Team_KGP at SemEval-2021 Task 7: A Deep Neural System to Detect Humor and Offense with Their Ratings in the Text Data

Anik Mondal
Indian Institute of Technology Kharagpur
pjanik2000@iitkgp.ac.in

Raksha Sharma
Indian Institute of Technology Roorkee
raksha.sharma@cs.iitr.ac.in

Abstract

This paper describes the system submitted to SemEval-2021 Task-7 for all four subtasks. Two subtasks focus on detecting humor and offense from the text (binary classification). On the other hand, the other two subtasks predict humor and offense ratings of the text (linear regression). In this paper, we present two different types of fine-tuning methods by using *linear layers* and *bi-LSTM* layers on top of the pre-trained BERT model. Results show that our system is able to outperform baseline models by a significant margin. We report F1 scores of 0.90 for the first subtask and 0.53 for the third subtask, while we report an RMSE of 0.57 and 0.58 for the second and fourth subtasks, respectively.

1 Introduction

Automatic humor and offense detection have considerable importance in the modern age, especially in chatbots and virtual assistants (Augello et al., 2008). Detection of humor and offense in the text is a highly subjective phenomenon that varies with age, gender, race, socio-economic status, *etc*. Therefore, it is one of the most challenging research fields in Natural Language Processing (Taylor, 2009).

Task-7 of SemEval 2021 (Meaney et al., 2021) is concerned with humor detection (binary classification) primarily. The first subtask is to check if a text is humorous, another subtask follows up: how humorous is the text (rated from 0-5). We also have to predict whether the text is generally offensive (binary classification task). The last subtask predicts how generally offensive a text is (rated from 0-5), regardless of whether it is classed as humorous or offensive overall. The dataset consists entirely of English texts.

The traditional methods deployed earlier for humor detection include Support Vector Machine (SVM) with RBF kernel, Random Forest Classifier, and SGD with Logical Classifier, all of which provide modest results (de Oliveira and Rodrigo, 2015). Recently more state-of-the-art transformers have provided better results (Weller and Seppi, 2019). The system presented in this paper is fine-tuning one of the best and most popular state-of-the-art models, Bidirectional Encoder Representations from Transformers (BERT). We have used pre-trained BERT embeddings to represent the words and used the features of the last layer of the 12-layers BERT Model to *detect* and *rate* both *humor* and *offense*. BERT can model complex interactions between different levels of hierarchical information (Tenney et al., 2019). This task is perhaps the first task that focuses on the humor and offense ratings and combines humor and offense detection.

Our model obtained an F1 score of 0.92 and 0.56 for the first and the third subtasks, respectively, on the test data. We obtained an RMSE of 0.57 and 0.58 for the second and fourth subtasks, respectively, on the test data.

The rest of the paper is organized as follows. Section 2 discusses the related work. Section 3 describes the dataset for the task. Section 4 provides the experimental setup and evaluation metric. Section 5 presents the systems implemented to address the task. Section 6 shows the results and Section 7 concludes the paper.

2 Related Work

We found the following works very relevant for the task to be addressed, *i.e.*, humor and offense detection.

Lexical Syntactic Feature (LSF): This architecture (Chen et al., 2012) was used to detect offensive language in social media. It bridges the gap between message-level and user-level offensive language detection. In particular, this paper incorporates a user's writing style, structure, and specific cyberbullying content as features to predict the user's potentiality to send out offensive content.

Netflix-style collaborative filtering: This method (Gultchin et al., 2019) identifies a mean humor direction, analyses sense-of-humor word embeddings to predict individual differences in word humor. It proposes Netflix-style collaborative filtering to predict humor ratings.

BERT-base and BERT-large: BERT models have been used previously for automatic humor detection and scoring (Mao and Liu, 2019). The pre-trained model is fine-tuned on the available data, producing appreciable results.

Our system lies in close proximity to the last work. We have fine-tuned a pre-trained BERT model as well. The difference lies in the fine-tuning process. Different approaches have been used to building the neural network, varying from linear layer approach to bi-LSTM approach, for both the classification and regression tasks.

3 Dataset

The dataset used in this paper is obtained from Task 7 of SemEval 2021. In this task, the organizers collected labels and ratings from a balanced set of age groups from 18 to 70. The annotators also represented a variety of genders, political stances, and income levels. The training set consists of 8000 texts, and the development and test datasets consist of 1000 texts each. The variation of the number of texts *vs*. word length of texts is shown in Figure 1.

