Humour Sense Prediction

Linus Aronsson **Trilokinath Modi** **Akshita Pingle**
linaro@student.chalmers.se modi@student.chalmers.se pingle@student.chalmers.se

Abstract

The project focuses on one aspect of natural language processing research area, specifically humor detection, and rating the quality of humor. The selected topic is one from SemEval 2021 competition and the dataset is available for participants. Neural network architectures such as neural network with CBoW embedding, bidirectional RNN, LSTM, GRU and networks with pretrained word embedding BERT are employed for both the detection and rating tasks. Overall, it is observed that good results are achieved with most of the methods, but the best results are obtained when the network is trained with pretrained word embeddings.

Introduction and Problem Statement

In this report various machine learning architectures will be investigated for the purpose of performing text document classification and regression. The data used is from an ongoing SemEval 2021 competiton (CodaLab 2021). The competition is titled "*HaHackathon: Detecting and Rating Humor and Offense*. The dataset consists of a set of sentences which have been labeled to be either humorous or not. Additionally, each sentence is labeled with a real valued rating of how humorous it is in the range $[0, 5]$. The goal is therefore to classify which of the sentences are humorous (classification) and predict the rating (regression). Each prediction task is done individually, using separate models.

Models

Humor detection is a well studied area within NLP since understanding humour is an important part of various applications such as chatbots. Using BERT for this purpose has been shown to achieve state of the art accuracy by Annamoradnejad (2020). For this reason, the BERT architecture was the main model implemented. In order to display the usefulness of BERT, several other common options for document classification was implemented as well. This includes Recurrent Neural Networks (RNN) and various variations of this (i.e., LSTM and GRU). Each model used is listed below and their functionality is briefly described.

RNN

RNN's is a type of artificial neural network (ANN) developed specifically to handle sequence based data. Since sentiment analysis deals with analysis of word sequences, the use of RNNs for this context has consequently resulted in state of the art results in many cases. RNN's essentially achieve this by allowing the weights in the network to flow backwards, as supposed to strictly forward as in a traditional Feedforword Neural Network (FFNN). This effectively gives the network a memory such that it can consider sequentially dependant features better than a FFNN. Additionally, an extension of RNN's is to make them bidirectional. This means that the input sequence is processed in both directions. This is useful in a NLP context since the order of words can be analyzed more precisely this way. All RNN variations implemented in the project are bidirectional.

LSTM

A common issue with ANN's is the *vanishing gradient problem* (Hochreiter 1998). This issue essentially means that the more layers that are added to the network, the more difficult it is for the ANN to learn. RNN's especially suffer from this since they are naturally very deep networks because each word in the sequence will correspond to a layer. LSTM networks is a variation of RNN's that were implemented to fix this problem such that it can work with longer sequences.

GRU

GRU's were similarly implemented for the purpose of avoiding the vanishing gradient problem (Cho et al. 2014). It works roughly like LSTM's except that it contains less parameters and is therefore generally more computationally efficient. Whether GRU's or LSTM's are superior is not clear, and depends on the use case. For this reason both methods were implemented and tried.

BERT

BERT is a transformer based technique developed by Google (Devlin et al. 2018). BERT utilizes a very important machine learning concept, namely transfer learning. It does this by training context-dependant word embeddings on a very large general purpose corpus. Context-dependant means that it may produce different embeddings for the same word depending on the context in which it is used. The embeddings can then be optionally fine-tuned for the particular domain specific use case and subsequently fed as input into some additional model. In this project, two additional models were utilized. The first model passes the BERT embeddings into a linear layer while the second passes them into a LSTM network with attention.

Data Preparation

A well labeled data was acquired from CodaLab, SemEval 2021 competition. The labeled data contained 8000 documents. From this 8000 documents, 1000 documents were randomly separated but with stratification as explained in next section. It is also observed that the longest document has 300 tokens and the average number of tokens in each document is 117 after NLTK tokenization.

