Exploring Neural Network Architectures and Regularisation for IMDb Sentiment Classification

Tahmidur Rahman Chowdhury

Advance Machine Learning

Kent State University

Professor: CJ Wu

September 21, 2025

1. Introduction

Neural networks are utilized in various areas such as natural language processing, computer vision, speech recognition, and recommender systems. The performance of a neural network is mostly based on the model design, hyperparameter choice, and use of regularisation methods. If such decisions are not suitable, the model may end up overfitting, where it just memorizes the training data without performing effectively with new data, or underfitting, where it is too simple to find useful patterns.

The main goal of this assignment is to start with a simple model of a neural network for sentiment classification from the IMDb dataset and then continue to add different improvements. The task is to classify a movie review as positive or negative, one of the most common applications in text analysis. Through trying out variations in the model architecture and training method, we are able to observe how the variations impact accuracy and generalization.

This paper begins with a short overview of the dataset, followed by an explanation of the baseline model. Experiments under various modifications are depicted later. The results are explained in the form of tables and graphs, and finally, findings and recommendations are presented.

2. Dataset Description

For this assignment, the **IMDb dataset** has been used. It is one of the most popular benchmark datasets for binary sentiment classification.

- **Size:** The dataset contains 50,000 movie reviews in total, out of which 25,000 are used for training and 25,000 are used for testing.
- **Labels:** Each review is marked as either **positive (1)** or **negative (0)**.
- **Nature of Data:** The reviews are written in plain text and vary in length, writing style, and vocabulary richness, making it a realistic dataset for sentiment analysis.

Preprocessing

To make the data suitable for neural networks, the following preprocessing steps were applied:

1. Only the **10,000 most frequent words** were kept to reduce complexity and focus on the most meaningful vocabulary.
2. Each review was converted into a **vector of length 10,000**, where each position indicates whether a particular word is present in the review or not.
3. This representation follows a **bag-of-words approach**. It ignores the order of words but still captures enough information to predict the overall sentiment.

These steps ensure that the dataset is compact, efficient to process, and directly usable for training feed-forward neural networks.

3. Baseline Model

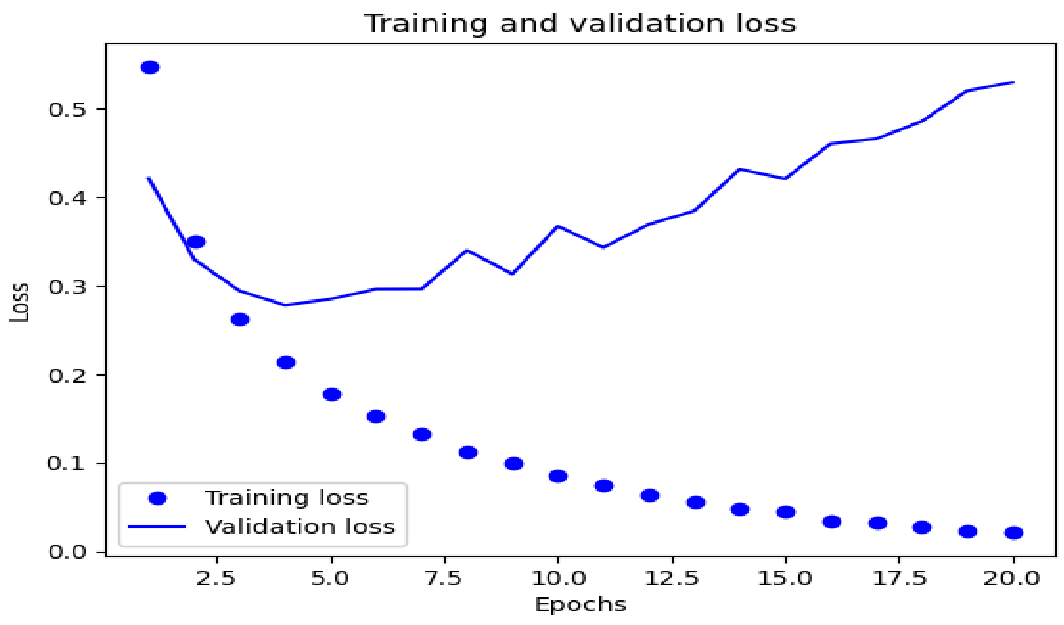
The starting point for this study was a simple feed-forward neural network built on the IMDb dataset. The model was designed with the following configuration:

