HaHackathon 2021: Humor and Controversy HaHaHacked

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

Abstract

Our group has selected 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's main task is binary humor classification, and our adaptation is controversy detection. Due to the small training set, our 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. For humor classification, our ensemble system achieved a 0.95 Accuracy and 0.96 F1 on the evaluation set. As for controversy detection, our models achieve a 0.53 Accuracy and 0.61 F1. 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 as well as 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. Additionally, our group selected a multitude of features which we identified our model may find useful through error analysis. Our final system is a ensemble system composed of a fine-tuned RoBERTa model and a feed-forward neural network which combine logits for a final logistic regression layer.

2 Task description

HaHackathon is a shared task from SemEval 2021. The text data used was 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

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

by training it on the controversial humorous texts, evaluating controversial classification performance the same way as humor classification evaluation.

3 System Overview

After testing a variety of configurations on our validation set, our final system was composed of two stacked classifiers—a fine-tuned RoBERTa classifier and a multilayer perceptron—with a metaclassifier that takes the concatenated logits of the two classifiers to determine the output. A visual outline of the ensemble can be found in Figure 1. Additional details on our other configuration tests can be found in the Approach section below.

We used Hugging Face's `Transformers` library to fine-tuned the RoBERTa-base pretrained model.[2] We also used the library's RoBERTa tokenizer to tokenize the input sentences. The training documents are then fed through the RoBERTa tokenizer, a byte-level BPE tokenizer, which generates the input for the model to create and classify sentence-level embeddings.

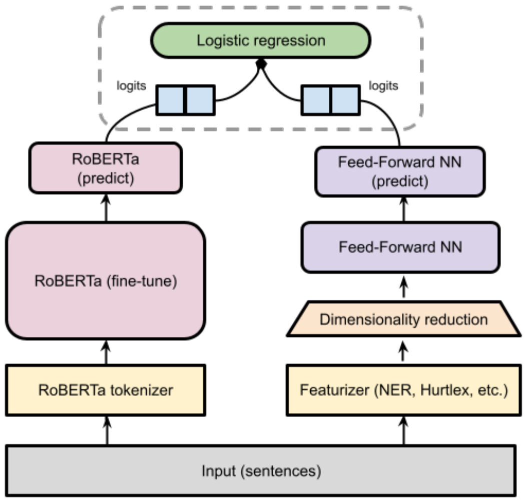

Figure 1: Illustration of ensemble architecture.

The feed-forward classifier was created using `PyTorch` modules. It comprised of 3 hidden layers of 40 neurons with a dropout rate of 0.4. The classifier was trained on linguistic features we identified as being relevant to the classification task, which were named entity recognition (NER), letter counts, word counts from features of two lexicons—Empath (Fast et al., 2016) and Hurtlex (Bassignana et al., 2018)—as linguistic features. Some of these features were chosen because other teams reported them as being beneficial to their models (Meaney et al., 2021), while others were chosen by hypothesis. Feature vectors were constructed from these inputs. Features were ranked by their ability to predict the labels in the training. The lowest scoring 30% of features were removed from future feature vectors before the MLP was trained on the resulting vectors.

Both models outputted logit vectors of length 2. The concatenated vectors were used to train a metaclassifier, which was a logistic regression model. The metaclassifier was trained using stochastic gradient descent and cross-entropy loss.

To avoid over-fitting the ensemble, we trained the metaclassifier on out-of-fold predictions. The training data was split into 3 folds using `scikit-learn`. The first two folds were used to train the fine-tuned RoBERTa classifier and MLP classifier. The last fold was used to generate predictions from these classifiers and train the metaclassifier on those predictions. The folds were stratified so that they preserved label frequency. The process continued k-times, until the metaclassifier was trained on all the training data once. Over the course of one epoch, the two base classifiers were trained on the all the training data twice. To avoid over fitting over several epochs, we increased the dropout rate on the RoBERTa model to 0.165.

We decided to set a random seed as well, since the ensemble did not train as predictably on the adaptation task, and we wanted to be able to compare across our experiments. Tables of hyperparameters and optimizers used for our final ensemble configuration can be found in Appendix A.

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.