Training and Test data

The training and test set is split accordingly to maintain this ratio of 0s and 1s when it comes to the classification task. By doing this there will be enough data for each instance while predicting. This is similar to stratified train-test split. The humors classified as "non-humor" doesn't have any humor rating. These records are, hence, removed in the train and test data while dealing with regression task. The training data is slightly imbalanced for both classification and regression tasks but it is good enough to perform prediction efficiently. Additionally, tokenization is performed using natural language toolkit NLTK to split the sentences and punctuations into words. All tokens are then converted to lower cased.

Preprocessing

Preprocessing is done on the data to make it appropriate enough for classification. Since the models cannot handle text directly, all the sentences needs to be converted into numerical form. Continuous Bag of Words(CBOW) approach is used for this purpose. CBoW computes a sum or mean of word embeddings over a document. Word embeddings here represents the similarity of the surrounding words with respect to target word. After using this approach, it converts the set of words to set of vectors. The output of this approach is a large sparse matrix of numbers which can now be used by RNN. However, this preprocessing step doesn't apply to any model having BERT. BERT itself is a machine learning technique that provides embedding for the words based on their context.

Experiments

Preliminary

The dataset contains 8000 samples of documents. One can expect that there will be little improvement in the accuracy if the entire dataset consists of unique words. However, this is rare for problems where day to day language is used which applies to the dataset in this project. Hence, one can expect the dataset to have common subjects (i.e., common words) where the humor occurs. Because of this, as a preliminary experiment, topic modelling using LDA and collapsed Gibbs sampling approach on the text was done to identify if there exists a set of words or subjects where humor occurs more often. This part of the experiment can be considered as unsupervised learning.

A few words on early stopping and the number of iterations used for training are mentioned below.

- **Early Stopping**. Overfitting can be mitigated to some extent using dropout probability, however, the model can still overfit to training data. To avoid overfitting of training data, an early stopping criteria is imposed in all models with an exception of BERT related models. Specifically, if the validation test accuracy reduces for 4 consecutive times during training then the training stops. However, for BERT related models, the training stops as soon as validation accuracy decreases but only after 3 epochs.
- **Iterations**. All the experiment mentioned is repeated 5 times and average over the results is taken to achieve a stable estimate of model performance. The BERT related models are repeated only twice.

Model Configurations

The configuration of each model is described in the list below. The model configurations is same for both classification and regression tasks.

- **CBoW**. A continuous bag of words approach is used to create word embeddings. The average over all tokens in the document is taken to form a embedded vector of size 32. This is fed as input into a simple neural network with no hidden layer.
- **Bidirectional RNN**. A bidirectional recurrent layer is connected to an output layer maintaining a dropout probability of 0.20 to avoid overfitting of the training data. The recurrent layer itself contains an input, hidden and output layer. The input of RNN is the word embeddings generated from the dataset. The output layer spits out a concatenated tensor of first and last token. This concatenated tensor is then connected to the final output layer.
- **Bidirectional LSTM**. A bidirectional LSTM is connected in an identical fashion as the RNN. The only difference is the working nature of these methods, i.e. incorporation of forget, input and output gates in case of LSTM's.
- **Bidirectional GRU**. A bidirectional GRU is connected in an identical fashion as RNN's. The only difference is the working nature of these methods, i.e. incorporation of reset and update gates in case of GRU's.
- **BERT**. The pretrained BERT word embeddings are connected to a final output layer. The word embeddings are then adjusted by training the network using the given dataset.
- **BERT + LSTM**. This model feeds the pretrained BERT embeddings for each word into a bidirectional LSTM. The output of the LSTM is done using attention (i.e., an average over all outputs of the LSTM network).

Classification

The first task is to identify if a given document is humorous or not. This supervised learning task is performed by first training a network. The train-validation split is done by a holdout method with ratio $(80 : 20)$. Since the dataset is slightly imbalanced, both accuracy and F1-score are used as the evaluation metrics. Cross entropy loss is used for all classification models.

Topic 3	white	people	black	mental	red
Topic 1	women	said	health	girls	people
Topic 4	wife	people	love	good	night
Topic 2	wife	man	said	like	tell
Topic 0	people	life	said	wife	gay

Table 1: High-frequency words in topics

Regression

The second task is to predict humor rating. One can think of this as a regression task. The final output layer contains one neuron spitting out the humor rating. The methods used for training the networks is similar to the methods used for the classification task. However, the loss function is considered for this task is "mean squared error". A final output layer of 1 neuron is added to the network architecture.

