**0.27**
Negative Sentiment	**0.51**	**57%**	0.71	0.22
Positive Sentiment	**0.51**	51%	**0.76**	0.21

Table 1: The results of the unsupervised experiment. Bold indicates the best features according to the corresponding metric.

our task. In our experiments, if the first tweet is funnier, the corresponding label is `1`. If the second tweet is funnier, the corresponding label is `0`. We place the funnier tweet based on a coin flip, so the resulting training/test sets have roughly balanced labels.

There are three factors that lead to the creation of a data point in the supervised system: two tweets and the hashtag that prompts the two tweets. Therefore, to fully represent a data point, we believe it needs to account for the two tweets as well as the hashtag, which is a unique aspect of the HW dataset. In the following sections, we explain three model that we experimented with: a feed-forward neural network with hand-crafted features, a token-level recurrent neural network (RNN) model and a character-level convolutional neural network (CNN) model.

4.3.1 Feed-Forward Neural Network Model

As the base classier, we used a fully connected neural network with three layers of sizes 256, 128, and 1, and ReLU activation functions. Using the manual features as the input, we trained the network with binary cross-entropy loss and Adam optimizer (Kingma and Ba, 2014) for 12 epochs using the Keras library[7]. We also experiment with the presence of dropout layers after the first two layers in order to prevent the model from overfitting the training data.

Hand-Crafted Features The following features are available for each tweet:

a) LexRank

b) Positive Sentiment

c) Negative Sentiment

d) Tweet Embedding

Furthermore, a hashtag is represented by its own embedding. For both the tweet and hashtag embeddings, we use 200-dimensional GloVe vectors, trained on 2 billion tweets[8]. Given the unique language of Twitter, we believe it is important to use Twitter-specific embeddings. The hashtag embedding is then the average of the individual hashtag tokens; the same holds true for the tweet embedding. If a token is not in the embedding corpus, its embedding defaults to the embedding for the 'unknown' token. For tweet tokenization we use a python wrapper for the ark-twokenizer[9]. We also believe that the use of embeddings trained on Twitter text will aid in providing external knowledge that is needed to perform at a high level in this task. For example, in Figure 2, one tweet makes a reference to Harry Potter. Since an embedding for the token 'potter' is present in the GloVe embeddings, this could potentially aid in the understanding of the tweet's humor.

4.3.2 Recurrent Neural Network Model

Given the widespread effectiveness of recurrent neural networks for language modeling (Mikolov et al., 2010; Sutskever et al., 2011; Graves, 2013; Bengio et al., 2006), we implemented a token-level RNN-based model with the goal of learning better representations for both tweets and hashtags, which can be fed into the same FFNN as manual features. Given a sequence of tokens from either a tweet or a hashtag, we convert it into a sequence of GloVe vectors. Each sequence of vectors is fed into a Long Short-Term Memory unit (LSTM) (Hochreiter and Schmidhuber, 1997), which consumes an input vector at each time-step and produces a hidden state. The final hidden state constitutes the vector representation for the two tweets as well as the hashtag. We concatenate the three vector representations and

[7]`http://keras.io`

[8]`http://nlp.stanford.edu/projects/glove/`

[9]`https://github.com/myleott/ark-twokenize-py`

provide it as input to the FFNN from the previous section. We apply rmsprop (Tieleman and Hinton, 2012) as the learning algorithm for this model.

4.3.3 Convolutional Neural Network Model

Often, a joke in a tweet is based on a pun, such as combing two words to make a new 'word'. For example, one of the top-10 tweets for the hashtag *DogJobs* uses the word 'barktender', combining the words 'bark' and 'bartender'. This property of the data leads to a high proportion of out of vocabulary (OOV) words. For example, in the GloVe embeddings that we used, the percentage of OOV tokens is 32.27%. Since a token level model cannot understand single-token puns, we introduce a new character-level CNN model for this task .

The model consists of two CNN layers of convolutions; sized 5 and 3, each with 100 filters and max pooling of length 2. The input to the convolutions layers is a trainable character embeddings of size 50. The output of the CNN layers is passed to a fully connected layer of size 256. The representations of two tweets, learned by the these layers, are concatenated and fed to the same FFNN as in the previous section.

