Consumer Complaints: Text Classification

Name:	**Akanksha Singh**
Registration No./Roll No.:	19022
Institute/University Name:	IISER Bhopal
Program/Stream:	DSE
Problem Release date:	August 15, 2022
Date of Submission:	November 30, 2022

1 Introduction

Businesses can better understand consumer concerns and experiences by investing in customer assistance. Understanding your customers' complaints and treating them as extremely useful feedback to incorporate into your customer service plan to enhance the brand experience is the best method to guarantee your company's growth.

The data used for this project is collected from the Consumer Financial Protection Bureau (CFPB), a federal U.S. agency that acts as a mediator when disputes arise between financial institutions and consumers. The data was downloaded directly from the CFPB website for training and testing the model. It included one year's worth of data (March 2020 to March 2021). Retail banking, credit cards, debt collection, mortgages and loans, and credit reporting are the five categories for the complaints given. The training dataset categories are visualized in the bar graph (Figure 1). Word cloud shows the complaints for credit reporting in Figure 2.

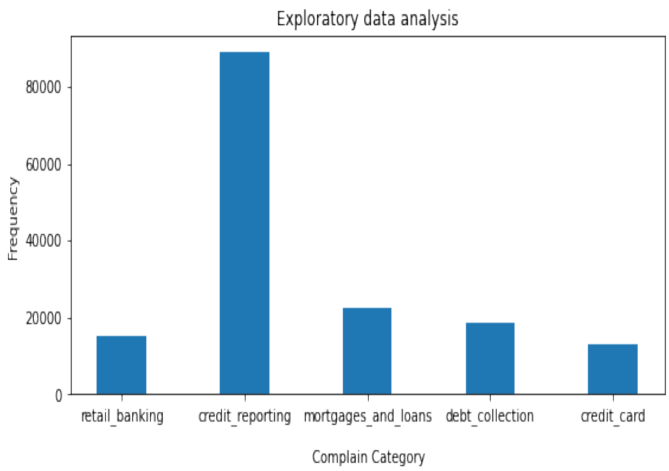

Figure 1: Overview of categories in training dataset in bar graph

2 Methods

2.1 Data Cleaning

The training data used for this project has 158360 rows and two features, 'Complaint' and 'Category.' The 'Complaint' contains all the complaints made by customers in the English language, whereas the 'Category' represents predefined classes like retail banking, debt collection, etc. Since the 'Complaint'

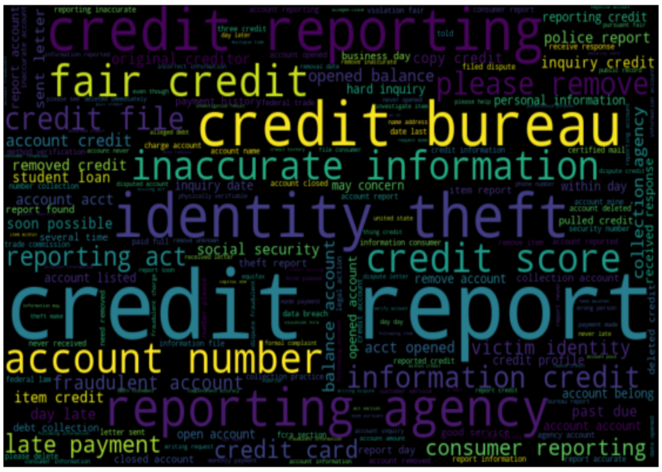

Figure 2: Word cloud for credit reporting

section in training contains ten null values; therefore, the data is cleaned by eliminating those rows. It would not greatly impact the model training since the dataset is already large.

2.2 Data Pre-processing

(i) **Conversion to lowercase:** The unpunctuated data is then passed as an input to lowercase, which aided in pre-processing and later parsing stages of the NLP application. Max words used for training is 10000.

(ii) **Tokenization:** We used sentence tokenization to break our data.

(iii) **Pad sequencing:** We padded each data point to 100 sequence lengths.

