LING 573 Paper

Eli Bales, Pangbo Ban, Avani Pai, Hilly Steinmetz
University of Washington
Seattle, WA USA
{elibales, pbban, apai3, hsteinm}@uw.edu

Abstract

Our group hopes to tackle SemEval 2021 Task 7, HaHackathon, a shared task focusing on humor detection using short jokes sourced from Twitter and Kaggle. While this shared task has many subtasks, our team is first focused on the binary humor classification task and will expand into controversy classification later on. Due to the small training set, our current system uses RoBERTa's pre-trained model with an additional neural network classifier layer. Moreover, we have added a BERTweet model as much of the input data was sourced from Twitter. Based on the initial results, both models achieve around an approximately 0.93 F1 and accuracy score, with BERTweet providing slightly better results. We hope that adding semantic features and ensembling with a random forest will help remedy problems found in our error analysis. Additionally, we discuss why humor detection is such a difficult but important task, as well as our different considerations when designing and implementing our system.

1 Introduction

Humor is a form of figurative language that serves many social functions. It can make interlocutors feel closer to one another and introduce ideas in a non-intimidating manner (Cohen, 1999). On the other hand, humor has the potential to offend, as what one person may find funny, another may take offense to.

Humor, therefore, presents a unique challenge to NLP systems. Because humor involves long-distance word relationships, world knowledge, and pragmatic cues (Taylor, 2014), systems that classify humor need to preserve and attend to this linguistic information.

Being able to identify a joke can have several downstream applications: it helps improve the interaction between humans and computers, identifying controversial humor may point to hateful or offensive content, and recognizing if a joke is controversial may have the potential to assist in user-content moderation, as users may use jokes to veil bigoted or harmful content.

Our team designed a system to classify humorous text. We participated in the HaHackathon shared task (Meaney et al., 2021). In this report, we describe how we trained a feed-forward neural network on pre-training RoBERTa and BERTweet sentence-level embeddings to develop a baseline system that classifies text as humorous or not. We evaluated our models on the development data set, and found fine-tuning each model provided significant improvement. We discuss what we can do to improve the systems moving forward.

2 Task description

HaHackathon is a shared task from SemEval 2021. The text data used in the shared class were collected from Twitter and from the Kaggle Short Jokes dataset[1] and labelled by 20 human readers aged 18-70. Readers classified the texts as humorous or not and rated how humorous the texts were on a 1-5 scale. Readers also classified humorous texts as offensive and provided a rating of how offensive the text was on a 1-5 scale. To identify which jokes were controversial, any joke with a humor rating variance higher than a set threshold was marked "controversial" (Meaney et al., 2021).

We developed a system to classify texts as humorous or not, curating data by randomly splitting the shared task's training set of 8,000 texts into 6,400 training texts, 800 development/validation texts, and 800 test texts. Performance is evaluated on the training and test sets by calculating its F-score and accuracy. We then extended our system by training it on the controversial humorous texts, evaluating controversial classification performance the same way as humor classification evaluation.

[1] https://www.kaggle.com/datasets/abhinavmoudgil95/short-jokes

3 System Overview

Our baseline system encodes text as a sentence-level RoBERTa embedding and uses these embeddings to train a two-layer feed-forward neural network classifier. The RoBERTa tokenizer and pre-trained model were imported from Hugging Face's `Transformers` library[2]. The feed-forward classifier was created using `PyTorch` modules.

The training documents are then fed through the RoBERTa tokenizer, a byte-level BPE tokenizer, which generates the input for the model to create the embeddings. Trained on our input embeddings, we introduce a final neural-network classifier layer. Once the system has finalized, we tokenize and create embeddings for our development data, which we batch and run through the classifier.

The feed-forward classifier had hidden linear layers composed of 100 neurons. For optimization, our model uses stochastic gradient descent with a learning rate of 0.0001. We used a batch size of 32 for calculating gradients. The output layer was composed of a softmax function and we used cross entropy as our loss function.

To evaluate, our system takes the predicted labels for each document from the classifier output and compares them to the true labels. Accuracy and f1-score are calculated using the standard `scikit-learn` evaluations[3].

