Improving Humor Detection Model for Mandarin

Sida Zhong
EECS
University of Michigan
Ann Arbor, Michigan, USA
astar@umich.edu

Zhong Zheng
EECS
University of Michigan
Ann Arbor, Michigan, USA
ongzheng@umich.edu

Vijay Sharma
EECS
University of Michigan
Ann Arbor, Michigan, USA
vsharm@umich.edu

1. Introduction

Humor is used by humans every day to convey emotions and ideas in almost all cultures and languages. Understanding humor is an essential task for the NLP community because it has wide applications in many fields, such as chatbots and QA systems. However, humor is too subjective and difficult for natural language models to understand. Currently, most advanced humor detection models are based on the analysis of English [3], while other languages are underrepresented in this field. Therefore, our project aims at applying the advanced NLP techniques to Mandarin and improving the performance of the Mandarin humor detection model.

Aiming to improve the performance of the Mandarin humor detection model, our project has two tasks. Task 1 is a binary classification task: given a Mandarin text, we want the machine to classify whether it is humorous or not. Task 2 is a ternary classification task, in which we will predict the degree of humor. The degree of humor is measured by three levels. Zero is a little humorous, one is moderately humorous, and two is the most humorous. Given a joke, our model will output its humor level. For example, if the input is “这是一个笑话”, our model will output a degree of 0, 1, or 2. We will refer to these two tasks as Task 1 and Task 2 in the rest of the paper.

Our main contributions are that we propose a new model which combines pretrained Chinese BERT encoder, LSTM, and selected features. After training and testing in a Mandarin dataset from a competition [1], we achieve an F1-Score of 0.832 on Task 1 and outperform the previous best open-source model by 0.109. As for the ternary classification task, we achieve the F1-Score of 0.521, which outperforms the previous best model by 0.02.

During our project, we distributed the work evenly. Sida implemented the Naive Bayes model, translated the Mandarin texts into English through the Google API, and found theories about candidate features to capture humor. Zhong implemented the simpletransformers classifier based on BERT, built the BERT + LSTM model, and integrated selected features into the model. Vijay focused on transferring the English model into Mandarin, including retraining ColBERT in Mandarin. He also implemented the RoBERTa simpletransformers models, made plots, and did feature engineering. We all wrote roughly the same amount on slides and reports. All of the work mentioned above will be discussed later in this report.

2. Related Work

We refer to two previously constructed models and a set of competition results to inform our

investigation. We also outline what is novel about our work.

2.1 LSTM for Mandarin

The first model we reference is a bidirectional LSTM model created by Huanyong Liu trained for binary humor detection in Mandarin [2]. This was the best open-source Mandarin model we found for this task. This model uses a Chinese word embedding named token_vet_300. It contains word embeddings of length 300. The model uses word tokenization, splitting each sentence into single words and converting each word into corresponding word embedding if it exists in the dictionary. For example, if the sentence is “我吃 苹果”, which means “I eat apples”, after the tokenization, it would be “我”, “吃”, “苹”, “果”. With that, we can tokenize words and calculate the word embeddings. Then, the model takes the embedding sequence as input.

We believed that this model could be improved as it achieves a 0.723 F1-score on Task1. We noticed that the model did not use any custom features or pre-trained models, so we identified these as potential areas of improvement.

2.2 ColBERT for English

The second model we reference is called ColBERT and is a model designed by Issa Annamoradnejad for binary humor classification in English [3]. This was the best open-source English model we found for the task. This model splits texts by sentences and computes embeddings for each sentence. Then, it passes the output embeddings for each sentence into parallel feed-forward neural networks, concatenates the resulting outputs, and finally passes the concatenated data into another feed-forward neural network for classification.

This model achieved a 0.982 F1-score on a large English dataset which led us to believe we could apply insights from the ColBERT model to the Mandarin version of the task. We were mainly curious about how the ColBERT architecture would perform on Mandarin text, and more generally if a BERT-based model would prove more powerful than the vanilla LSTM used by the best open-source Mandarin model.

2.3 Competition Results

We also refer to competition results from the 2019 conference of Chinese Computational Linguistics which is where we obtained our data [1]. The most significant results are that of the best performing model of the competition, which achieved a 0.9488 F1-score on Task 1 and 0.5011 F1-score on Task 2. We noticed that this Task 1 score was much higher than the Task 1 score of 0.723 achieved by the highest performing open-source model we found. The competition model that achieved this score is not accessible to the public, however, so we aim to achieve comparable or better results to this model while also making our process and model open-source.