4.3.4 Results

The results of the supervised experiments are presented in Table 3. Because we assign labels to the training examples based on a random coin flip, we performed three runs for each system and present the average score (as well as the average for the other metrics). We also present the standard deviation of the three runs for a given system across the various metrics. The feature types are as follows: Basic is the three features from the unsupervised experiments: lex rank, negative sentiment, and positive sentiment. HTE is an embedding for the hashtag from a specific file of tweets. TE are embeddings of the two tweets that constitute a single pair, one embedding for each tweet. DRPT indicates that we have added dropout of 0.5 between the fully connected layers of the FFNN. Because there is a noticeable performance gain when adding TE to the Basic+HTE system, this could potentially occur merely because of the added dimensions in the feature space. In order to address this, we experimented with the RTE feature, which is a random embedding for a given tweet, as opposed to the normal methodology for creating a tweet embedding. Finally, RNN and CNN are the systems described in Sections 4.3.2 and 4.3.3, respectfully. We also experimented with a character-level LSTM model, but it showed similar results while requiring a substantially longer training time.

5 Discussion

The low accuracies of the unsupervised methodologies suggest that such a simple approach does not work for this complex task. It is interesting to see that the positive sentiment and negative sentiment features perform almost identical. However, these features have quite strong negative correlation (-0.470). One hypothesis is that for certain hashtag, either positivity or negativity will play a more important role. The validity of this hypothesis is discussed further in relation to the supervised experiments below. Also of note in the results of the unsupervised experiments is the poor performance of LexRank. We believe this is because of the high variability of the language in the tweets, even within a specific hashtag. We also point out that in the work of Shahaf et al. (2015), the authors report accuracy below 50% when using n-gram perplexity as an unsupervised metric. The combination of these results dictate that language uniqueness is a poor unsupervised metric, regardless of dataset.

The complexity of this task, first revealed by the unsupervised experiments, is confirmed by the results of the supervised experiments. Two strong neural network models only surpassed random guessing by roughly 5%.

One goal of the Basic system was to determine if a supervised system could learn an effective weighting of the three basic features, allowing it to outperform the results of the unsupervised experiments. Another goal was to see if, by representing the hashtag, the system could learn for which hashtag a given basic feature is most important. However, the addition of the hashtag embedding to the Basic features actually creates a decrease in performance. One possibility is that the current hashtag representation is not able to facilitate the desired performance increase. Alternatively, the results show that the presence of the *tweet* embedding creates a noticeable in-

System	Acc Micro Avg	Acc >0.5	Max Acc	Min Acc
Basic	0.513 (±0.0035)	53.5% (±1.1628)	0.764 (±0.0185)	0.287 (±0.0187)
Basic+HTE	0.501 (±0.0007)	48.4% (±2.6854)	0.762 (±0.0165)	0.327 (±0.0084)
Basic+TE	0.542 (±0.0007)	65.9% (±0.6713)	0.769 (±0.0073)	0.310 (±0.0777)
Basic+HTE+TE	0.546 (±0.0075)	69.8% (±1.6444)	0.798 (±0.0294)	0.329 (±0.0068)
Basic+HTE+RTE	0.502 (±0.0114)	48.4% (±10.422)	0.711 (±0.0265)	0.287 (±0.0257)
Basic+HTE+TE+DRPT	0.554 (±0.0078)	72.1% (±3.0765)	0.765 (±0.0361)	0.364 (±0.0327)
HTE+TE	0.541 (±0.0058)	66.7% (±3.5524)	0.749 (±0.0507)	**0.371** (±0.0143)
RNN (token-level)	0.554 (±0.0085)	73.3% (±1.6444)	0.786 (±0.0779)	0.298 (±0.0150)
CNN (character-level)	**0.637** (±0.0074)	**92.4%** (±2.2076)	**0.864** (±0.0515)	0.359 (±0.0401)

Table 3: The results of the supervised experiments. Bold indicates the best system(s) according to the corresponding metric.

crease in performance. Having both together produces a very marginal increase in micro average, although the increase in percentage of hashtags with accuracy greater than 50% is non-trivial. Furthermore, the poor performance of the system with the random tweet embedding shows that even averaging of individual token embeddings can provide a useful representation of a tweet's semantics.

The presence of dropout in the FFNN appears to have a positive effect for the task, as the system that employed it has the highest accuracy on the task. This accuracy is also shared by the RNN system. However, it is interesting to note that the correlation of individual hashtag accuracies between the RNN and Basic+HTE+TE+DRPT systems is 0.557. This leads us to believe that even though the accuracies of the systems are the same, they are capturing different views of the data, and therefore perform better on different hashtags. This also suggests that an ensemble system could be effective for this task.

By comparing the performance of the RNN system with the HTE+TE system, we are able to see that in fact the RNN system is able to learn representations for the task that are more effective than simply averaging of token embeddings. We are able to make this claim by the fact that the representations learned by the RNN system feed into the same FFNN as the feature-based approach.

The superior performance of the CNN model, compared to the performance of the RNN model, suggests that even a complex neural network system cannot perform well on this task using only token-level information. Large amount of jokes in this dataset are based on puns, which leads to a large number of out of vocabulary words, even for embeddings trained on Twitter data. The fact that the character-level model performed substantially better than all other models suggests that this model can better represent OOV words (which, for example, is important for understanding puns) and use this information to decide which tweet is funnier.

