Sentiment Analysis Model Report

This report will explain the preprocessing steps, and design choices for the neural network, and provide training and testing accuracies for the sentiment analysis model. Also, comments on the output and potential improvements are included.

1. Preprocessing Steps

The preprocessing steps are essential for preparing text data before training any machine learning model. For the sentiment analysis task, the following preprocessing steps were applied:

a. Tokenization: Reviews were tokenized into individual words using the `word_tokenize` function from the `nltk.tokenize` module. Tokenization breaks down the text into individual tokens, which will be further processed.

b. Stopword Removal: A list of stopwords from the `nltk.corpus` module was used to remove common English stopwords. Stopwords are common words like "the," "is," "and," etc., which do not add much meaning to the text and can be safely removed.

c. Lemmatization: Lemmatization was used to reduce words to their base or root form. The `WordNetLemmatizer` from the `nltk.stem` module was employed for this purpose. Lemmatization ensures that related words are reduced to the same base form, improving model generalization.

d. TF-IDF Vectorization: The preprocessed text data was converted into a numerical format suitable for machine learning using the `TfidfVectorizer` from the `sklearn.feature_extraction.text` module. The TF-IDF vectorizer calculates the importance of each word in the document relative to the entire corpus and generates a sparse matrix of features. Selected 20% of vocabulary size as maximum features as they will be the most decision-making features(Pareto principle).

2. Design Choices for the Neural Network

The neural network architecture chosen for this sentiment analysis task is a sequential model with three dense hidden layers and one output layer. Here are the design choices:

a. Activation Function: The ReLU (Rectified Linear Unit) activation function was used for the hidden layers. ReLU helps with faster convergence and reduces the likelihood of vanishing gradient problems.

b. Output Layer Activation: For binary classification, the sigmoid activation function was used in the output layer. The sigmoid function squashes the output between 0 and 1, making it suitable for binary classification tasks.

c. Dropout Regularization: Dropout layers were added after each dense layer to prevent overfitting. Dropout randomly sets a fraction of the neurons to zero during training, forcing the network to learn robust features.

d. Regularization: L2 regularization was applied to the dense layers to add a penalty on the model's complexity, reducing the risk of overfitting.

3. Training and Testing Accuracies

The model was trained on the IMDB sentiment analysis dataset, and the training and validation accuracies were monitored during training. After training, the model was evaluated on a separate test dataset to obtain the test accuracy.

a. K-Fold Cross-Validation: In some cases, the available dataset might be limited, and a single train-test split could lead to suboptimal model evaluation due to the randomness in the split. K-fold cross-validation ensures that every data point is used for both training and validation at least once, maximizing the utilization of the available data.

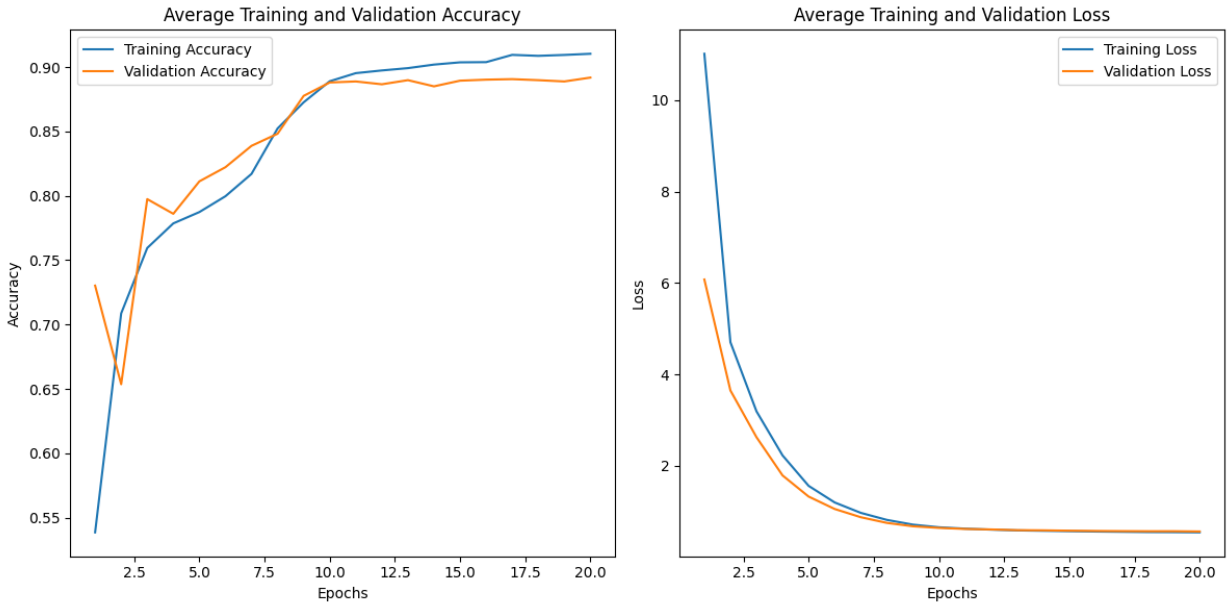
   So, to evaluate the model's performance more robustly, we employ K-fold cross-validation. The training dataset is divided into K subsets (5 folds) of approximately equal size. The model is trained K times, with each fold serving as the validation set once, while the remaining K-1 folds are used for training. We then average the evaluation metrics across all K folds to obtain more reliable performance estimates.

b. Results: Using K-fold cross-validation, we achieved an average validation accuracy of 88.66% over K folds and the best validation accuracy was 88.88%. This demonstrates the model's ability to generalize well across different subsets of the data, reducing the risk of overfitting or underfitting. The test data accuracy was 87.62%.

Result Table

Fold Data	***Accuracy (in%)***
1st	88.68
2nd	88.60
3rd	88.60
4th	88.88
5th	88.54
Average	**88.66**
Best	**88.88**
Test Data Accuracy	**87.62**

Best model's Training and Validation accuracy, loss vs epoch count plot



4. Output Comments

The model achieved reasonably good validation and test datasets accuracy, demonstrating its ability to generalize to unseen data. The use of K-fold cross-validation has increased the confidence in the model's performance, making it more reliable for real-world applications. The plots of accuracy and loss against epochs show that the model converged and was not overfitting. The training accuracy improved as the number of epochs increased, but the validation accuracy plateaued, indicating a good balance between fitting and generalization.

5. Potential Improvements

a. Hyperparameter Tuning: Fine-tuning hyperparameters such as the learning rate, dropout rate, L2 regularization, and the number of neurons in hidden layers could further improve the model's performance.

b. Different Architectures: Trying different network architectures, such as using LSTM or GRU layers for sequential data, could be beneficial for capturing long-term dependencies in the text. Since the provided dataset has an average length of each corpus of just around 200 so the LSTM or RNN etc. will trade-off training time with very slight or almost no increase in accuracy.

c. Ensemble Methods: Combining multiple models using ensemble techniques like bagging or boosting might boost the overall accuracy.

d. Data Augmentation: If the dataset is limited, data augmentation techniques like adding noise, flipping, or rotating the text could help generate more samples.

In conclusion, the sentiment analysis model using a feedforward neural network and K-fold cross-validation achieved satisfactory performance on the provided dataset by employing appropriate preprocessing steps and a well-designed neural network architecture. Fine-tuning hyperparameters and exploring other architectures may further improve the model's performance. The model can now be used to analyze the sentiment of text data and can be extended to handle larger datasets and real-world applications.