We implemented a variety of methods to improve upon the baseline model. Our baseline approach employed the vanilla RoBERTa model, so we fine-tuned it on the HaHackathon dataset to provide the model with the additional contextualization for the task at hand. Along the same vein, we also fine-tuned the BERTweet model to compare its performance against both the vanilla RoBERTa and fine-tuned RoBERTA models to gain a better understanding at how the domain of the training data affects its performance. We also tried a linguistically-informed approach to improving over the baseline model by incorporating named entity recognition (NER), letter counts, empathy-ratings (Fast et al., 2016), and Hurtlex (Bassignana et al., 2018) as linguistic features, which were reported to being beneficial to the HaHackathon task (Meaney et al., 2021). The ensemble architecture consists of a random forest classifier, which has been trained on the aforementioned lexical features, and the fine-tuned RoBERTa model. The output probability distributions of both models are then combined and fed into a logistic regression model to generate the final classification. The logistic regression model in this case, is learning how to weigh the outputs of the random forest classifier and the fine-tuned RoBERTa model.

[2] https://huggingface.co/docs/transformers/index

[3] https://scikit-learn.org/stable/

4 Approach

To begin, we created the training, development, and testing data sets from the shared task's 8000 provided texts. Such a split is random and the size of the data sets can vary depending on the purpose of those using our system. We chose an 80/10/10 split, but may move to a 90/5/5 split if we need more training data.

With only 8000 texts to split amongst a training, development, and testing set, our team decided the best approach would be to use a pre-trained model that we fine-tune and modify to fit our task. This approach was also used by many other teams participating in this shared task, with BERT, ERNIE 2.0, ALBERT, and RoBERTa all being utilized. Some groups ensembled different models and used hard voting to determine classes.

Our other design choices were also governed by the task at hand. We decided to go with the RoBERTa Base model for two reasons, the first being that it was one of the two LMs included in the topmost successful model architectures submitted in the original competition. The second reason is that our dataset is fairly small, so we decided to use an LM like RoBERTa to take advantage of its powerful word embeddings. Because a large portion of the data was drawn from Twitter, We also examined whether BERTweet (Nguyen et al., 2020) would perform better on the task than a large language model like RoBERTa.

Our group was successfully able to fine-tune both the base RoBERTa model and the base BERTweet model on our training data, training for only one epoch to not over-tune. Fine-tuning the base model was a common approach used by other groups, who also often fine-tuned for only one epoch (Meaney et al., 2021).

We built an ensemble classifier composed of the RoBERTa classifier and a random forest to try and capture certain orthographic and lexical features. We picked the Random Forest classifier in the experiment based on the comparison amongst five common classifiers trained on lexical features

System	Acc	F1
Initial	0.6013	0.7478

Table 1: Results of our intial system on humor detection with F1 and accuracy for the dev dataset

System	Acc	F1
RoBERTa	0.9237	0.9371
BERTweet	0.9387	0.9504

Table 2: Results of our fine-tuned systems on humor detection with F1 and accuracy for the dev dataset

by Khan et al. (2021), where the Random Forest achieves the best performance in the task. However, our initial tests of this model had poorer performance than the fine-tuned transformers. We are currently working on including a multitude of semantic features to help our model better identify humorous texts. One of such features is the recognition of named entities. In our error analysis, one problem we found was that our model struggled with understanding jokes about historical figures or organizations. To combat this, we utilized NER to extract which jokes were about real-world figures and concatenated this encoding into our embeddings. We have also looked into adding information extracted from using Hurtlex (Bassignana et al., 2018) or term-frequency-inverse-document-frequency (TF-IDF).

5 Results

The result of our initial system on the humor detection task with the metrics of accuracy and F1 is shown in Table 1. Amongst several runs, the accuracy of our system was between 60% and 61.5%.

For this preliminary experiment, we only used the development data to train the model to guarantee a quick and smooth run on the cluster considering the size of the RoBERTa model. Hence, we are not able to conduct a direct comparison between the performance of the baseline system for this humor detection shared task and ours, and thus the standard baseline result is not included in this report.

After fine-tuning our models and selecting the correct loss function, we saw a significant increase in our models' ability. Our results are shown in Table 2. BERTweet has a slightly better performance than RoBERTa does, which we attribute to a majority of the input data being drawn from Twitter. These are the results without ensembling with a random forest that targets the extra lexical features like NER, Hurtlex, and TF-IDF, so we hope that the combination of our fine-tuned models and the random forest classifier will further increase our results.