(iv) **Splitting data:** We have split the data into the training set, validation set, and test set in a ratio of 16:4:5.

2.3 Classification Models

We employed four classification models: Stacked RNN, Stacked LSTM, Feedforward, and Transformer. For all the models, the Embedding layer gives out the input of 10000 words with a dimensionality of 100. The further architecture is briefly explained below:

(i) **Feedforward:** We used Flatten layer to decrease the dimensionality, then Dense layers were added as fully connected layers, reducing the output until we got 5 logits passed through Softmax for predicting the output class. The activation function used is ReLU, epochs are set as 5, and batch size is 32. We used Adam Optimizer, Cross Entropy Categorical Loss (since we multi-class nominal labels classification). Architecture can be visualized in Figure 3.

(ii) **Stacked RNN:** We used SimpleRNN layers of the same size, which give the full sequence as output to the next layers, to the last SimpleRNN layer, which gives the last output sequence to the Dense layer that gives 5 logits passed through Softmax for predicting the output class. The default activation function Tanh is used, epochs are set as 10, and batch size is 32. We used Adam Optimizer, Cross Entropy Categorical Loss. Architecture can be visualized in Figure 4.

(iii) **Stacked LSTM:** We used Bidirectional layers with full sequence output as a return, then two Dense layers were added as fully connected layers, reducing the output until we get 5 logits passed through Softmax for predicting the output class. The activation function used for one of the Dense layers is ReLU, epochs are set as 10, and batch size is 32. We used Adam Optimizer, Cross Entropy Categorical Loss (since we multi-class nominal labels classification). Architecture can be visualized in Figure 5.

(iv) **Transformer:** We defined a Token and Position Embedding layer function and a transformer block. We use an embedding layer, one transformer block, a pooling layer, and dense layers that give 5 logits passed through Softmax for predicting the output class. The activation function used for one of the Dense layers is ReLU, epochs are set as 10, and batch size is 32. We used Adam Optimizer, Cross Entropy Categorical Loss. Architecture can be visualized in Figure 6.

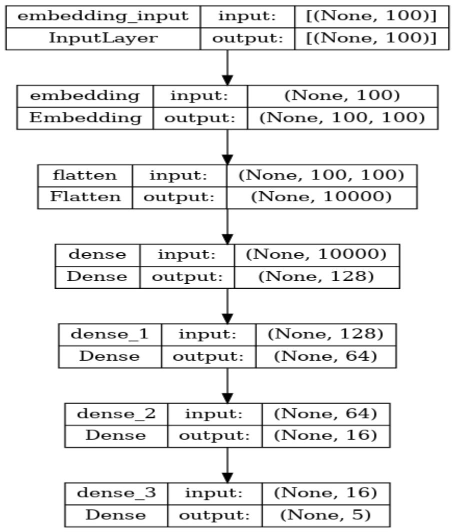

Figure 3: Feedforward architecture

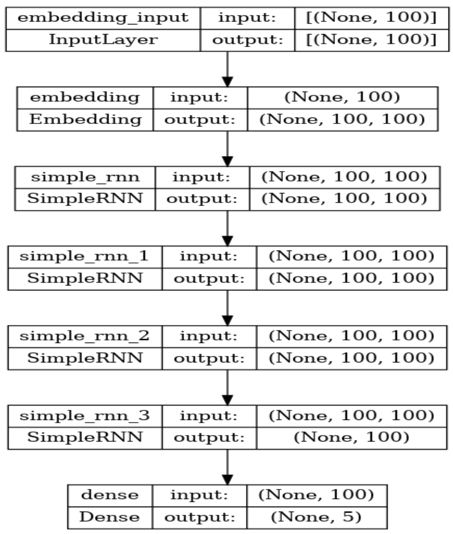

Figure 4: Stacked RNN architecture

3 Evaluation Criteria & Analysis of Results

For evaluation, we considered four criteria accuracy, precision, recall, and F1 score. The results are mentioned in table 1. Using accuracy as the main criterion, we chose Stacked bidirectional LSTM, which gave the best results with an accuracy of 86.5%, followed by a transformer with 85.5%, feedforward with 84.7%, and stacked RNN with 63.2%. The results are mentioned in table 2.