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

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

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 a 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 successfully fine-tuned 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. Our selection of the Random Forest classifier in the experiment is based on the comparison amongst five common classifiers trained on lexical features 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 therefore subsumed further improvements into our model from four dimensions. First, beside the random forest ensemble, we built and experimented on the structure of a neural network featurizer and the logistic regression ensemble, which was selected as the main structure of our final model. Second, we included 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, we noticed that our model struggled with understanding jokes about historical figures or organizations. To combat this, we utilized Name Entity Recognition (NER) to extract which jokes were about real-world figures and incorporating this encoding into our embeddings via concatenation. Beside NER, We have also incorporated features based on the Hurtlex (Bassignana et al., 2018) which are categorized hurtful terms, as well as delta term-frequency-inverse-document-frequency (Delta TFIDF) (Martineau and Finin, 2009), letter and punctuation counts, and sentence-level empathy ratings (Fast et al., 2016) in the final model. Third, we further incorporated the featurizer with two kinds of feature selection techniques in dimension reduction, which are K-Best and Principal Component Analysis (PCA) provided by the `scikit-learn` package. After several experiments, we picked K-Best which had outperformed PCA in the accuracy and F1 scores of both tasks. Last but not least, we have implemented two different training approaches considering the potential over-fitting of our stacked model. One method is the training of the meta-classifier on out-of-fold predictions from the RoBERTa model and the MLP, while the other is the late fusion of outputs from these two models. We applied the K-fold approach in our final model due to its greater performance in boosting the accuracy and the F1 score of our model and its lowered possibility of over-fitting the training data.

5 Results

The result of our system on the humor detection primary task on different stages with the metrics of accuracy and F1 is shown in Table 1, and the result on the controversy detection adaptation task is exhibited in Table 2. As more improvements in model implementation has been incorporated, the accuracy and F1 score of our model on two tasks noticeably increased.

In the preliminary experiment of our baseline model, we only used the development data to train the model without fine-tuning to guarantee a smooth run on the cluster considering the size of the RoBERTa model. After fine-tuning our models and selecting the correct loss function, we saw a significant increase in our models' ability. BERTweet has a slightly better performance than RoBERTa does, which we attribute to a majority of the input data being drawn from Twitter.

Starting from that, we further implemented the ensemble technique with a neural network classifier as the featurizer of lexical features and a logistic regression meta-classifier, which gave us better accuracy and F1 scores on the development set of both tasks compared to that of our D3 model in the previous stage. Our result of the evaluation set on the primary task is 0.002 off the top 10 leaderboard of the shared task, and our final stacked model achieved top 10% accuracy but not F1 on the adaptation task. Notice that our evaluation set differed from the evaluation set of the paper in where we have compiled 10% of our training data.

6 Discussion

Our fine-tuned BERTweet and RoBERTa models performed quite well on the primary humor detection task despite being trained on only one epoch. BERTweet performed better on our development data set than RoBERTa, which we attribute to the in-domain advantage of BERTweet's training data.

Some simple error analysis of the primary task data revealed that both the fine-tuned BERTweet and RoBERTa models made patterned errors. The models had difficulty classifying more sarcastic humor or humor that is written in a "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." Our models also frequently misclassified 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 that had low offense ratings and relatively low humor ratings (see Table 4).[3] 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).

Our ensemble model also performed well, landing us in the top ten for the primary task and the adaptation task, although we used a held-out portion of the training data as our evaluation set, rather than the evaluation set available on CodaLab. We theorized that the combined power of the BERTweet and RoBERTa language models coupled with lexical features such as the Hurtlex lexicon and named-entity recognition would enable the model to extract salient signals towards identifying controversial topics, thereby facilitating controversy detection. While the main motivation was to perform well on the the adaptation task, we also saw slight improvements, around a 0.01 point increase in both F1 and accuracy, over the fine-tuned models for the primary task as well.

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

Although our model preformed comparatively well with the leaderboard on the adaptation task, the overall numbers are still rather low. This task proved to be the most difficult out of all the tasks in HaHackathon. Our model's accuracy did no better than chance at predicting whether or not a joke was controversial. The ensemble model tended to predict that the joke was controversial as opposed to not, about 81% of the time. The training data is well-balanced with a nearly equal quantity of controversial jokes versus non-controversial jokes. There is no clear pattern between controversial and non-controversial jokes, both in format and content, so it is difficult to speculate as to why the model over-predicts jokes as controversial. It could be the case that our lexical features did as intended and made salient information pertaining to particular groups or targets as well as hurtful sentiment, therefore leading the model to over-predict as many of the jokes poke at specific groups of people (women, Muslim people, black people, people with disabilities, etc).