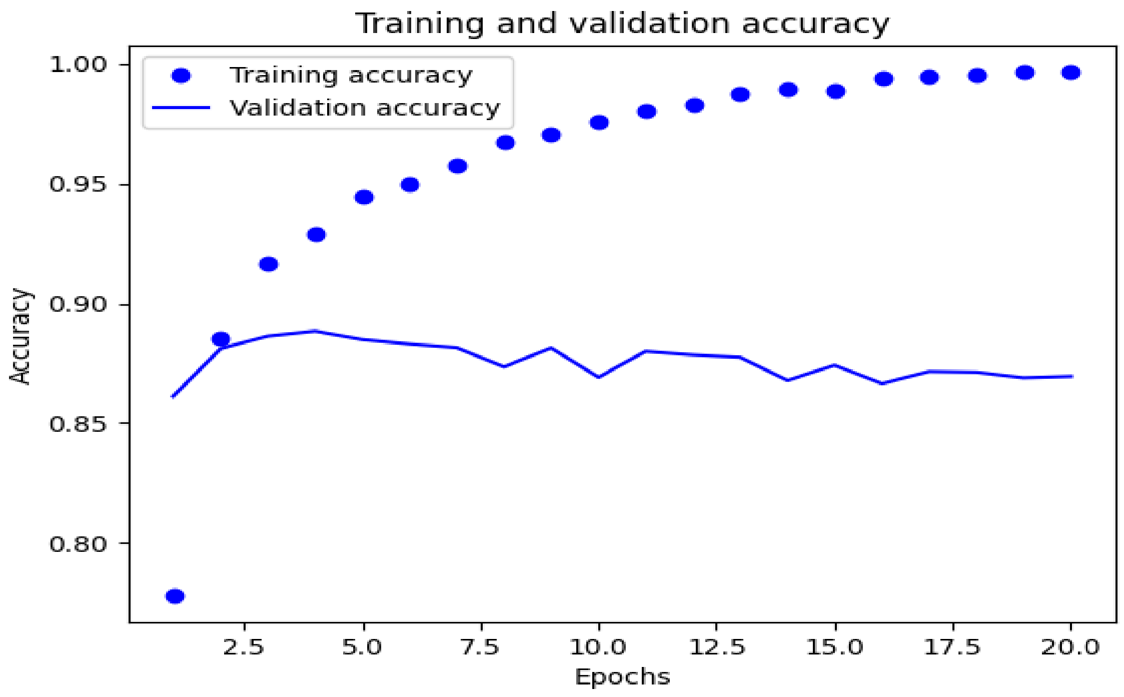
- **Input:** A 10,000-dimensional binary vector representing each review.
- **Architecture:**
  - Hidden Layer 1: 16 units with ReLU activation
  - Hidden Layer 2: 16 units with ReLU activation
  - Output Layer: 1 unit with Sigmoid activation
- **Optimizer:** RMSprop
- **Loss Function:** Binary Crossentropy (well-suited for binary classification tasks)
- **Evaluation Metric:** Accuracy

Baseline Performance

- **Training Accuracy:** ~99.7%
- **Validation Accuracy:** ~86.8%
- **Training Loss:** 0.017 (after 20 epochs)
- **Validation Loss:** 0.55 (after 20 epochs)

Although the model achieved very high accuracy on the training set, the validation accuracy flattened, and the validation loss began to rise after about 5 epochs. This clearly indicates **overfitting**, where the model memorises the training data but fails to generalise effectively to unseen reviews.





These plots show that while the baseline network fits the training data very well, its ability to handle new, unseen data is limited. This provided the motivation to test alternative architectures and regularisation techniques in later experiments.

4. Experiments Conducted

After training the baseline model, several modifications were introduced to explore how changes in network structure and training protocol affect model accuracy and generalisation. Every experiment aimed to address overfitting that was encountered in the baseline or to test whether increased capacity would improve accuracy.