2.4 Our Contributions

We ultimately improve on the previous best open-source Mandarin model in two main ways. First, we use BERT to embed the text of each joke instead of sequential standard word embeddings. Second, we add additional features to the BERT embeddings before passing them into the second part of our model. These advancements and the resulting performance improvements will be discussed in depth throughout the rest of the paper.

3. Methodology

3.1 Dataset Description

Our dataset was published in 2019 at The Eighteenth China National Conference on Computational Linguistics [1]. The dataset has two parts. First, Task 1 is for binary

classification. It contains the text of the **jokes** in Mandarin, the **id** of the text, and the **labels** to distinguish if it is a humorous text or not. For example, one piece of data in subset one would be {id: 1, jokes: "这是一个笑话", labels:0}. There are 16,480 texts for this task. 11,536 of them are labeled as 1, meaning humorous, and 4,884 of them are labeled as 0, which means not humorous. Next, Task 2 is for ternary classification, which has the same column of **jokes** and **id,** but the **labels** classify the humorous texts into three degrees. 1 is a little humorous, 3 is moderately humorous, and 5 is strongly humorous. For the convenience of training, we replace 1, 3, and 5 with 0, 1, and 2 respectively. Task 2 contains 16,670 samples. 8,624 are weak humor, 4,408 are moderate humor, and 3,638 are strong humor. We believe that this dataset is a perfect fit for our task since it contains both texts and labels for binary classification and ternary classification.

3.2 Data Preprocessing

Since the test set from the competition masks their labels, we create our own test set by randomly splitting the original training set in the competition to be our own fixed training and testing set. We noticed that the original training sets for both tasks were not balanced which would likely undermine the performance of our models on the test data. To achieve better performance for our model, we use the random library in Python to randomly select data from different labels and balance the training datasets. For the test data, we balance it with the same methods and make performance metrics more meaningful by ensuring that we evaluate our models with equal weight given to each label. As a result, we can safely use accuracy and F1 score to evaluate our model.

Overall, we have balanced train and test datasets for both Task 1 and Task 2. For Task 1, the training dataset has 8,000 samples, and the testing dataset has 1,600 samples. For Task two, the training dataset contains 9,000 samples and the testing dataset has 1,800 samples.

3.3 Models Description

3.3.1 Naive Bayes Model

First, we train a Naive Bayes classifier on our training data by using the CountVectorizer from the scikit-learn library. We include both unigram and bigram in our bag-of-words and apply add-k smoothing, where k = 0.4, to calculate the probability of the test text under each category. We use F1-score to evaluate our Naive Bayes model. This model provides a good starting point for us.

3.3.2 ColBERT and simpletransformers

Next, we tried four models inspired by ColBERT, the highest performing English model.
For Task 1 we first apply the ColBERT model to our Chinese dataset. This is done by changing the English BERT encoder in ColBERT with a Mandarin BERT encoder, and by retraining the entire model with our Mandarin training data. The model can then be effectively applied to the Mandarin training data. We completed this process for Task 1 and obtained an F1-score of 0.7461.

Then, we translate the entire Mandarin dataset to English and retrain ColBERT to classify the translated test set. The Mandarin data was translated using the Google Translate API. We found it was necessary to retrain ColBERT on our translated English training data, as ColBERT pretrained on the original English data did not

perform well on the translated texts. Ultimately, by retraining on the translated English data we achieved an F1-score of 0.7444 on the translated test data.

Third, we use the simpletransformers Python library to train a BERT-based classification model [5]. Different than ColBERT, the simpletransformers classifier compute the embedding based on the whole texts through a BERT base Chinese encoder and then do the classification work. This model outperforms both ColBERT methods on Task 1 with an F1-score of 0.8098.

Lastly, we train a simpletransformers RoBERTa-based classification model. This model is the same as the simpletransformers BERT-based model but uses a Mandarin RoBERTa model instead of BERT to compute the embeddings. This method achieved an F1-score of 0.7981 on Task 1 which is almost the same as the BERT-based simpletransformers model.

For Task 2, we apply both simpletransformers models but do not retrain ColBERT due to lack of promise on the binary classification task. The BERT-based model achieves an F1-score of 0.4822 and the RoBERTa-based model achieves 0.4606.