MSE and residual plots are diagnosed to comment on model performances.

Evaluation

Preliminary

Humors are generally short texts and quite abstract in nature. The prior distribution hyper parameters i.e. Dirichlet hyperparameter β and topic hyperparameter α is set as $\{\alpha, \beta\} = \{0.01, 0.01\}$. Topic modelling for 5 topics with these hyperparameters gave some insights to dataset. For example, one topic suggested that some humor documents are related to color due to presence of words like "white", "black", "red", "people". These words infer the documents are either related to human race or the color is used with rare words like "mole", "belt" etc. And many topics suggested that often humor documents are "conversational". Conversational, implying, the tokens in these topics contains words like "tell", "said", "says", "said" etc. The topics also suggested that the conversational humor is often related to couples because of the presence of words "wife", "girlfriend", "man", "love" etc. Some words picked from the topics are shown in Table 1. The rare words were analysed under same topics which gave insight that some humor are related to places, activities, advertisements and funerals, due to presence of words like "tennessee", "after-work", "extra", "curricula", "classifieds", "advertisements", "weddings", "funerals" etc.

Classification

CBoW First a CBoW word embedding is averaged out over document and is passed as input to a simple neural network. On average of 5 iterations, the training stopped at 20-25 epochs. Accuracy and f1 score were recorded and finally the network is tested against test data.

RNN, LSTM and GRU

Effect of dropout probability and embedding dimension: The dropout probability is picked from the set $\{0.01, 0.20\}$, whereas the embedding dimension is picked from the set $\{32, 64, 128\}$. All six combinations are tested for network made using RNN, LSTM and GRU. The results are presented in Table 2 and Table 3.

P(dropout)		**0.01**	**0.20**
RNN	**32**	0.829	0.827
	64	0.817	0.827
	128	**0.830**	0.811
LSTM	**32**	0.858	**0.870**
	64	0.863	0.864
	128	0.862	0.870
GRU	**32**	0.861	0.868
	64	0.867	0.864
	128	0.866	**0.868**

Table 2: Effect of dropout probability and embedding dimension on validation accuracies

P(dropout)		**0.01**	**0.20**
RNN	**32**	0.875	0.866
	64	0.873	**0.886**
	128	0.860	0.802
LSTM	**32**	0.888	**0.909**
	64	0.898	0.891
	128	0.878	0.882
GRU	**32**	0.890	0.880
	64	0.909	**0.913**
	128	0.842	0.909

Table 3: Effect of dropout probability and embedding dimension on validation f1 scores

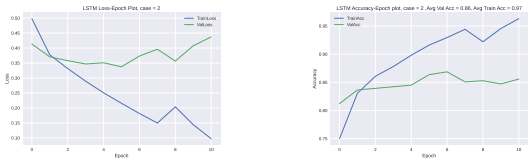
It can be inferred from Table 2 and Table 3 that dropout probability 0.20 worked slightly better than dropout probability 0.01, hence, dropout probability of 0.20 will be kept fixed in all the remaining sections. It can also be noted that there is no significant difference in performance of LSTM and GRU, however, both LSTM and GRU worked much better than RNN's. Moreover, there is no notable difference in performance with changing embedding dimension.

Effect of neural architecture: Using the results obtained from effect of dropout probability and embedding dimension, we can ruleout RNN for further assessments. Effect of neural architecture deals with cases involving, adding a hidden layer with 2*(128 = gru size) neurons before final output layer, adding an additional GRU layer. The results are presented in table 4. It appears as adding these additional layers have no effect on the network. One may think that the network doesn't need these additional activation opportunities.

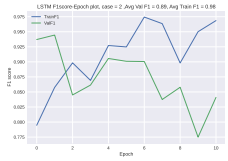
	Architecture	**Hidden layer**	**2 RNN layers**
GRU	**Accuracy**	0.859	**0.864**
	F1 Score	0.862	**0.890**

Table 4: Effect of network architecture on GRU

Among all the experiments above, a randomly selected experiment images is shown in Figure 1.