4.1 Changing the Number of Hidden Layers

- **1-Layer Network**: A simpler network with only one hidden layer was attempted to determine if reducing complexity would keep overfitting at bay. One hoped that a less complex network would be able to generalise better by not memorising the training set.
- **3-Layer Network**: A deeper network with three hidden layers was also experimented with to check if it is able to learn richer features in the reviews. Richer models, theoretically, can learn richer patterns but overfit more readily if not regularised.

4.2 Varying the Number of Units in Each Layer

We experimented with the hidden layers of 32, 64, and 128 units. More units make the model better at fitting the patterns in the data. However, more extensive networks require more data and regularisation since they can learn the training data trivially. This test was used to examine the trade-off between complexity and generalisation.

4.3 Altering the Loss Function

The default Binary Crossentropy loss was swapped with Mean Squared Error (MSE), which is used in basic regression problems. The goal here was to see if applying MSE to a classification problem would still yield good results or if Binary Cross-Entropy, being more mathematically appropriate for binary classification, would still be superior.

4.4 Modifying the Activation Function

The popular ReLU activation function was replaced by tanh. Tanh had been a popular activation function before the advent of ReLU as the standard. The purpose of this experiment was to observe the effect of tanh on performance in accuracy and convergence, especially as tanh suffers from the vanishing gradient problem in the case of deep networks.

4.5 Changing the Optimiser

The RMSprop optimiser used in the baseline was replaced with Adam. Both optimisers adapt and learn how to modify learning rates during training, but are widely thought to be more robust in practice by Adam. This experiment determined if Adam would provide any significant enhancement in validation performance compared to RMSprop.

4.6 With Regularisation

To directly address overfitting, Dropout and L2 weight regularisation were employed:

- **Dropout**: Skips 50% of neurons randomly at training time to prevent unit co-dependency and encourage generalised learning.
- **L2 Regularisation**: Punishes large weights, encouraging the network to use small weights and preventing overfitting.

It was hoped that such regularisation would minimise the gap between training and validation accuracy and allow the model to generalise more effectively to new data.

