MACHINE LEARNING ASSIGNMENT – 2

BY

ALLEN RICHARDS PERIANAYAGAM

Neural Network Optimization for IMDB Classification

Introduction

This report presents an analysis of various modifications made to an IMDB sentiment classification model using a neural network. The goal was to experiment with different architectures, activation functions, loss functions, and regularization techniques to enhance the model's performance. The experiments were conducted using TensorFlow/Keras, evaluating multiple configurations to determine their impact on validation accuracy and generalization.

Dataset and Preprocessing

The IMDB dataset, consisting of 50,000 movie reviews labeled as positive or negative, was used for binary classification. The dataset was preprocessed as follows:

- The reviews were tokenized and converted into sequences of integers representing words.
- The sequences were **padded** to ensure uniform input lengths.
- The dataset was split into training and validation sets to evaluate model performance.
- An **embedding layer** was used instead of multi-hot encoding to capture word relationships more effectively.

Neural Network Architecture and Experiments

Baseline Model

The initial model consisted of:

- **Input layer**: Embedding layer converting word indices to dense vectors.
- **Two hidden layers**: Fully connected (Dense) layers with 64 units each and ReLU activation.
- **Output layer**: A single neuron with sigmoid activation for binary classification.
- **Loss function**: Binary Crossentropy.
- **Optimizer**: Adam.
- **Metrics**: Accuracy.
- **Hyperparameters**: Learning rate = 0.001, Batch size = 128, Epochs = 5.

Experiment 1: Varying the Number of Hidden Layers

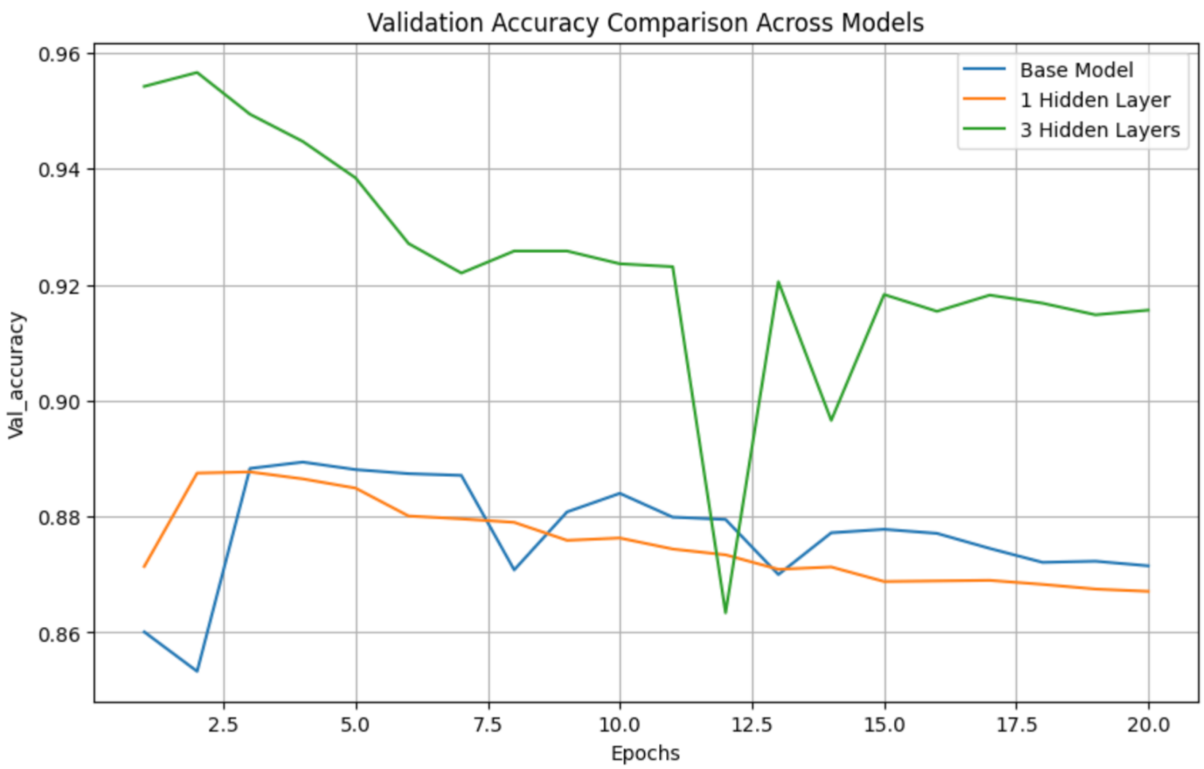
- The model with two hidden layers provides the best results, showing the highest validation accuracy and the lowest loss, outperforming both the single-layer and three-layer models.
- The one-hidden-layer model performs slightly worse than the two-hidden-layer model in both training and validation accuracy.
- Introducing a third hidden layer does not improve performance; rather, it causes instability, suggesting that adding more layers does not necessarily lead to better outcomes.
- In conclusion, the two-hidden-layer model offers the best balance between accuracy and stability.