One final analysis we perform is to determine if the test hashtags that are most similar to the training hashtags actually perform better than those that are less similar. To determine this, we represent a hashtag by its average embedding. We then hold a given hashtag out and calculate the cosine similarity with the average of the remaining hashtags' embeddings. This represents how similar a test hashtag is to the remaining hashtags for training. We then calculate the correlation between this similarity and the accuracy of the test hashtag. We did this for the results of the Basic+HTE+TE+DRPT system. Unfortunately, the correlation is relatively low (0.223). However, this low correlation could also be explained by the fact that averaging of individual tokens for the hashtag doesn't appear to be the appropriate representation for this task.

Lastly, we note the stability of results for the same systems across multiple runs. None of the systems (aside from the one with the random tweet embedding) have a standard deviation in micro accuracy above 0.01, which shows that even by randomly assigning labels to the dataset, the better systems are able to distinguish themselves.

6 Conclusion

In this paper, we have presented the HW humor dataset. We motivate the need for such a dataset, while also describing our collection process. Our dataset is several orders of magnitude greater than the only existing comparable dataset, the NYCC dataset. Lastly, we present the results of both supervised and unsupervised experiments. The results of our experiments show that this task cannot be solved with a simple token-level approach, and requires a more complex system working with puns at the character-level in order to solve the task with an accuracy that is substantially greater than random guessing.

There are numerous avenues for future work. We acknowledge that responding to these hashtags often requires external knowledge, such as titles of movies or names of bands. Our results show that semantic representations alone cannot capture this. In such cases, this external knowledge is mandatory to understanding why a tweet is funny. Systems that make effective use of external knowledge sources will have a better chance to recognize the humor in a tweet and will therefore have higher performance in this task.

An ambitious implementation for interacting with external knowledge sources is a Neural Turing Machine (NMT) (Graves et al., 2014). Interacting with a knowledge source requires discrete actions, such as querying/not querying, as well as deciding on the query string. Zaremba and Sutskever (2015) describe an algorithm for training an NTM with discrete interfaces. For example, an NTM might learn, for a given hashtag, which specific external knowledge source would be beneficial for deciphering the humor in response tweets, as well as how to determine which part of a tweet string refernces which external knowledge. Consequently, our dataset is of secondary interest for researchers who seek to interact with query interfaces via NTMs.

References

Francesco Barbieri and Horacio Saggion. 2014. Automatic detection of irony and humour in twitter. In *Proceedings of the International Conference on Computational Creativity*.

Yoshua Bengio, Holger Schwenk, Jean-Sébastien Senécal, Fréderic Morin, and Jean-Luc Gauvain. 2006. Neural probabilistic language models. In *Innovations in Machine Learning*, pages 137–186. Springer.

William Boag, Peter Potash, and Anna Rumshisky. 2015. Twitterhawk: A feature bucket approach to sentiment analysis. *SemEval-2015*, page 640.

Pawel Dybala, Michal Ptaszynski, Rafal Rzepka, Kenji Araki, and Kohichi Sayama. 2013. Metaphor, humor and emotion processing in human-computer interaction. *International Journal of Computational Linguistics Research*, 4(1):1–13.

Günes Erkan and Dragomir R Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research*, pages 457–479.

Alex Graves, Greg Wayne, and Ivo Danihelka. 2014. Neural turing machines. *arXiv preprint arXiv:1410.5401*.

Alex Graves. 2013. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*.

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

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

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*, pages 89–94. Association for Computational Linguistics.

Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

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*, pages 531–538. Association for Computational Linguistics.

Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernockỳ, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *INTERSPEECH*, volume 2, page 3.

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*, pages 208–215. Association for Computational Linguistics.

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*, pages 66–70. Association for Computational Linguistics.

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*, pages 1065–1074. ACM.

Ilya Sutskever, James Martens, and Geoffrey E Hinton. 2011. Generating text with recurrent neural networks. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 1017–1024.

Tijmen Tieleman and Geoffrey Hinton. 2012. Lecture 6.5-rmsprop. *COURSERA: Neural networks for machine learning*.

Miaomiao Wen, Nancy Baym, Omer Tamuz, Jaime Teevan, Susan Dumais, and Adam Kalai. 2015. Omg ur funny! computer-aided humor with an application to chat. In *Proceedings of the Sixth International Conference on Computational Creativity June*, page 86.

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*, page 114. ACM.

Wojciech Zaremba and Ilya Sutskever. 2015. Reinforcement learning neural turing machines. *arXiv preprint arXiv:1505.00521*.

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*, pages 889–898. ACM.