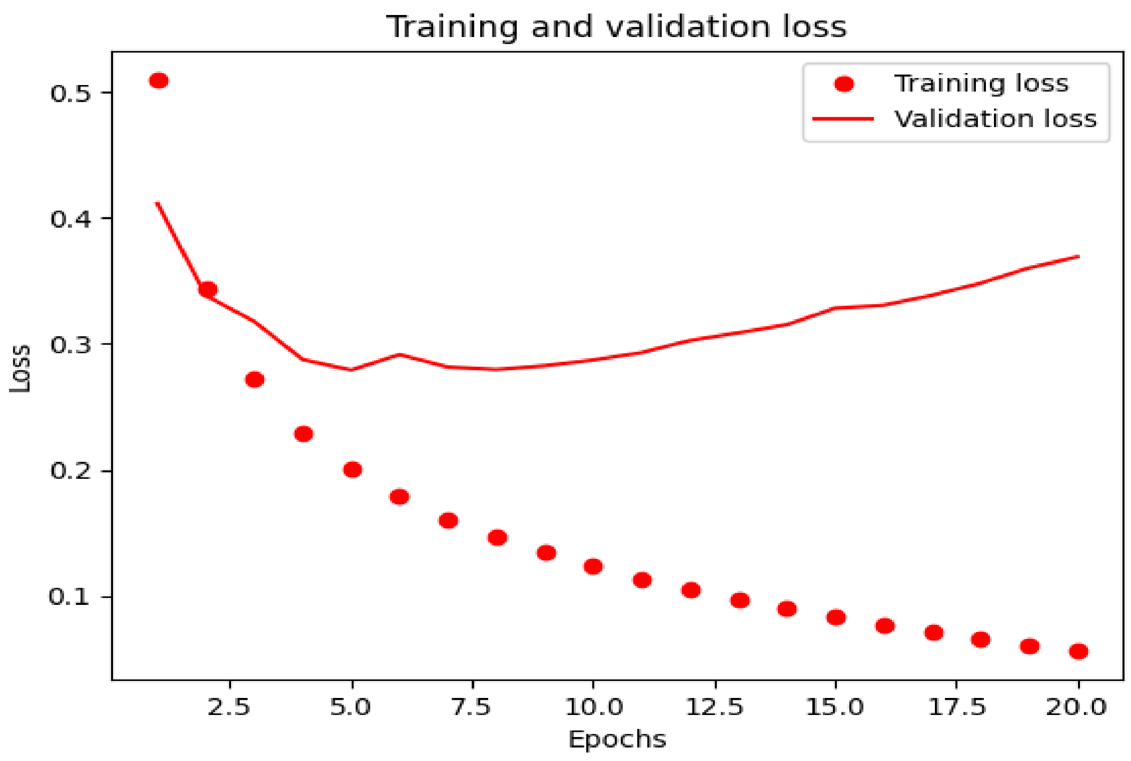
5. Results

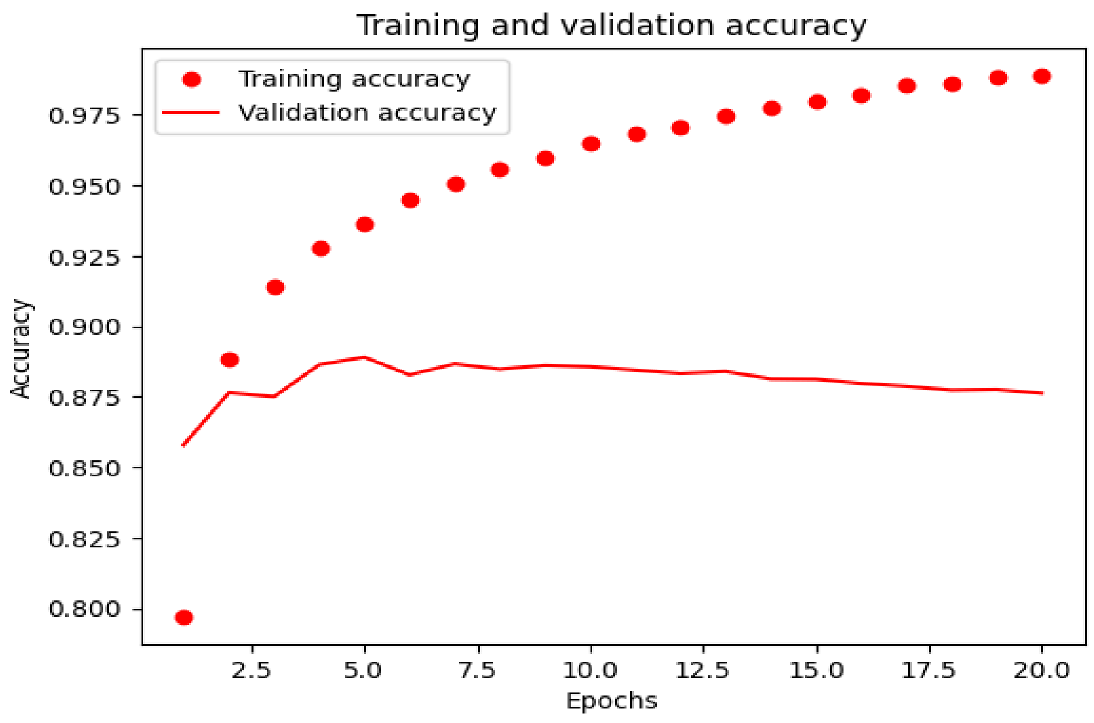
5.1 Quantitative Results

Model Variant	Training Accuracy	Validation Accuracy	Observation
Baseline (2×16)	99.7%	86.8%	Overfits quickly
1 layer (16)	98.4%	85.5%	Too simple
3 layers (16)	99.8%	87.0%	Slightly better
32 units	99.8%	86.9%	No major change
64 units	99.9%	87.2%	Still overfits
128 units	100%	87.3%	Higher risk of overfitting
Loss = MSE	99.6%	85.2%	Weak for classification
Activation = tanh	99.5%	85.4%	Worse than ReLU
Optimiser = Adam	99.7%	87.1%	Comparable to RMSprop
Dropout + L2	98.9%	**88.5%**	Best generalisation