Ultimately, out of all the candidate models, we decide that a BERT-based classification model that is simpler than ColBERT is the most promising for both Task 1 and Task 2. Due to the slightly higher F1-score of the BERT-based

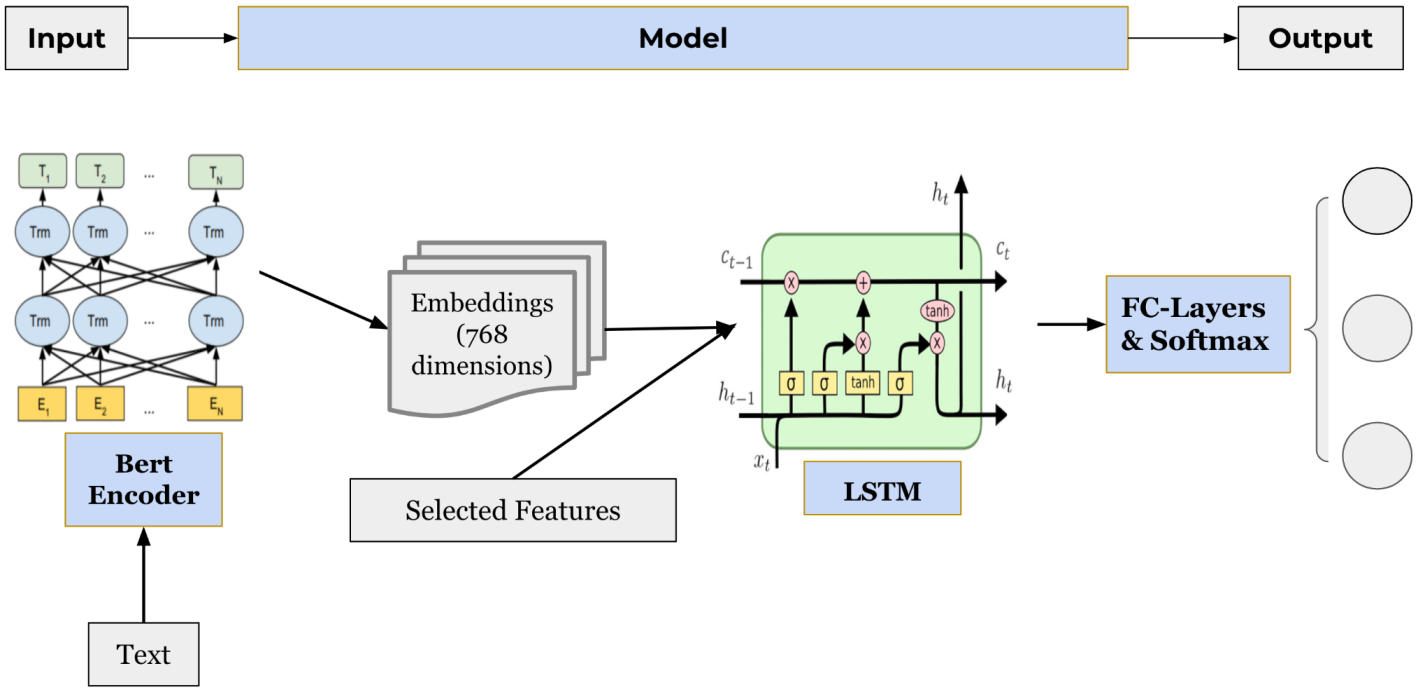

Fig. 1: Model architecture

model on Task 2, we opt to use BERT instead of RoBERTa.

3.3.3 BERT + LSTM model.

To further improve our classification accuracy, we integrate the idea of BERT and LSTM and build a more powerful model with BERT, LSTM, and custom-selected features (Fig. 1).

We first input our raw text into a pretrained Chinese BERT model named Bert-Base-Chinese to compute the embeddings with the self-attention mechanism of BERT. The embedding contains 768 dimensions as it is a base BERT model. Then, we append a feature vector to the 768-dimension embedding, resulting in a more comprehensive feature embedding in a higher dimension. We then send the feature embedding to a bidirectional LSTM encoder and compute a 400-dimensional new vector. Finally, we add a fully connected layer with the softmax activation function to do the actual classification work.

