Urdu Text Sentiment Analysis Using Recurrent Neural Networks

Mohammad Ahmad Safvi
School of Electrical Engineering and Computer Science
National University of Sciences and Technology
Islamabad, Pakistan
msafvi.bscs21seecs@seecs.edu.pk

Syed Arsal Rahman
School of Electrical Engineering and Computer Science
National University of Sciences and Technology
Islamabad, Pakistan
srahman.bscs21seecs@seecs.edu.pk

***Abstract*—Sentiment analysis aims to identify the sentiment expressed in textual data. In this paper, we focus on Urdu text sentiment analysis using recurrent neural networks (RNNs). Urdu is a widely spoken language in South Asia, and analyzing sentiment in Urdu text can be valuable for various applications such as validating public opinion on certain topics by simply scanning through texts**

***Index Terms*—component, formatting, style, styling, insert**

I. Problem Statement

The goal is to develop a model capable of accurately predicting sentiment (positive or negative) from Urdu text. This involves:

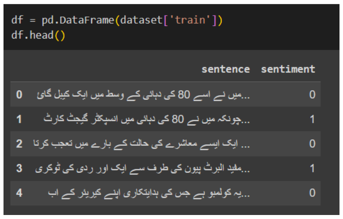
- Acquiring a dataset of Urdu text with sentiment labels.
- Implementing an RNN-based model for sentiment analysis.
- Evaluating the model's performance on Urdu text sentiment classification.

II. Dataset

A. IMDB Urdu Dataset

To increase the availability of sentiment analysis dataset for a low recourse language like Urdu, we opted to use the publicly available Urdu IMDB Dataset from HuggingFace [1]. This dataset takes the popular IMDB Dataset and translates it using google translator. This is a binary classification dataset having two classes as positive and negative. The reason behind using this dataset is high polarity for each class. It contains 50k samples equally divided in two classes.

We Divided the data into three parts of 80%, 10% and 10% for training, testing and validation respectively



```
df = pd.DataFrame(dataset['train'])
df.head()
```

	sentence	sentiment
0	میں نے اسے 80 کی دہائی کے وسط میں ایک کیبل گائ...	0
1	چونکہ میں نے 80 کی دہائی میں انسپکٹر گیجٹ کارٹ...	1
2	ایک ایسے معاشرے کی حالت کے بارے میں تعجب کرتا ...	0
3	مفید البرٹ پیون کی طرف سے ایک اور ردی کی ٹوکری...	1
4	یہ کولمبو ہے جس کی ہدایتکاری اپنے کیریئر کے اب...	0

Fig. 1. A preview of our data stored as a dataframe

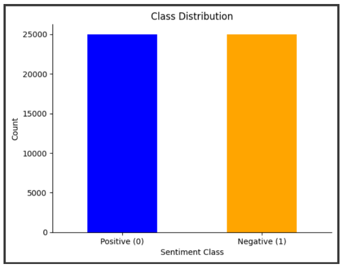


Fig. 2. Plot showing the equal distribution of sentiments across reviews

B. Pre-processing

We utilize regular expressions to remove all non-urdu characters and keep only words along with spaces. We then tokenize the words using nltk's tokenize function to extract words from the sentences and perform further preprocessing in the form of discarding words that are 1 letter long and only maintaining stemmed words (using the Porter stemmer) in our text.

We also exclude words that exist in a list of urdu stopwords from an urdu_stopwords csv file [3] we acquired. Finally we reverse and join the tokenized words back.

We use this joined text to find the max length of a sentence which is to be used for padding purposes, and then proceed to tokenize and sequence the text, making it finally ready for input into the model.

III. Model Architecture

Our model consists of an embedding layer, with our vocabulary having a size of 111221 unique words, and our embeddings having a dimensionality of 600. Following is our recurrent module, the Gated Recurrent Unit (GRU) layer of 256 units. We chose to also add a dropout layer with a 50% chance, along with 2 fully connected dense layers with the prior being of 12 neuron with activation ReLU and the latter being the output layer of 1 neuron with sigmoid as its activation.

We use Binary Cross Entropy as our loss function, alongside the Adam optimizer and ran our model in training for 2 epochs.