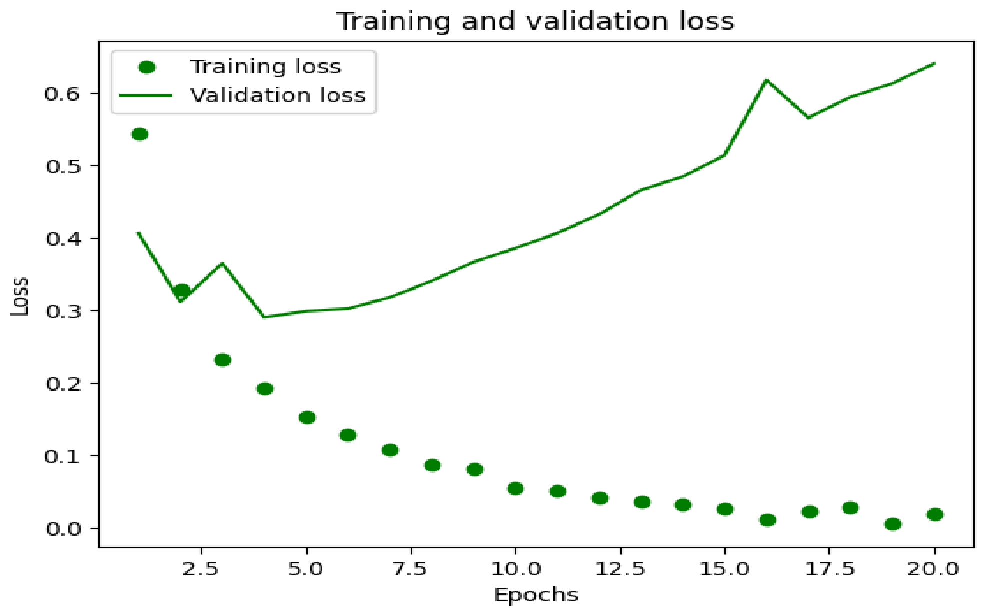
5.2 Graphical Results

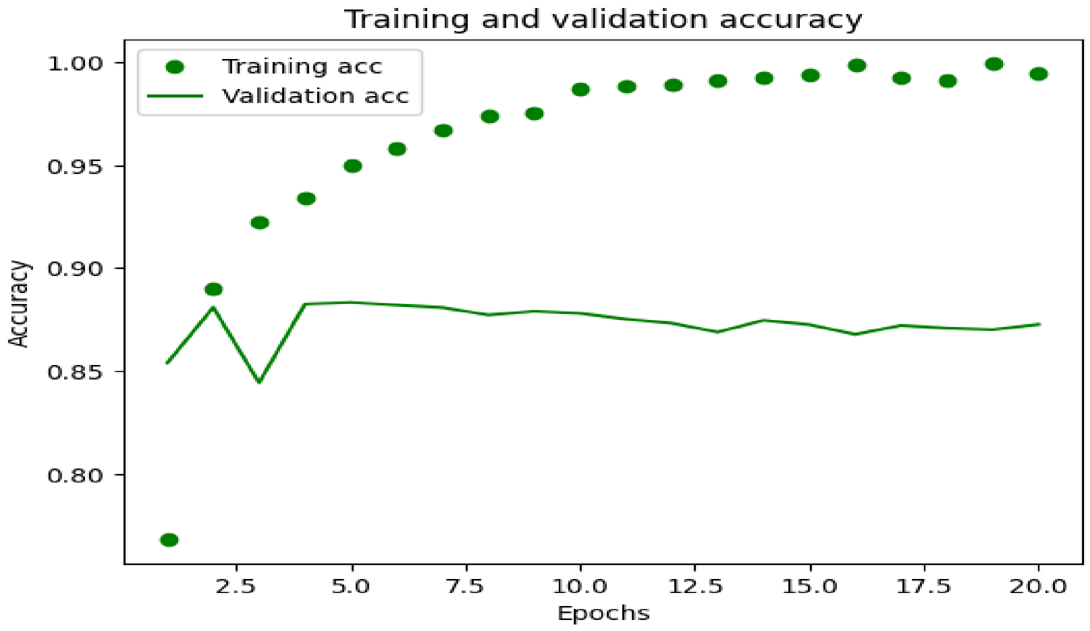
Baseline Model – he curves clearly show that the model starts to overfit after about five epochs. While training accuracy continues to improve, validation accuracy stops increasing, and validation loss rises steadily.