Additionally further examination of the data and task description revealed that the definition of controversy is perhaps a little unintuitive. A joke is deemed controversial based on whether the variance between annotated humor ratings is greater than that of the training dataset variance as a whole. In other words, a joke is controversial if there are varying opinions on whether or not is was funny. Our understanding of controversy is more related to offensiveness - a joke is controversial if the subject matter pokes at a sensitive topic or marginalized group and this is heavily influenced by the socio-cultural and -political context at the time during which annotation took place.

Our approach clearly reflects this thinking and can partially explain why our model struggles to identify controversy. Additionally, the data supports the definition of controversy as a function of varying "funniness", as the Spearman correlation coefficient between humor rating and controversy in the training data is around 0.213, whereas it is -0.038 between offense rating and controversy. This shows that there is a stronger relationship between humor rating and controversy than between offense rating and controversy.

In order to do well in the controversy detection

System	dev		eval	
	Acc	**F1**	**Acc**	**F1**
baseline	0.6013	0.7478	-	-
fine-tuned RoBERTa	0.9237	0.9372	-	-
fine-tuned BERTweet	0.9387	0.9504	-	-
ensemble	0.94	0.9514	0.9538	0.9627

Table 1: Results of our system on the humor detection primary task with F1 and accuracy. he baseline results used a different evaluation set.

System	dev		eval	
	Acc	**F1**	**Acc**	**F1**
baseline (Meaney et al.)	-	-	0.4731	0.6232
ensemble	0.5051	0.5974	0.5304	0.6054

Table 2: Results of our system on the controversy detection adaptation task with F1 and accuracy. The baseline results used a different evaluation set.

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 on training data

	Predicted 1	Predicted 0	Total
Labeled 1	181	67	248
Labeled 0	177	68	245
Total	358	135	

Table 4: Controversy Detection Classification Matrix of Ensemble Model with Lexical Features

task, it would be more lucrative to make the model sensitive to the "funniness" of the joke as opposed to its potentially offensive content. Given that "funniness" is highly subjective and considering the main downstream application of a controversy detection model as being content mediation in online forums and social media, a better approach might be to design a model that takes both user interaction signals as well as linguistic signals as input. This way, if a tweet or post is suddenly trending and receiving lots of *mixed* types of attention, it can quickly be flagged as controversial and requiring moderation/mediation. Another method to improve performance on this task could be to train the model to predict the humor rating, and add a classifier head to predict whether it is controversial, which aligns with how the data was annotated. Based on the leaderboard results however, most models that performed well on the humor rating regression task did not do particularly well on the controversy detection task. This lends further support toward a model design that takes human signals directly as input in order to do well on a task that is inherently subjective.

7 Conclusion

While our adaption task barely does better than chance, our placement within the top 10 or within a few spots of the top 10 for the controversy and humor detection tasks respectfully allows us to be quite happy with our results. Through error analysis, we feel we have a clear understanding of where our model fails and succeeds, and what we would do further down the line if we were to continue with this system. Choosing a pre-built language model like RoBERTa and BERTweet proved to be quite successful due to their robust embeddings, and ensembling these models with lexical features is what brought our system into the top 10 for controversy. While we approached this shared task believing humor and jokes to be a difficult hurdle for NLP due to their variance among human judges, we were pleasantly surprised at what the correct application of design approaches, embeddings, and lexical features may achieve.

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*.

Justin Martineau and Timothy W. Finin. 2009. Delta tfidf: An improved feature space for sentiment analysis. In *ICWSM*.

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*.

Appendix A: Configurations & Hyperparameters

Learning rate	2e-5
Batch size	32
dropout	0.165
Optimizer	AdamW

Table 5: Hyperparameters and Configurations for RoBERTa

Learning rate	8e-3
Batch size	32
dropout	0.40
Optimizer	Adagrad
Hidden Layer Size	40

Table 6: Hyperparameters and Configurations for MLP

Learning rate	1e-2
Batch size	32
Optimizer	SGD

Table 7: Hyperparameters and Configurations for Logistic Regression Metaclassifier