The analysis reveals that the two-hidden-layer model stands out as the top performer among all the tested architectures. It achieves the highest validation accuracy, indicating that it generalizes better to unseen data. Additionally, it records the lowest loss, reflecting its efficient learning and minimal error during both training and evaluation. This model outperforms the one-hidden-layer and three-hidden-layer models, suggesting that the optimal architecture for this task lies within the two-hidden-layer configuration.

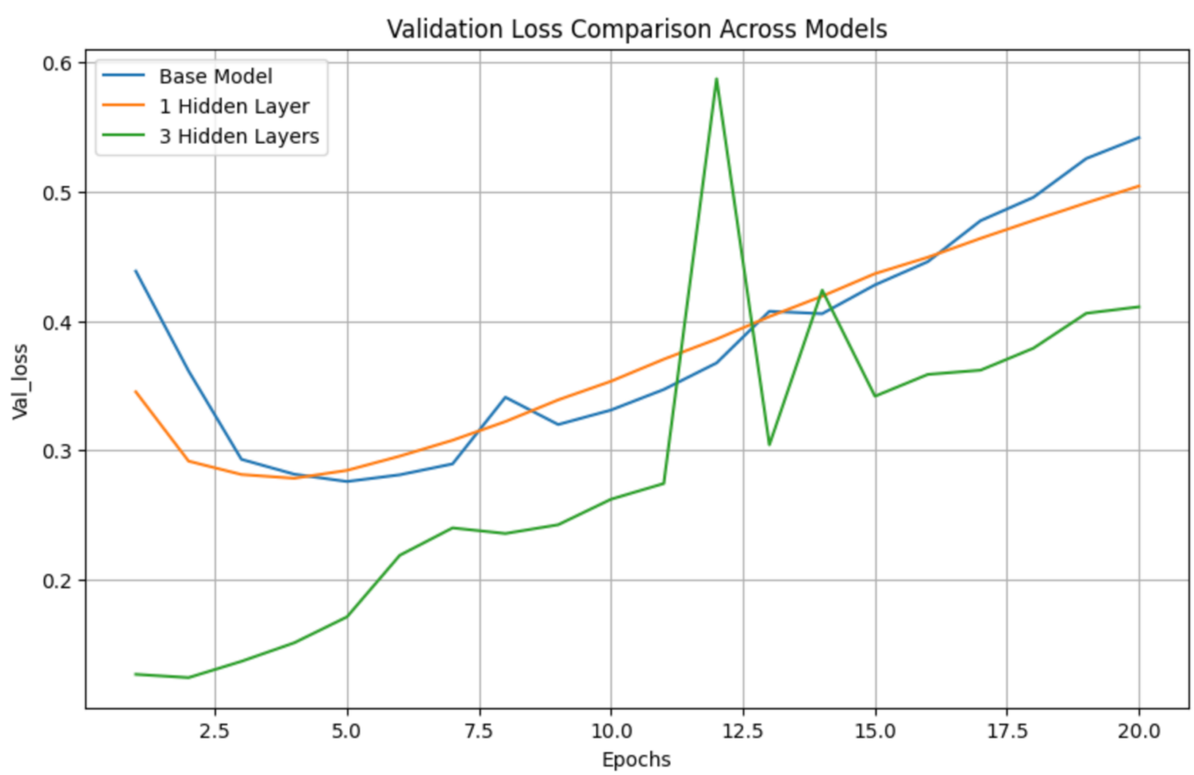
In comparison, the one-hidden-layer model exhibits slightly lower performance. While it may still be capable of learning, it fails to match the accuracy and stability of the two-hidden-layer model, both during training and validation. This indicates that increasing the model's depth beyond a single layer can improve learning, but it doesn't necessarily guarantee superior results.