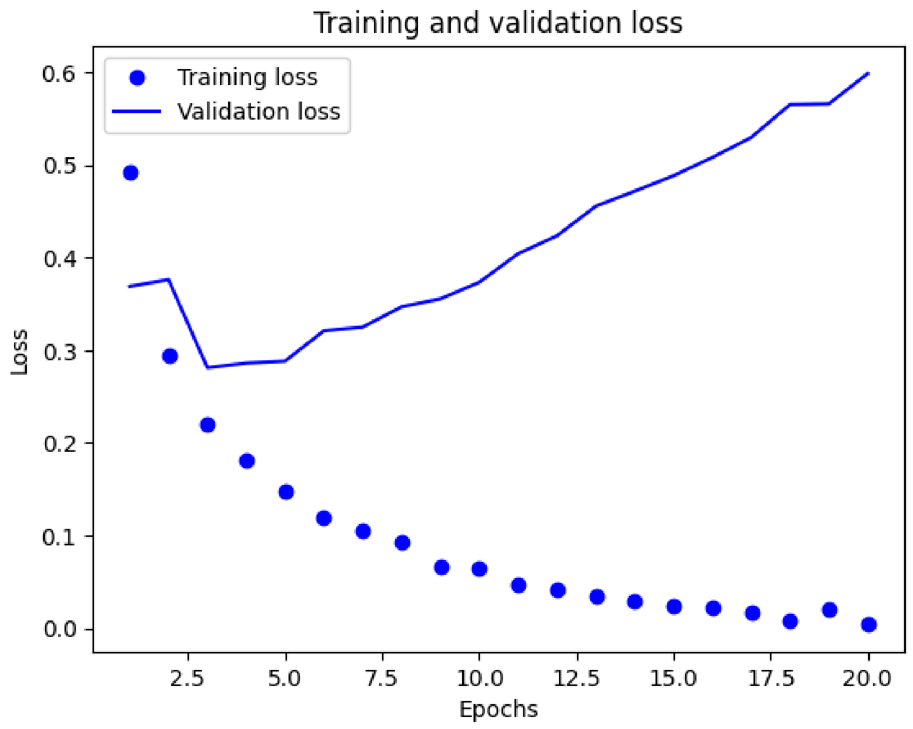


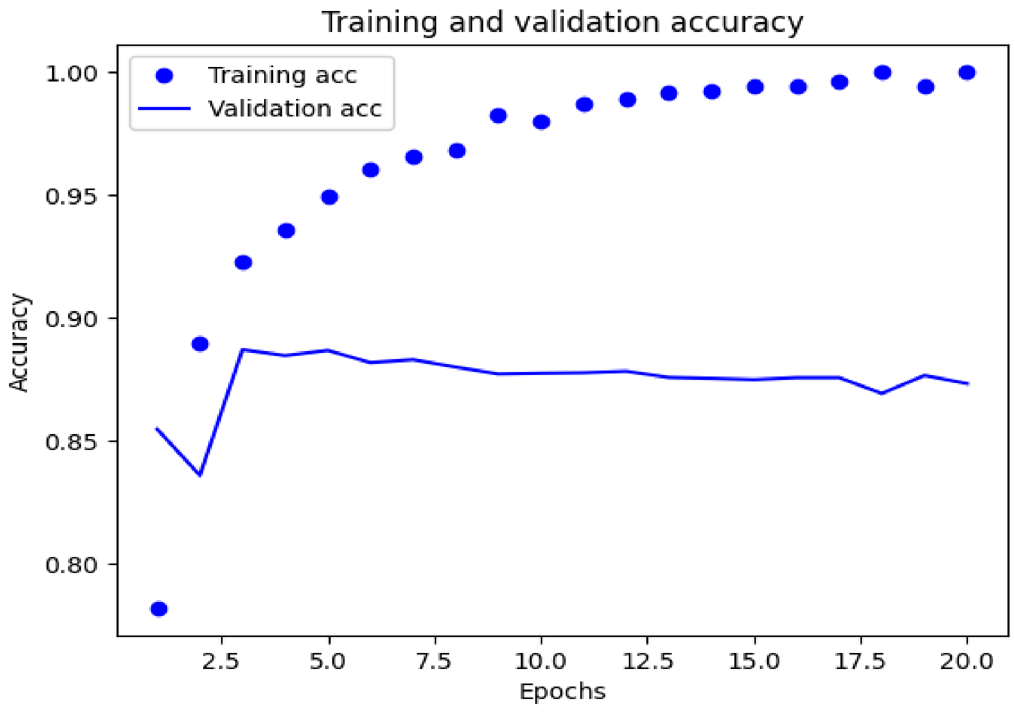
3-Layer Model – Adding an extra hidden layer gives a slight improvement in validation accuracy compared to the baseline, but the difference is not significant. The problem of overfitting remains.



