Summary

The given IMDB dataset has 50000 reviews. These reviews contain 10,000 words in total. All of these reviews have already been labelled. Here are the procedures I took to identify the best set of hyperparameters to improve test and validation accuracy.

I started by importing all of the modules needed to create and run a model. Those 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, metrics
from keras.models import Sequential
from keras.layers import Dense
from keras.utils import np_utils
from tensorflow.keras.utils import to_categorical

from keras.datasets import imdb
```

Then the IMDB dataset was loaded.

The dataset is separated into two sections: training and testing. Each section contains 25000 reviews. The data is then vectorized because we cannot input integers into neural networks after that. The training set is further separated into two parts: partial training and validation set.

MODEL BUILDING:

To construct the first three models, 16 hidden units, the ReLU activation function, the rmsprop optimizer, and the binary cross entropy loss function were used. Each model has a modified number of hidden layers. The accuracy for training, validation, and testing is shown in the following table.

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

We can say that one hidden layer performs better than the other two layers in the IMDB dataset because there are less data samples available. Multiple layers cause overfitting, which lowers test and validation accuracy.
The next two models are constructed using a single hidden layer, the ReLU activation function, the rmsprop optimizer, and the binary cross entropy loss function. The hidden units are now 32 and 64 in number. Results are displayed in the following table.

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

Similar overfitting issues as with hidden layers are being caused by increasing the hidden units. To achieve the best accuracy, the optimal combination to employ is 16 concealed units.

Let's now compare the accuracy to binary cross entropy while altering the loss function to mse. These models have one hidden layer, 16 hidden units, a ReLU activation function, and an 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:

Several factors influence the performance of a model in a neural network. The first is the size of the data sample. To improve accuracy, we require additional data samples. Now, one layer with 16 hidden units outperforms other hidden layers in this case with 25000 samples. Considering that the ReLU function gives the model a benefit. 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.

The model might perform better in the case if we train with multiple layers, utilise regularizers, and dropouts and have a large amount of data.
Finally, for the provided dataset, one hidden layer, 16 hidden units, the Binary cross entropy loss function, the RELU activation function, and the RMSPROC optimizer are the best hyperparameter combinations.