As a result, this model gives us the best performance, so we will choose it as our final model and talk about it in detail in later parts.

4. Experiment

4.1 Feature Selection

We looked into the dataset to see which features can indicate a humorous text.

4.1.1 Punctuation

Punctuation is a good indicator of humor. When language is stated in some special forms, such as dialog or questions, it is more likely to be humorous. Punctuation is a good indicator of such information. For example, colons are good indicators of dialog. As a result, we selected four punctuation marks as our features.

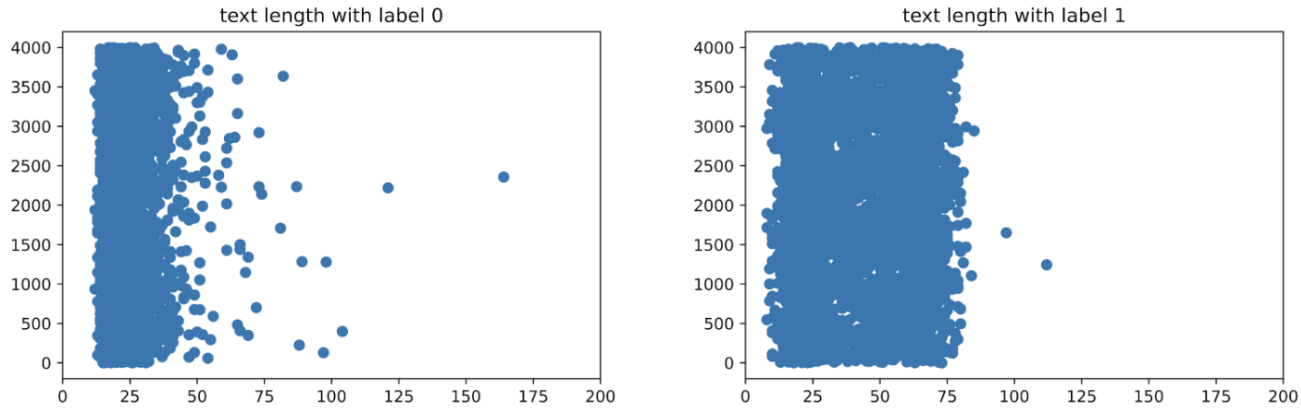

Fig. 2: Task 1 text length distribution differences among labels

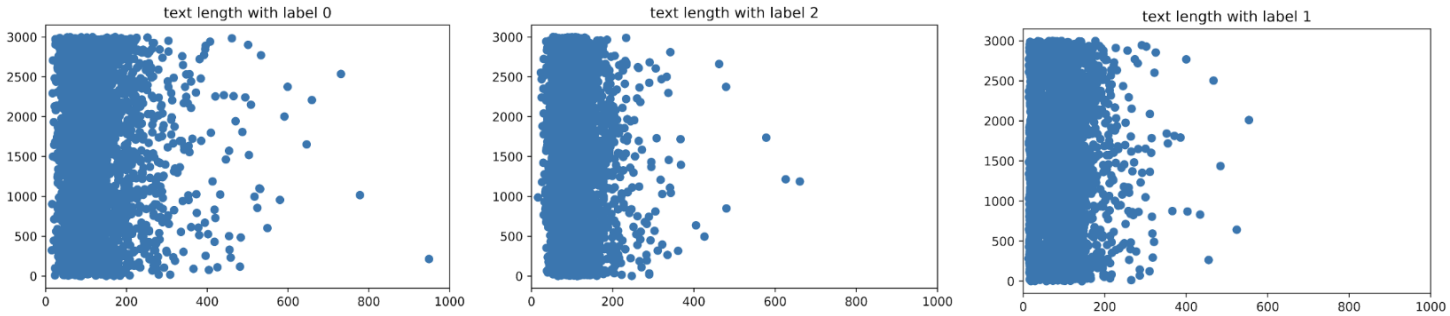

Fig. 3: Task 2 text length distribution differences among labels

- **Colons**: Colons suggest dialog. Many jokes involve several characters, and they interact with each other through conversation to generate humor.
- **Question Marks:** Question marks suggest questions. Many jokes generate humor by giving surprising answers to common questions.
- **Exclamation Marks:** Exclamation marks suggest strong emotions. Non-humorous texts are usually just simple narration without any emotions.
- **Chinese Comma (、):** Comma suggests some parallel items. Humor can be generated when some unrelated items are mentioned together in an unusual way.