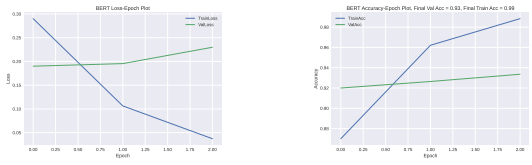


(a) (b)

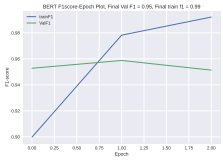


(c)

Figure 1: (a) is the LSTM loss-epoch plot, (b) is the LSTM accuracy-epoch plot and (c) is the F1-score epoch plot



(a) (b)



(c)

Figure 2: (a) is the BERT loss-epoch plot, (b) is the BERT accuracy-epoch plot and (c) is the F1-score epoch plot

BERT

Effect of pretrained BERT embeddings: Two pre-trained embeddings, specifically, "Distilbert-base-uncased" and "xlnet-base-cased" are compared for BERT model performances. Distilbert is a pretrained word embedding that is trained on distilled bert base uncased using 65M parameters. Whereas, XLNet model is trained on XLNET english model using 110M parameters. The XLNET word embedding was chosen following the research paper (Yang et al. 2019). The results are shown in Table 5.

Both the pre-trained models work extremely well, producing high accuracy and f1 score.

	BERT	
Embedding	**Accuracy**	**F1 Score**
Distil-base-uncased	**0.933**	0.951
XLNET-base-cased	0.932	**0.956**

Table 5: Effect of pre-trained word embedding

The Distilbert-base-uncased iterations are shown in Figure 2.

Regression

The distribution of humor rating for regression task is shown in Figure 3.

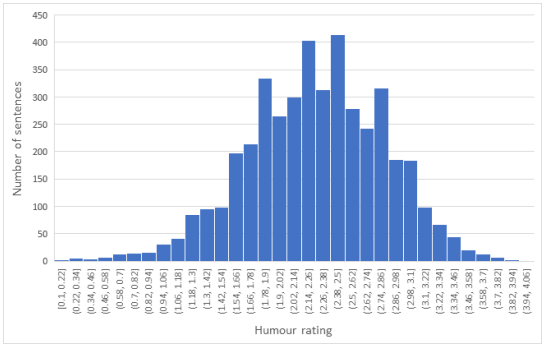
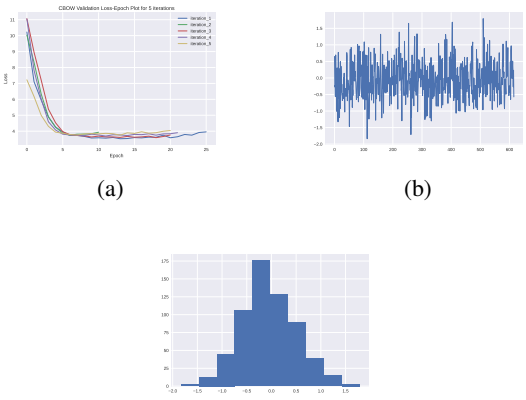


Figure 3: Distribution of humour rating in the training data for regression task

Performance of CBOW: The performance of CBOW approach can be analysed using the figure 4. The figure 4(a) implies the training helps in reducing the validation loss from a range of 11 to a range of 4. No trend is observed in case of residual plot, implies a good fit. However, the range of residual values suggest there are some potential outliers which may result in poor overall performance.

