Summary

Given IMDB dataset has 50000 reviews. A total of 10000 words are used in these reviews. All these reviews are already labelled. Following are the steps I followed to find the best combination of hyperparameters to produce better validation and test accuracy.

First, I imported all modules required to build and run a model. They are

```
import os
from operator import itemgetter
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import tensorflow as tf

from keras import models, regularizers, layers, optimizers, losses, met
rics
from keras.models import Sequential
from keras.layers import Dense
from keras.utils import np_utils, to_categorical

from keras.datasets import imdb
```

Next loaded the IMDB dataset

The dataset is divided into two parts one for training and the other of testing. Each part has 25000 reviews in it.

After that, the data is vectorized as we cannot feed integers into neural networks

Further training set is divided into two parts as partial training and validation set.

Building model:

The first three models are built with 16 hidden units, ReLU activation function, rmsprop optimizer, and binary cross entropy loss function. The number of hidden layers in each model is Changed. The following table presents the training, validation, and test accuracy

Combination	Training accuracy	Validation accuracy	Test accuracy
One hidden layer	0.9450	0.8894	0.8824
Two hidden layers	0.9983	0.8696	0.8582
Three hidden layers	0.9984	0.8440	0.8326

Since the data samples in IMDB dataset are limited we can say that one hidden layer performances better compared to other two layers. Multiple layers are leading to overfitting thus leading to fall in validation and test accuracy.

Next two models are build with one hidden layer, ReLU activation function, rmsprop optimizer, and binary cross entropy loss function. The number of hidden units are changed to 32 and 64. The following table shows the results

Combination	Training accuracy	Validation accuracy	Test accuracy
16 hidden units	0.9450	0.8894	0.8824
32 hidden units	0.9925	0.8731	0.8615
64 hidden units	0.9906	0.8698	0.8591

Increasing the hidden units is also leading to the same problem of overfitting as that of the hidden layers. 16 hidden units is best combination to use as it is producing the best accuracy.

Now, let’s check the accuracies by changing the loss function to mse and compare it with binary cross entropy. These models are one hidden layer,16 hidden units, ReLU activation function, rmsprop optimizer

Combination	Training accuracy	Validation accuracy	Test accuracy
Binary cross entropy	0.9450	0.8894	0.8824
MSE	0.9830	0.8782	0.8701

For the given dataset it is evident that the binary cross entropy loss function is better to use compared to MSE loss function.

Next model is built using one hidden layer,16 hidden units, rmsprop optimizer, Binary cross entropy loss function to compare the performance of tanh and relu activation function

Combination	Training accuracy	Validation accuracy	Test accuracy
relu	0.9450	0.8894	0.8824
tanh	0.9925	0.8737	0.8637

Relu has given better accuracy than tanh. So, I’m going to use relu as my activation function for the next models.

Using dropout

In the next model, I used the Drop out (0.5). In this model, I used one hidden layer, 16 hidden units, rmsprop optimizer, Binary cross entropy loss function and relu activation function

Combination	Training accuracy	Validation accuracy	Test accuracy
Without drop out	0.9450	0.8894	0.8824
With drop out (0.5)	0.9772	0.8844	0.8753

Using dropouts is leading to fall in accuracy. This might be so as the model is not overfitting and usage of dropout is a disadvantage in such a situation.

Using regularizers:

The next two models are built using one hidden layer, 16 hidden units, rmsprop optimizer, Binary cross entropy loss function, and relu activation function. L1 and L2 regularizers are used and compared their performance

Combination	Training accuracy	Validation accuracy	Test accuracy
No regularizer	0.9450	0.8894	0.8824
L1 regularizer	0.9993	0.8759	0.8667
L2 regularizer	0.9925	0.8798	0.8706

From the above table, it is evident that regularizers are not needed for this data as its usage is not helping to improve the accuracy.

Comparing adam optimizer with rmsprop. The model is built with one hidden layer, 16 hidden units, a Binary cross entropy loss function, and a relu activation function

Combination	Training accuracy	Validation accuracy	Test accuracy
rmsprop	0.9450	0.8894	0.8824
Adam	0.9816	0.8833	0.8721

Rmsprop has performed better than adam for the given IMDB dataset

Conclusion:

There are several factors that affect the performance of a model in a neural network. The first one is the amount of data sample. We need more data samples to get better accuracy.

Now, In this scenario with 25000 samples, one layer performs better with 16 hidden units than other hidden layers. Since the model is obtaining a good with the ReLU function.

When I tried with different optimizers and regularizers, the accuracy is going down, I realise that we don’t really need to use it unless it is required.

Maybe in the scenario when we have a huge amount of data and we train with multiple layers and use this optimizer, regularizers, and dropouts may help the model to perform better.

Finally, for the given dataset the best combination of hyperparameters is one hidden layer, 16 hidden units, Binary cross entropy loss function, relu activation function, and rmsprop optimizer