SENTIMENT ANALYSIS ON IMDB MOVIE REVIEWS

Executive Summary

The IMDB dataset is a binary classification problem where the goal is to predict whether a movie review is positive or negative. The dataset consists of 50,000 reviews with 25,000 for training and 25,000 for testing. Pre-Processing is performed on the data and it includes converting words to integers, and then integers to tensors using a technique known as One hot encoding. Later, we feed data to the neural network model and evaluate the performance by trying different approaches like using different activation functions, loss functions, and different number of hidden layers. Finally, we end up with a model with the best hyperparameters based on the validation accuracy.

Problem

The IMDB dataset is a binary classification problem where the goal is to predict whether a movie review is positive or negative.

Technique

Dataset and Preprocessing:

The IMDB dataset is a collection of movie reviews with sentiment labels (positive or negative). The dataset is preprocessed by converting each review into a sequence of word embeddings, where each word is represented by a vector of fixed size. The maximum vocabulary size is set to 10,000.

Also, the reviews, which were originally a sequence of words, were changed into a sequence of integers, where each integer represented a different word. Even though we now have a list of numbers, they are not appropriate for our neural network model's input. The integers must be transformed into tensors. The list of integers could be transformed into a tensor with integer data type and shape (samples, word indices). To accomplish that, we need to make sure that each sample is the same length, thus we would need to pad each review with dummy words (integers) to make sure that they are all the same length.

The dataset is split into a training set (50%) and a validation set (50%). The neural networks are trained on the training set and evaluated on the validation set.

Hyperparameter Tuning:

The number of hidden layers, activation functions, and loss functions are the hyperparameters that are tuned to obtain the best performance.A dropout rate of 0.5 has been used, means that 50% of the input units are randomly set to zero at each update , which helps prevent units from co-adapting too much to each other and reduces the dependence of the model on specific features.A grid search is performed to search for the best hyperparameters. The grid search is performed over the following hyperparameters:

Number of Hidden Layers: 1, 2, 3
Activation Functions: relu, sigmoid, tanh

Loss Functions: binary_crossentropy, mean_squared_error

Results: The table below shows the accuracy and validation loss for each approach.

Hidden Dense Layers	Activation Function	Loss Function	Last-layer Activation Function	(Epochs,BatchSize)	Accuracy(%)	Validation Loss
1	relu	Binary_crossentropy	Sigmoid	(10,512)	88.2	0.289
2	relu	Binary_crossentropy	Sigmoid	(10,512)	88.1	0.366
3	relu	Binary_crossentropy	Sigmoid	(10,512)	87.7	0.486
1	relu	mean_squared_error	Sigmoid	(10,512)	88.2	0.087
2	relu	mean_squared_error	Sigmoid	(10,512)	88.1	0.091
3	relu	mean_squared_error	Sigmoid	(10,512)	87.9	0.097
1	tanh	Binary_crossentropy	Sigmoid	(10,512)	87.7	0.305
2	tanh	Binary_crossentropy	Sigmoid	(10,512)	86.1	0.478
3	tanh	Binary_crossentropy	Sigmoid	(10,512)	87.0	0.432
1	tanh	mean_squared_error	Sigmoid	(10,512)	87.5	0.091
2	tanh	mean_squared_error	Sigmoid	(10,512)	86.6	0.105
3	tanh	mean_squared_error	Sigmoid	(10,512)	87.5	0.105
1	tanh	Binary_crossentropy	Sigmoid	(3,512)	88.3	0.089

First, we trained the model for 10 epochs with a batch size of 512,but the validation results started to deteriorate after 3 epochs of training. So,we set the epochs to 3,to avoid over-training. After doing so, the best accuracy (**88.3%**) and a validation loss of **0.089** is achieved by using a neural network with 1 hidden layer, sigmoid activation function for the last layer, tanh activation function for the hidden layer for which we have used the dropout as a fully connected layer takes up the majority of the parameters, co-dependency between neurons during training reduces each neuron's own power and causes the training data to be overfit, and mean_squared_error as a loss function.Finally,we are able to classify the reviews.