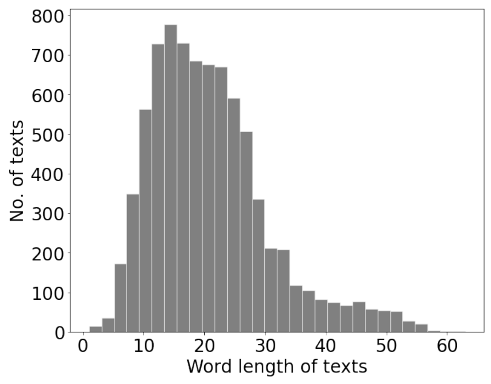

Figure 1: The distribution of lengths of the texts from the training set.

Each text is represented by a unique ID, and each subtask has a separate column for each text. The annotators were asked:

1. Is the intention of this text to be humorous?
2. (If it is intended to be humorous according to the rater) How humorous do you find it? (1-5)

The annotators could also give a rating of 0 to the second question if they do not get it due to the text's structure or content. The label of the first task is *is_humor* and is based on the majority class given by 20 annotators. In case of a tie, the humor label was selected. The humor rating is based on the annotators' average rating, under the label *humor_rating*.

Table 1 gives an insight into the distribution of *is_humor* label. The distribution of *humor_rating* is shown in Figure 2.

is_humor	**No. of texts**
0	4932
1	3068

Table 1: The distribution of texts considered either humorous (1) or not (0).

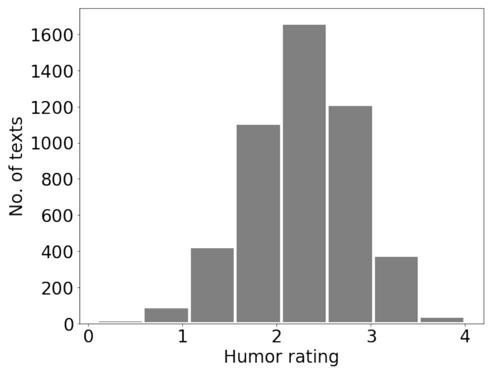

Figure 2: The distribution of *humor_rating* of the texts from the training set.

The annotators were further asked:

3. Is this text generally offensive? (0 or 1)
4. (If the rater considers the text to be generally offensive) How generally offensive is the text? (1-5)

By generally offensive, the organizers mean that the text targets a person or group for merely belonging to a specific group and ask users if they think that a significant number of people would find this offensive. If the variance of a text was higher than the median variance of all texts, the humor of the text was labeled as controversial under the label *humor_controversy*. It is a binary classification task.

In the last question, we consider the ratings 1-5 and also consider a no rating to be 0. This score (average of all the ratings) was calculated regardless of whether the text is classed as humorous or offensive overall, under the label *offense_rating*.

Table 2 gives an insight into the distribution of *humor_controversy* label. The distribution of *offense_rating* is shown in Figure 3.

humor_controversy	No. of texts
0	2467
1	2465

Table 2: The distribution of texts considered either offensive (1) or not (0).

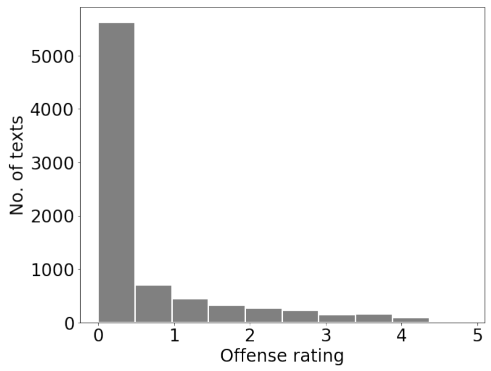

Figure 3: The distribution of *offense_rating* of the texts from the training set.

4 Experimental Setup and Evaluation Metric

Preprocessing: A data pipeline preprocesses the raw data to remove irrelevant features. The stop-words and emojis (if any) are removed from the data, and the output of the pipeline is lower-case stemmed word sequences. The stop-words are selected from the NLTK stop-words list (Sarica and Luo, 2020).

Max Sequence Length: From Figure 1, we observe that there is no sequence having a length greater than 70 words. So, we restrict our *max_seq_length* parameter of the BERT Tokenizer to 70. We pad the sentences to the maximum sequence length.

Adam Optimizer: For optimizing the model, we use *Adam* Optimizer. Adam combines the best properties of the *AdaGrad* and *RMSProp* algorithms to provide an optimization algorithm that can handle sparse gradients on noisy problems and works well on deep neural networks (Zhang, 2018). For subtasks-1 and 3, we used a learning rate of $2e$-5, and for subtasks-2 and 4, a learning rate of $1e$-5 was used.

Experimental Tools: We used Google Colab to run the experiments, which provided us GPU to run our model. The external libraries used are publicly available Transformers (version 3.0.0) from the PyTorch HuggingFace API [1] and Python-based Scikit-learn package.