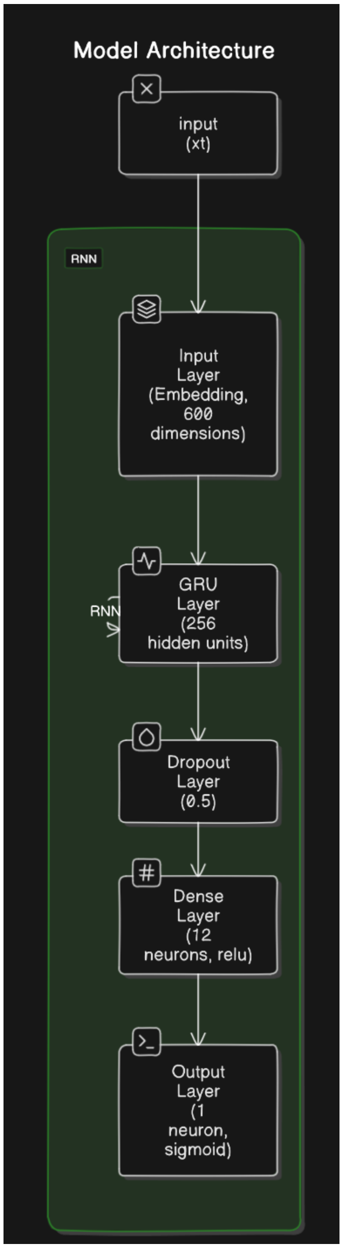


Fig. 3. Model Architecture

A. Experiments and Analysis

We firstly tried a simple LSTM layer with 256 Units that at 3 epochs reached a validation accuracy of 85% and training accuracy of 89%. Another configuration we tried was a Bidirectional GRU layer the achieved similar results to the one we achieved using a normal GRU model but at the cost of more computational power and time,

Additionally we also tried using multiple GRU layers and multiple Bidirectional GRU layers of unit 126 that achieved accuracy's of 85% and 83% respectively

Moreover we tried adding a 1D convolution layer with kernel size of 5 between the embedding and a GRU layer that achieved no better accuracy then what we achieved using a just the GRU layer. Lastly we also tried a multi modal approach where we extracted features using a an embedding layer followed by a single GRU layer of 256 units and another with 1D convolution layer of kernel size 5 after which both the features were concatenated and feed into a classifier with Dropout with p=0.5 followed by two dense layers of size 12 and 1 respectively with relU and sigmoid as there activation function which achieved no better results then what we had achieved previously

We also experimented with different epochs of 2 , 3 and 4, for our chosen model we trained it on 2 epochs after training it on 4 epochs which resulted in over fitting in the model and even a drop in validation accuracy with training being 94% and validation dropping to 85%, the right balance was found at 2 epochs with a validation accuracy of 87% and training accuracy of 89%

IV. Results

Our final training results are tabulated as following:

TABLE I
Results

	Epoch 1	**Epoch 2**
Training Loss:	0.4930	0.2710
Training Accuracy:	0.7566	0.8925
Validation Loss:	0.3886	0.2992
Validation Accuracy:	0.8304	0.8754

And the both the training and validation losses and accuracies plotted:

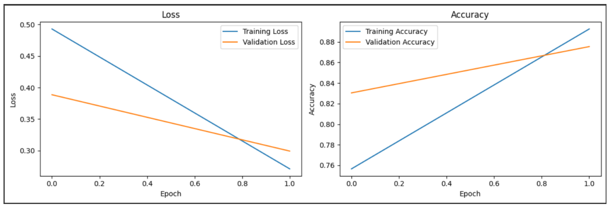


Fig. 4. Loss and Accuracy Plots

Along with these training results, our model also acheived a test loss of 0.2894 and a test accuracy of 88.08%. However, this is no better than what has been documented in previous reports as mentioned ahead.

V. Conclusion

Our model performed comparably to other available RNN models such as the GRU and BiLSTM models in geekforgeeks implementation [2] on the English IMDB dataset, which acheived accuracies of 88.14 and 87.48% respectively.

Inference was performed and our model was successfully able to classify the sentiment on two urdu statements as positive and negative accurately.

.

VI. Appendix

Our google colab file where the source code can be found here

References

[1] IMDB Urdu Reviews https://huggingface.co/datasets/imdb_urdu_reviews

[2] Geeks for Geeks Sentiment analysis with an recurrent neural network https://www.geeksforgeeks.org/sentiment-analysis-with-an-recurrent-neural-networks-rnn/

[3] Urdu Stopwords https://www.kaggle.com/datasets/itsnobita/urdu-stopwords