4.1.2 Text Length

Although some short texts can also generate humor, most short texts we find are just simple narration that does not contain any humor. Therefore, we plot the distribution of text length with respect to different labels (Fig. 2, 3). We find that different labels can have different text length distributions, especially for Task 1. Therefore, we choose to include text length information in our model.

4.1.3 Special Groups:

Many humorous jokes are story-based, in which characters have different roles, such as dad and mom, wife and husband. If some of the groups are mentioned, the text is more likely to be a humorous story. Therefore, we choose to use special groups as a feature. The special groups we choose include mom and dad, wife and husband, doctor and patient, attorney and defendants.[4]

4.2 Feature Weight

Based on our previous analysis, we count the number of texts that contain each of the features with respect to different labels in both tasks (Fig 4, 5). We find that our selected features have different distributions among different labels in Task 1. However, these features have quite similar distributions across different labels in Task 2, which makes them not so helpful in distinguishing humor degrees. The reason is that these features are to distinguish between humor and non-humor instead of how humorous it is.

	Colons	Question Marks	Exclamation Marks	Commas	Mean Text Length	Sepcial Groups
Label 0	290	529	214	0	23.57	158
Label 1	**1809**	**1247**	**950**	**60**	**40.86**	**965**

Fig. 4: Feature distribution for Task 1

	Colons	Question Marks	Exclamation Marks	Commas	Mean Text Length	Sepcial Groups
Label 0	2600	1650	1365	213	124.04	1285
Label 1	1898	1328	1069	99	84.43	1068
Label 2	2578	1700	1458	135	107.5	1390

Fig. 5: Feature distribution for Task 2

Therefore, we choose to include custom-selected features only in Task 1.

For each feature, we use one bit to encode it into our embeddings. Based on how distinguishable each feature is, we assign different weights to their corresponding indicator bit. As the colons are the most distinguishable feature, we set its indicator bit to be 1.5 if a text contains a colon. For other punctuations, their weights are all 1. For special groups, the weight we use is 0.5. If a text does not contain that information, the corresponding bit is set to be 0. Besides, we divide the text length by 100 as its weight in embeddings to capture the length information. Then, we concatenate the feature vector with our model's output to be the final embeddings.

4.3 Model Training and Evaluation

We train our model on Colab GPU with the hyperparameters listed here:

- Learning rate: 1e-6
- Weight decay: 1e-4
- Loss function: CrossEntropy Loss

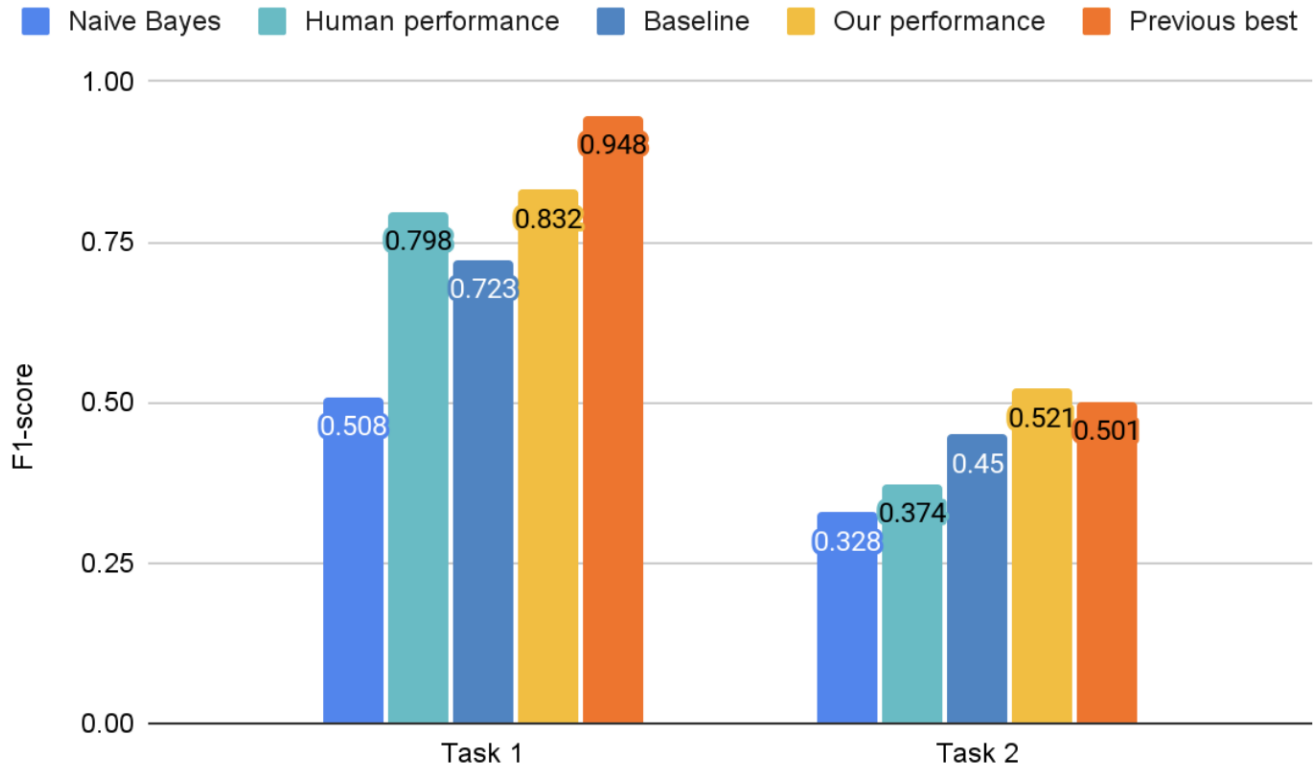

Fig. 6: Results comparison

- Batch Size: 2
- Epoch: 10

It takes about 10 minutes to train one epoch. To evaluate our model, we use Macro F1-Score as our evaluation metric.

Our code can be found in the Github repository here.

5. Results and Analysis

5.1 Baselines

We choose different baseline models for the two tasks. For Task 1, we use the LSTM model suggested by Huanyong Liu [2], which is also the best open-source Chinese model we found as our baseline. For Task 2, we choose the median F1-score in the competition as our baseline. Our baselines are both non-trivial, making the comparison more meaningful.

5.2 Best Performance

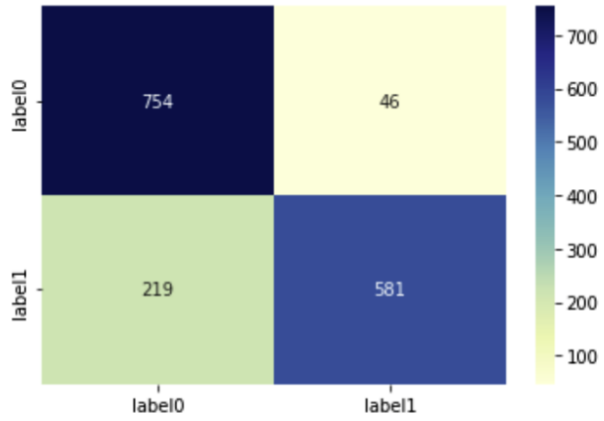

Fig. 7: Confusion matrix for Task 1

Fig. 8: Confusion matrix for Task 2

Training our BERT + LSTM model with custom selected features, we get our best results (Fig 6). On Task 1, we achieve an F1-score of 0.832, and on Task 2, our model achieves an F1-score of 0.521. On both tasks, our model outperforms the Naive Bayes model significantly. Compared to previous work, our model beats the LSTM baseline suggested by Huanyong Liu on binary classification by 0.1. For ternary classification, our model outperformed the previous best model in the competition by 0.02.

We also make confusion matrices to further analyze our results (Fig 7, 8). On Task 1, our model gives a more accurate prediction of non-humorous text. Only 46 non-humorous texts are predicted as humorous while 219 humorous texts are labeled as non-humorous. Therefore, our model is good at distinguishing non-humorous texts but fails to capture all kinds of humor, so it classifies many non-humorous texts as humorous. On Task 2, our model has better performances on weak humor and strong humor but gets confused on somewhat humor, which is reasonable because extremely strong and extremely weak humor is easier to distinguish than somewhat humor in the middle.

5.3 Human Performance

To further evaluate our model, we labeled 600 texts in the test set for each task by hand as the human performance. We also compute our hand-label F1-score. Surprisingly, the hand-label F1-score is only 0.79 for task1 and 0.37 for task2. Therefore, the humor detection task is non-trivial, and even some human beings cannot perform better than the machine learning model.