Evaluation Metric The metric for the classification tasks (subtask 1 and 3) is F1 score.

$$F_1 = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall} \quad (1)$$

For the regression tasks (subtask 2 and 4), the metric is Root Mean Square Error (RMSE).

$$RMSE = \sqrt{\sum_{i=1}^{N} \frac{(Predicted_i - Actual_i)^2}{N}} \quad (2)$$

5 System

The task organizers suggested the following baseline models:

1. Classification task: For the classification tasks, the baseline strategy is a Naive Bayes model with bag of words features.

2. Regression task: The baseline strategy proposed for the regressions tasks is the Support Vector Regression (SVR) model.

BERT has proven promising for many NLP tasks (Devlin et al., 2019). Our system implements fine-tuning strategies on pre-trained BERT architecture. It is a bidirectional transformer pre-trained using a combination of masked language modeling objective and next sentence prediction on a large corpus comprising the Toronto Book Corpus and Wikipedia.

5.1 BERT base model

We have experimented with the BERT-base model from PyTorch HuggingFace API. It is the bare BERT Model (BertModel) transformer outputting raw hidden-states without any specific head on top. The 768 hidden features are extracted from the last layer of the 12-layered BertModel.

[1] `https://huggingface.co/`.

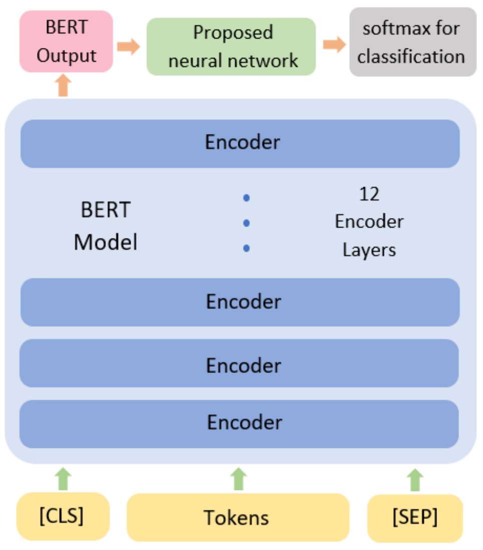

Figure 4: The proposed model architecture for the classification subtasks

5.1.1 Classification Tasks

We have experimented with two independent approaches.

- **Linear approach**: After experimenting with single and double layers, we decided to use three linear layers on top of the features, as it produced better results. The layers are of the dimensions 768×512, 512×256, and 256×2 respectively. Rectified linear unit (ReLU) is used as the nonlinear activation function at the first two layers' output, followed by a dropout regularization of 0.1.

- **Bi-LSTM approach**: We use one bi-LSTM (Bi-directional long short term memory) layer of 512 dimensions at first instead of the initial linear layer. We keep the rest of the architecture the same (including the activation and regularization), altering the dimensions as required.

  The output layer is a softmax layer for the classification job (for both approaches). The reason behind experimenting with the Bi-LSTM model is that it fully considers the context information and can better obtain the text representation of the comments (Xu et al., 2019).

In the BERT training process, the model receives pairs of sentences as input in a specific format. A [CLS] token is inserted at the beginning of the first sentence and a [SEP] token is inserted at the end of each sentence, with the words in the middle being converted to tokens, as shown in Figure 4.

5.1.2 Regression Tasks

Three neural network architectures have been tried on top of the BERT features enumerated below for the regression tasks.

- **3 linear layers**: The same linear layer architecture used for the classification tasks is implemented here; only the dimensions of the last layer are altered appropriately. 3 linear layers of dimensions 768×512, 512×256, and 256×1 respectively are used.

  ReLU is applied as the activation function at the output of the first two layers, followed by a dropout regularization of 0.1. The loss function used for this regression task is the mean squared error (MSE) loss function.

- **One linear layer**: Only one linear layer is used in this approach of the dimension 768×1. MSE loss function is used for the cross-entropy loss.

- **Bi-LSTM layers**: One bi-LSTM layer of 512 dimensions is applied on top of the extracted features from BertModel. On top of that, a linear layer is used to predict the ratings. The same loss function as used in the first two approaches is applied here.

6 Results and Analysis

For each subtask, we trained our data on the entire training set of 8000 texts. We used the development set of 1000 texts as cross-validation data. The results tabulated in this section are reported on the gold test set of 1000 texts.

Each proposed model for all the subtasks was run for 10 epochs with a batch size of 32. Hyperparameters are discussed in Section 5.