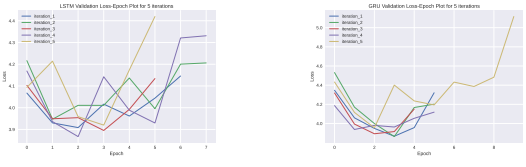


(a) (b)

(c)

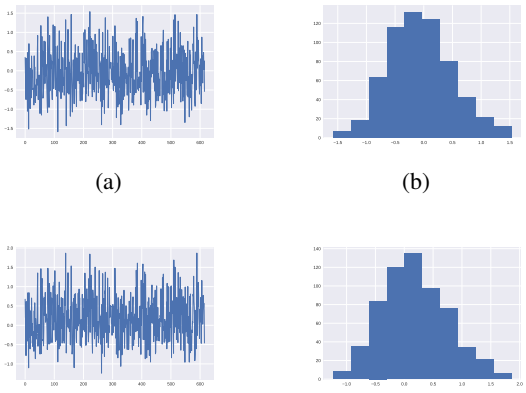
Figure 4: (a) is the validation loss plot for 5 iterations of CBOW, (b) is the residual values when tested against test data (c) histogram of the residual values

Performance of LSTM and GRU: While experimenting, it was observed that for LSTM and GRU models, the validation loss increases rapidly after 3-4 epochs. Hence, for this case, the early stopping criteria is modified from 4 consecutive losses to 2 consecutive losses. The validation loss-epoch plot is compared in figure 5 and residual plots are compared in figure 6.



(a) (b)

Figure 5: (a),(b) is the validation loss plot for 5 iterations of LSTM, GRU respectively



(a) (b)

(c) (d)

Figure 6: (a),(c) is the residual plot for test data in case of LSTM, GRU respectively, whereas, (b),(d) are their respective histograms.

It is really difficult to compare which among LSTM and GRU is better. Both the methods requires roughly equal number of epochs. A comparison in histogram of figure 6 suggests that the frequency of observations where the predicted value differs from true value is not significantly different in either case.

Performance of BERT: While experimenting BERT, it was observed that the validation loss decreases up to epoch 3, post which it fluctuates within a range. However, the training loss at epoch 3 is significantly higher than future epochs. Hence, BERT performance is tested on both the cases.

Case - 1, With early stopping, the training stops as soon as the validation loss increases. **Case - 2**, training stops after 10 epochs, which is also when the training loss is decreased significantly. Figure 7 represents training loss and validation loss for this case. A comparison between residuals is presented in figure 8.

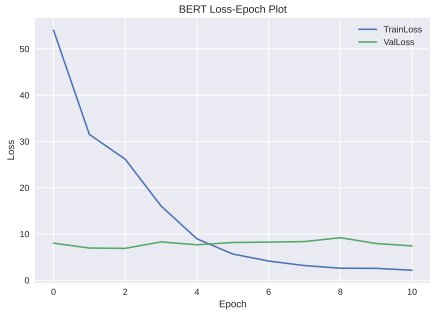
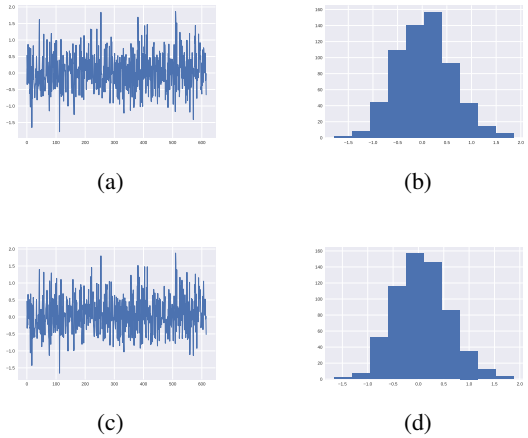


Figure 7: Training and validation loss for BERT model



(a) (b) (c) (d)

Figure 8: (a),(c) is the residual plot for test data in case of BERT two cases, whereas, (b),(d) are their respective histograms.

The plots presented in figure 8, doesn't pose any significant difference, just like the case of GRU and LSTM. However, the MSE value for no early stopping, 0.280, is slightly lower than MSE for the model when training stopped early, 0.313. The difference is not big enough to conclude a better model.

Results

Classification

Under evaluation section , the accuracy and f1 score discussed were all validation scores. Here, in table 6 the evaluation is done on test data. Referring to table 6, we can clearly state that BERT related models gave best performance for classification task, nevertheless, it is a consequence of pretrained embedding on humongous data. The other models were quite competitive with accuracies and F1 scores limiting towards 90%. Adding a recurrent network like LSTM over BERT model improves model performance a bit, but the complexity of model increases far more.