We used bidirectional LSTM layers since our data is non-time series data. The transformer with one block worked well since the number of classes is less, i.e., five. Feedforward worked better than RNN since there is no feedback from the output in the architecture. On the other hand, stacked RNN gave the worst results since our sequence length is large, i.e., hundred, and it might face gradient vanishing.

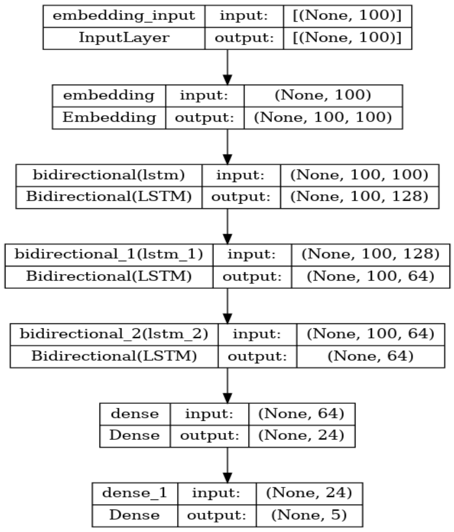

Figure 5: Stacked LSTM architecture

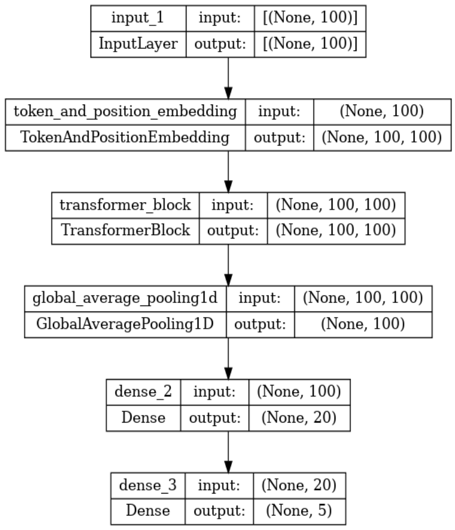

Figure 6: Transformer architecture

4 Discussions and Conclusion

Although the accuracies obtained for both deep learning models and statistical models do not have a significant gap, the training time for deep learning models is much less when compared with the latter. We achieved good accuracy even for five epochs. We could have experimented with different hyperparameters and arguments for better results. Drawing comparisons between vanilla and fine-tuned models would have rendered better insights about the dataset and application of deep learning models for such problem statements.

Since 'Credit Reporting' has 56% weightage in the training data; therefore our model will be more biased towards it. To understand the true value of complaints, we must change how we think about them. Given that there are some significant benefits to customer complaints, we should support them. They significantly affect the company. So, by identifying the advantages of complaints, we can enhance your brand's reputation and boost team output.

This project allowed us to learn Linux, tmux, and remote server access. It surely wasn't easy, but we are happy that we learned new things on the way.

Metrics	Feedforward	Stacked RNN	Stacked LSTM	Transformer
Accuracy	0.85	0.63	0.86	0.85
F1 score	0.79	0.34	0.81	0.80
Precision	0.78	0.39	0.82	0.82
Recall	0.79	0.36	0.81	0.79

Table 1: Comparison of model performance

Models	Training set		Validation set		Test set	
	Loss	Accuracy	Loss	Accuracy	Loss	Accuracy
Feedforward	0.062	0.978	0.891	0.847	0.871	0.847
Stacked RNN	0.915	0.613	0.912	0.625	0.899	0.632
Stacked LSTM	0.132	0.958	0.534	0.863	0.534	0.865
Transformer	0.16	0.941	0.61	0.855	0.599	0.855

Table 2: Performance of different set of data

5 Contribution

I contributed to the project by fine-tuning the models and organizing the repository of the project.

Directory path - `\DATA1\NLP\akanksha_19022\akanksha_19022(venv)`