Method	Accuracy	F1 score
Linear approach	0.9030	0.9233
bi-LSTM approach	0.8970	0.9177
Naive Bayes	0.8570	0.8840

Table 3: Accuracy and F1 score for the models trained on the humor classification task

The results of the considered BertModel and baseline methods for the *is_humor* subtask-1 are summarized in Table 3 in terms of the F1 score and accuracy. Table 4 shows the results of the *humor_controversy* subtask-3. We observe that the linear approach works better than the bi-LSTM

Method	Accuracy	F1 score
Linear approach	0.5301	0.5628
bi-LSTM approach	0.5203	0.5455
Naive Bayes	0.4374	0.4624

Table 4: Accuracy and F1 score for the models trained on the humor controversy task

approach in both cases. On the other hand, both the proposed models perform better than the baseline model (Naive Bayes with bag of words features).

Method	RMSE
3 linear layers approach	0.5741
Single linear layer approach	0.5847
Bi-LSTM layer approach	0.5694
SVR	0.8609

Table 5: RMSE for the models trained on the humor rating task

Method	RMSE
3 linear layers approach	0.5936
Single linear layer approach	0.6082
Bi-LSTM layer approach	0.5800
SVR	0.6415

Table 6: RMSE for the models trained on the offense rating task

Tables 5 and 6 represent the results of all the three approaches along with the given baseline (SVR) for the regression subtasks in terms of RMSE. Results show that the 3-layered method works better than the single-layer method. We observe that among all three methods, the bi-LSTM approach works the best for both the subtasks. Our proposed methods have significantly lower RMSEs than the baseline SVR model.

In the humor detection subtask, we ranked 48th with the leaders achieving an F1 score of 0.98. We achieved a rank of 23 in the humor rating subtask, leaders getting an RMSE of 0.49. In the humor controversy subtask, the leaders got an F1 score of 0.63, giving us a rank of 15. In the last subtask of offense rating, we achieved a rank of 38, with the leaders getting an RMSE of 0.41.

The section where our model struggles the most is detecting underlying sarcasm in sentences, especially where the context is explored for the first time. For example, in the text: “I asked my North Korean friend how it was there... he said he couldn’t complain.”, our model classifies the text as not humorous however in the gold set the text has been classified as humorous. This is one of several examples where our classifier mislabeled the text. The same problem exists with other subtasks too.

7 Conclusion

In this paper, we describe a system developed to address the SemEavl Task-7. The task has four subtasks, *viz.*, detecting humor, detecting offense, predicting humor rating, and predicting offense rating. Our system is able to perform all four subtasks with varying levels of performance for each task. Our system deploys linear layers and bi-LSTM layers independently to process the features produced by the BERT model. Results show that our system using BERT with Linear layers outperforms the baseline model by a significant margin for the first and the third subtasks. On the other hand, the bi-LSTM layers-based system gives the best performance for the other two fine-grained rating prediction tasks.

References

A. Augello, G. Saccone, S. Gaglio, and G. Pilato. 2008. Humorist bot: Bringing computational humour in a chat-bot system. In *2008 International Conference on Complex, Intelligent and Software Intensive Systems*, pages 703–708.

Y. Chen, Y. Zhou, S. Zhu, and H. Xu. 2012. Detecting offensive language in social media to protect adolescent online safety. In *2012 International Conference on Privacy, Security, Risk and Trust and 2012 International Confernece on Social Computing*, pages 71–80.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Limor Gultchin, Genevieve Patterson, Nancy Baym, Nathaniel Swinger, and Adam Kalai. 2019. Humor in word embeddings: Cockamamie gobbledegook for nincompoops. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2474–2483. PMLR.

Jihang Mao and W. Liu. 2019. A bert-based approach for automatic humor detection and scoring. In *IberLEF@SEPLN*.

J.A. Meaney, Steven R. Wilson, Luis Chiruzzo, Adam Lopez, and Walid Magdy. 2021. Semeval 2021 task 7, hahackathon, detecting and rating humor and offense. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*.

Luke de Oliveira and A. Rodrigo. 2015. Humor detection in yelp reviews.

Serhad Sarica and Jianxi Luo. 2020. Stopwords in technical language processing.

J. M. Taylor. 2009. Computational detection of humor: A dream or a nightmare? the ontological semantics approach. In *2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology*, volume 3, pages 429–432.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. Bert rediscovers the classical nlp pipeline.

Orion Weller and Kevin Seppi. 2019. Humor detection: A transformer gets the last laugh.

G. Xu, Y. Meng, X. Qiu, Z. Yu, and X. Wu. 2019. Sentiment analysis of comment texts based on bilstm. *IEEE Access*, 7:51522–51532.

Z. Zhang. 2018. Improved adam optimizer for deep neural networks. In *2018 IEEE/ACM 26th International Symposium on Quality of Service (IWQoS)*, pages 1–2.