	CBOW	LSTM	GRU	BERT	BERT+LSTM
Acc.	0.869	0.864	0.875	0.932	0.936
$\boldsymbol{F}_1$	0.892	0.890	0.896	0.944	0.947

Table 6: Classification results

Regression

Under evaluation section the discussion of model performances has revolved around the model's ability to perform in test data i.e. histograms of residuals. A summary of those results is presented in table 7. AWR implies accuracy within range, a metric established in this project to quantify the performance of various network architectures. AWR denotes fraction of observation where the difference between predicted humor rating and true humor rating is less than 0.5. AWR is used in table 7. Referring to table 7, it is not obvious to pick a particular model above other. Both the MSE and AWR are not significantly different than others.

	CBOW	LSTM	GRU	BERT
MSE	0.307	0.326	0.348	0.280
AWR	0.644	0.610	0.626	0.657

Table 7: Regression results

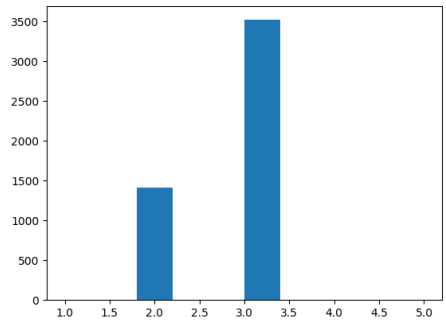


Figure 9: Rating divided in class

Discussion

The preliminary experiments like topic modelling didn't give an extended insight to dataset, but some inferences were made. Humor documents can be quite abstract which make it difficult to predict a short and good set of topics.

For the classification task, model superiority can be easily judged based on the evaluation metrics, however, regression task is far more challenging. BERT was superior for classification task, but CBoW performed equally good when compared to LSTM and GRU. To convince this, one can argue that the average document length is only 117 tokens including punctuation. And such short documents doesn't really benefit recurrent models as they are designed to remember far off valuable words. A good performance in CBoW model also indicates our initial claim under preliminary evaluation of topic modelling that some humor documents can be kept under a single topic sharing similar context.

For regression task, it can be a debatable argument to state that the CBoW worked as good as BERT, but can be supported by the results obtained. It seems both CBoW and BERT slightly edged over RNN based models(GRU and LSTM). But overall, the model performance for regression task using any model is questionable. It is partly due to the fact that rating a humor is itself debatable. It is often the case that ratings have a wider interval, and humor ratings may potentially have even high interval. Moreover, if we class humor rating into 5 levels then the distribution will be highly skewed, refer figure 9. The class with humor rating lying in interval $[1.5, 3.5]$ will have high number of instances as compared to other levels. This makes it even difficult for the algorithm, as a noisy model can always predict rating as 2 and still achieve a good accuracy.

Conclusion

In this project, we have successfully trained and tested various state-of-the-art NLP techniques. From these models, it is expected that a trained network should be able to correctly classify if the document is humorous or not. There is not great certainty if the model's predicted rating will match true rating with a 95% or 99% confidence interval.

The availability of data is increasing and so is the model abilities. In future, one can try character based embedding or train in even larger corpus of humorous data or a lot more possibilities to explore.

References

[Annamoradnejad 2020] Annamoradnejad, I. 2020. Colbert: Using bert sentence embedding for humor detection.

[Cho et al. 2014] Cho, K.; van Merrienboer, B.; Gülçehre, Ç.; Bougares, F.; Schwenk, H.; and Bengio, Y. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *CoRR* abs/1406.1078.

[CodaLab 2021] CodaLab. 2021. Hahackathon: Detecting and rating humor and offense. Accessed: 2021-01-10.

[Devlin et al. 2018] Devlin, J.; Chang, M.; Lee, K.; and Toutanova, K. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR* abs/1810.04805.

[Hochreiter 1998] Hochreiter, S. 1998. The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 6(02):107–116.

[Yang et al. 2019] Yang, Z.; Dai, Z.; Yang, Y.; Carbonell, J.; Salakhutdinov, R.; and Le, Q. 2019. Xlnet: Generalized autoregressive pretraining for language understanding.