6. Conclusion and Future Steps

6.1 Conclusion

Overall, the BERT-Base model [6] gives better performance than just using LSTM because BERT computes the embeddings based on the context and can detect humor more precisely. Additionally, we find that feeding BERT encodings into a bidirectional LSTM increases the predictive ability of our model. We believe this is because the LSTM learns to transform the BERT encodings to be more useful to the specific classification task. At the same time, our custom selected features are helpful in Task 1. When we encode them into our model, we increase the F1-score by 0.015. However, these features are not so helpful in Task 2, as all of the texts in Task 2 are humorous. Unfortunately, we do not find any distinguishable features for Task 2, which suggests finding features to tell the degree of humor is much harder than just telling whether it is humorous or not.

Overall, we find that applying NLP methods from one language to another is fruitful but non-trivial. Using ideas from ColBERT and research into humorous English text significantly accelerated our research process, but eventually, we had to develop new ideas to overcome the challenges particular to the Mandarin task.

6.2 Future Steps

As for future work, we first suggest implementing more sophisticated custom features. Our custom features are very simple yet resulted in noticeable performance improvements, so we suspect that further development of custom features will provide even better results.

Secondly, a different loss function might help to improve the performance on Task 2 if it takes the order of labels into account. Specifically, we recommend investigating the class encoding method outlined by Jianling Cheng in his 2008 paper [7]. This method increases the loss for misclassifications based on how far off the misclassification was. For example, classifying a text with label 2 as 0 would result in a higher loss than classifying it as 1.

Third, larger datasets are required for Mandarin humor detection to reach the same precision as English humor detection. The English dataset consisted of a balanced 200k samples, whereas the Mandarin Task 1 dataset had only 16,480 samples and was significantly unbalanced. We believe the main reason the ColBERT architecture performed much better in English than Mandarin is because of the discrepancy in dataset size. Until larger Mandarin datasets are created, sophisticated models will continue to overfit the data.

Lastly, we notice that humor is very subjective and hard to evaluate. Machine learning models are already very good at Task 1, but in Task 2 we see both humans and machines struggle to achieve high performance. The low human performance on Task 2 is particularly striking evidence of the subjective nature of humor. Thus, we believe that traditional performance metrics are not the right tool to evaluate a machine's sense of humor in the long run. Particularly, we think the concept of a sense of humor should be applied to machine learning models as humans find it acceptable for different people to have different senses of humor and will likely extend the same preference to machines.

References

[1] DUTIR-Emotion-Group. 2019. Dutir-emotion-group/CCL2019-chinese-humor-computation: CCL2019，"小牛杯"中文幽默计算任务的数据集及baseline. (October 2019). Retrieved April 1, 2022 from https://github.com/DUTIR-Emotion-Group/CCL2019-Chinese-Humor-Computation

[2] Huanyong Liu. 2018. Chinese Humor Sentiment. (December 2018). Retrieved February 14, 2022 from https://gitcode.net/mirrors/liuhuanyong/ChineseHumorSentiment?utm_source=csdn_github_accelerator

[3] Moradnejad. 2021. Moradnejad/colbert-using-bert-sentence-embedding-for-humor-detection: Novel model and dataset for the task of humor detection. (April 2021). Retrieved April 1, 2022 from https://github.com/Moradnejad/ColBERT-Using-BERT-Sentence-Embedding-for-Humor-Detection

[4] Mihalcea, R., Pulman, S.: Characterizing humour: an exploration of features in humorous seHumorSentiment?utm_source=csdn_github_accelerator Text Processing and Computational Linguistics, pp. 337–347. Springer, Berlin (2007)

[5] T. Rajapakse, “Simple transformers,” Simple Transformers, 2022. [Online]. Available: https://simpletransformers.ai/. [Accessed: 17-Apr-2022].

[6] Sikarwar, Ankur, and Gabriel Kreiman. “On the Efficacy of Co-Attention Transformer Layers in Visual Question Answering.” ArXiv.org, 11 Jan. 2022, https://arxiv.org/abs/2201.03965.

[7] Jianlin Cheng, Zheng Wang and G. Pollastri, "A neural network approach to ordinal regression," 2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence), 2008, pp. 1279-1284, doi: 10.1109/IJCNN.2008.4633963.