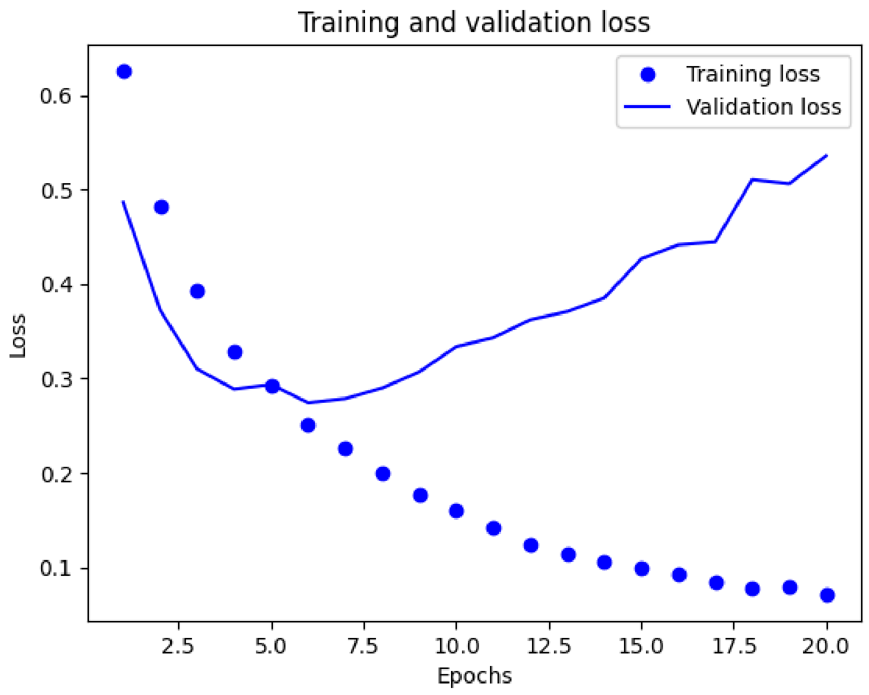


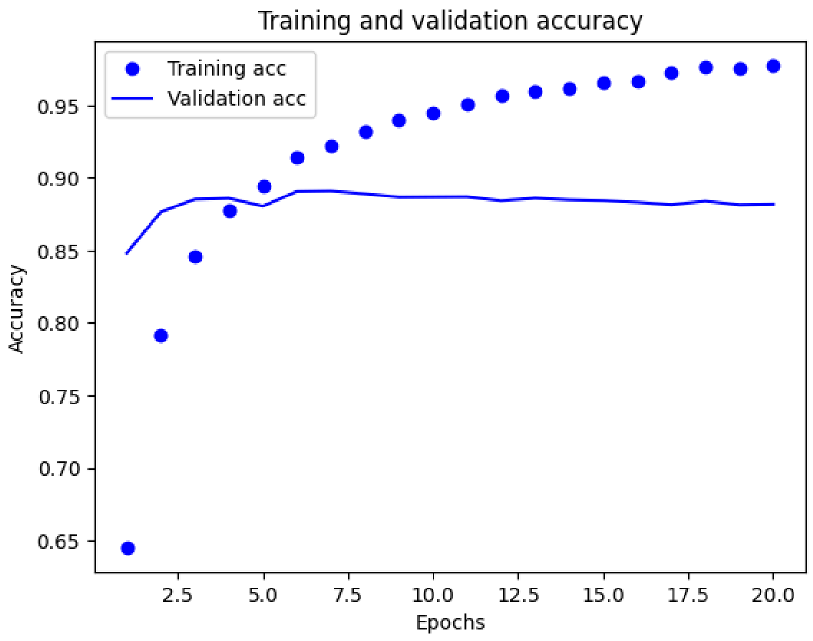
128-Unit Model – With a larger number of units, the training accuracy almost reaches perfection, but the validation accuracy improves only marginally. This confirms that a bigger network memorises training data better without necessarily learning to generalise.