System	Humor	Offense
RoBERTa misclassified	2.00	0.66
BERTweet misclassified	1.98	0.61
Dataset	2.24	1.02

Table 3: Average humor and offense ratings of data misclassified by our BERTweet and RoBERTa models

6 Discussion

Our fine-tuned BERTweet and RoBERTa models performed quite well on the task despite being trained on only one epoch. BERTweet performed better on our development data set than RoBERTa, and this is probably a result of the fact that a large portion of the data consisted of tweets.

Some simple error analysis of the data revealed that our BERTweet or RoBERTa models made patterned errors. The models seemed to have difficulty classifying more sarcastic humor or humor that is written in a more "matter-of-fact" tone. For instance, both models misclassified the joke "Years from now, historians will look back on this period of American History and move to Canada." They also seem to perform poorly on jokes that require world-knowledge or jokes that involve some sort of linguistic or orthographic knowledge (for instance, they both misclassified the joke "Our attention spans these days are" as not humor).

Interestingly, our error analysis also revealed that the models tended to misclassify jokes less offensive and slightly less confidently rated as humorous (see Table 3).[4] The fact that the offense ratings were lower among the misclassified sentences could mean that the models are sensitive to the fact that humor and controversy ratings are correlated. The misclassified jokes were only slightly less funny than the data set average, which could indicate that the models were picking up on linguistic cues less directly associated with humor. (Meaney et al., 2021).

While the language models had competitive results with the baseline, our ensemble model with achieving an accuracy of about 78%c did not. The

[4]The raters were asked to rate on a scale of 1-5 whether the text *intended* to be humorous, not whether it was humorous.

intent behind the ensemble approach was to capture the world-knowledge associated with humor via features like NER, punctuation counts, lexical categories, and hurtful words. Despite the poor performance, we still believe these features to be useful: an SVM trained on just those features is able to achieve a similar accuracy. On the other hand, we will further examine the significance and the interaction among the current features while incorporating more linguistic features. We hope to debug and refine this ensemble classifier when we adapt our system to detect controversial humor.

7 Conclusion

Our current system architecture produces strong results with a clear idea of where our errors lie. Our fine-tuned transformer models have accuracy scores on our development data that are near the top 10 submissions for Hahackathon's subtask 1a, and we only need a 1-2% improvement to break that threshold. We hope that introducing new features while ensembling with a random forest will help cover our model's lack of linguistic knowledge and real-world context, increasing our results on our evaluation data enough to approach HaHackathon's top 10 submissions and successfully adapt to detecting controversial humor.

References

Elisa Bassignana, Valerio Basile, and Viviana Patti. 2018. Hurtlex: A multilingual lexicon of words to hurt. In *CLiC-it*.

Ted Cohen. 1999. *Jokes: philosophical thoughts on joking matters.* University of Chicago Press, Chicago.

Ethan Fast, Binbin Chen, and Michael S. Bernstein. 2016. Empath: Understanding topic signals in large-scale text. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, CHI '16, page 4647–4657, New York, NY, USA. Association for Computing Machinery.

Dan Jurafsky and James H. Martin. 2022. *Speech and Language Processing*, 3rd edition (draft) edition. Online.

Atif Khan, Muhammad Adnan Gul, Abdullah Alharbi, M Irfan Uddin, Shaukat Ali, and Bader Alouffi. 2021. Impact of lexical features on answer detection model in discussion forums. *Complexity*, 2021.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*.

J. A. Meaney, Steven Wilson, Luis Chiruzzo, Adam Lopez, and Walid Magdy. 2021. SemEval 2021 task 7: HaHackathon, detecting and rating humor and offense. In *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021)*, pages 105–119, Online. Association for Computational Linguistics.

Dat Quoc Nguyen, Thanh Vu, and Anh Tuan Nguyen. 2020. BERTweet: A pre-trained language model for English Tweets. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 9–14.

Julia M. Taylor. 2014. Linguistic theories of humor. In Salvatore Attardo, editor, *Encyclopedia of Humor Studies*, volume 2, pages 455–457. SAGE Reference, Los Angeles, CA. Topic overview.

Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Well-read students learn better: On the importance of pre-training compact models. *arXiv preprint arXiv:1908.08962v2*.