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. Off initial results, our model can correctly identify humorous texts but struggles at identifying non-humorous texts. We hope that fine-tuning the RoBERTa model itself, adding semantic features, and introducing a recurrent neural network will remedy these downfalls. 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). Additionally, humor has the potential to offend and 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. Additionally, controversial humor may point to hateful or offensive content, which is using humor to veil its intentions. Recognizing if a joke is controversial may have the potential to assist in user-content moderation.

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 sentence-level embeddings to develop a baseline system that classifies text as humorous or not. We evaluated our model on the development data set, and found little improvement over chance. We discuss what we can do to improve the system 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 0-4 scale. Readers also classified humorous texts as offensive and provided a rating of how offensive the text was on a 0-4 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 6,600 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 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 `PyTorch`'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 64 for calculating gradients. The output layer was composed of a softmax function and We used binary 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].

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 current implementation only fine-tunes the classification layer based on the input embeddings, but we plan to later fine-tune the base model as well. This was a common approach used by other groups, who often fine-tuned for only one epoch (Meaney et al., 2021). Additionally, we are looking to possibly include semantic feature detection to help our model better identify humorous texts. Furthermore, we are looking to upgrade our neural network to a recurrent neural network to better capture long-distance dependencies and hopefully the "structure" of written jokes.

[2]https://pytorch.org/

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

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

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.

6 Discussion

Humor detection poses an interesting challenge for choosing how to preprocess input data. Common preprocessing procedures may normalize the data by converting all letter characters to lowercase, or may remove punctuation for clarity. However, in written form, humor and jokes make extensive use of all available tools to convey emotion, inflection, and other subcomponents of communication that are less recognizable in writing. Take the following joke from the training data as an example, "ME WHEN A NORMAL BUG IS ON ME: Eww. ME WHEN A LADYBUG IS ON ME: Evening, Ma'am." The capitalization helps the reader recognize the two different scenarios being discussed, whereas the punctuation, mainly the semi-colons, separates the scenario-setup and the author's reaction to the scenario. This is a common structure for humorous content online, and excessive preprocessing of the data could reduce a model's ability to recognize this structure. For this reason, our team used as little preprocessing as possible, but may explore how drastically preprocessing affects the model down the line.

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. Despite this, our overall accuracy lingered at around 60%, which is actually near the proportion of humor examples in our dataset. This occurred because our model was initially outputting the same probability distribution for each input data example. Once we initialized the weights however, we started getting varied distributions. This increased our accuracy by around 1%, so we decided to also try pretraining the classifier layer (Turc et al., 2019).

As mentioned earlier, we will also be fine-tuning the RoBERTa model so as to contextualize the embeddings further for the joke domain. We will also look into incorporating semantic features to improve the performance of the model over the baseline. Doing so will help make the most out of the limited data that we have. Features that provide additional information about joke structure and pragmatics, which are intuitively integral to the recognition of a statement as a joke, could provide similar clues to the model as to what is humorous beyond the general semantic and positional information supplied by RoBERTa embeddings.

7 Conclusion

We haven't seen any improvements in classification over chance. Our classifier had an accuracy of 60% score on the development set – 60% of the development set's texts are humorous. But, a working baseline system provides a springboard to improve the model by testing different hyperparameters, adding features, and modifying model architectures. Moving forward, we hope to improve the system by fine-tuning RoBERTa (or another transformer-based model) to this task. We also hope to examine the possibility of adding additional features to our model inputs.

References

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

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

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.

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.*