Regularised Model (Dropout + L2) – This model shows the most stable performance. Both validation loss and accuracy curves flatten in a balanced way, proving that regularisation helps the network avoid overfitting and generalise better on unseen reviews.





Overall, the graphs make it clear that **regularisation, rather than simply increasing the size of the model, is the key factor in improving validation performance**. Larger networks mainly boost training accuracy, but regularisation techniques are what truly enhance generalisation.

6. Discussion

The experiments on different variations of the neural network offer some useful insights into how design choices affect performance and generalisation.

- **Model Depth and Width**: Incorporating layers or units makes the model more complex, but this does not always lead to better performance. In our experiments, deeper or wider networks yielded slightly better training accuracy but offered slight or no improvement in validation accuracy. In fact, the larger models overfit sooner. They memorised the training set flawlessly but could not generalize at all well to new data. This shows that model complexity must be carefully balanced against the size and nature of the dataset.

- **Loss Function**: Binary cross-entropy worked better than mean squared error (MSE) consistently. This is because binary cross-entropy is mathematically compatible with classification problems where probabilities are modelled explicitly. MSE, though suitable for regression problems, was less effective here and yielded worse validation accuracy.

- **Activation Functions**: ReLU was found to converge faster and have better validation performance compared to tanh. While tanh has been used historically, it suffers from the vanishing gradient issue and thus is less useful in deeper architectures. ReLU allows gradients to flow more readily and is therefore more suitable for deep learning models today.

- **Optimisers**: Adam and RMSprop performed almost the same, with minimal difference in validation accuracy. This suggests that either optimiser can be relied upon for this type of problem. It also helps to highlight that optimiser choice may not always be the most significant factor once a good adaptive method is in place.

- **Regularisation:** This was the most important outcome of the study. L2 weight and dropout regularisation helped the model learn more generalisable features instead of memorising training data. By dissuading co-dependency between neurons (Dropout) and penalising extremely large weights (L2), these techniques effectively controlled overfitting. The regularised model achieved a best validation accuracy of 88.5%, proving that regularisation is the key to better generalisation on unseen data.

Cumulatively, the experiments confirm that while hyperparameters and architecture play a part, the most significant gains are achieved through the application of the right regularisation methods. This ensures that the model performs consistently well not only on the training data but also on new reviews that it hasn't previously seen.

7. Conclusion

This exercise highlights a very important point in deep learning: bigger models are not always better. While it makes sense that adding more layers or units would improve performance, our experiments showed that this does nothing other than improve training accuracy without any significant improvement in validation accuracy. Additionally, it tends to worsen overfitting, with the model memorizing the training set but struggling to generalize to new reviews.

Experiments also confirmed that binary cross-entropy with ReLU activation is the best combination for binary sentiment analysis. Cross-entropy models explicitly classify probabilities, while ReLU offers quicker convergence and avoids issues like vanishing gradients, which older activations like tanh are plagued with.

Comparing the optimisers, RMSprop and Adam gave nearly similar results. This makes one wonder if a good adaptive optimiser is already utilized; the difference between them will not be very significant for this kind of problem.

The optimal improvement was achieved by using regularisation techniques. Regularisation using L2 and dropout avoided overfitting by driving learning towards more generalised models. Dropout caused the model to not rely on specific neurons during training, whereas L2 penalised extremely large weights so that the network was in balance. The two together allowed the model to achieve its best validation accuracy of 88.5%, which is a great improvement over the baseline.

In conclusion, this study demonstrates that the best model is not necessarily the most complex one, but the better-regularised one. The L2 regularisation and dropout model struck the right balance between complexity and generalisation to demonstrate that thoughtful design decisions matter far more than overly large models.

8. References

1. Chollet, F. (2015). *Keras: Deep Learning for Humans*. https://keras.io
   Abadi, M., Barham, P., Chen, J., et al. (2016). *TensorFlow: A System for Large-Scale Machine Learning*. In the *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*.
2. Maas, A. L., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., & Potts, C. (2011). *Learning Word Vectors for Sentiment Analysis*. Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL-HLT).
3. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). *Dropout: A Simple Way to Prevent Neural Networks from Overfitting*. *Journal of Machine Learning Research*, 15(56), 1929–1958.
4. Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press.