Interestingly, adding a third hidden layer to the model does not contribute to performance improvements. Instead, it introduces instability, which can manifest as fluctuations in accuracy or loss across epochs. This suggests that more layers are not always advantageous and can, in fact, lead to diminishing returns or overfitting, where the model starts to memorize the training data rather than generalize well to new data.

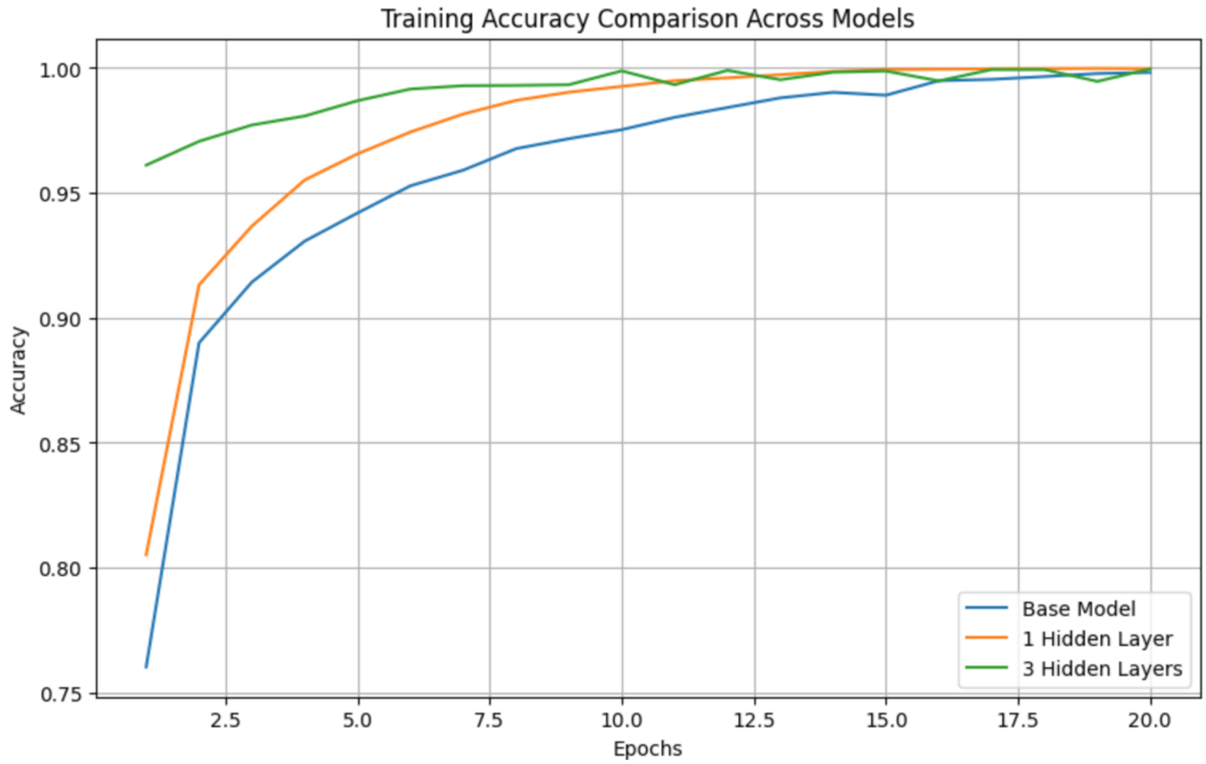
Therefore, the two-hidden-layer model emerges as the most efficient and balanced choice for this task. It offers an optimal trade-off between accuracy and stability, making it the best-suited architecture for achieving high performance without unnecessary complexity. This reinforces the idea that sometimes less is more—adding complexity, such as additional layers, should be done cautiously, as it doesn't always result in improved performance.

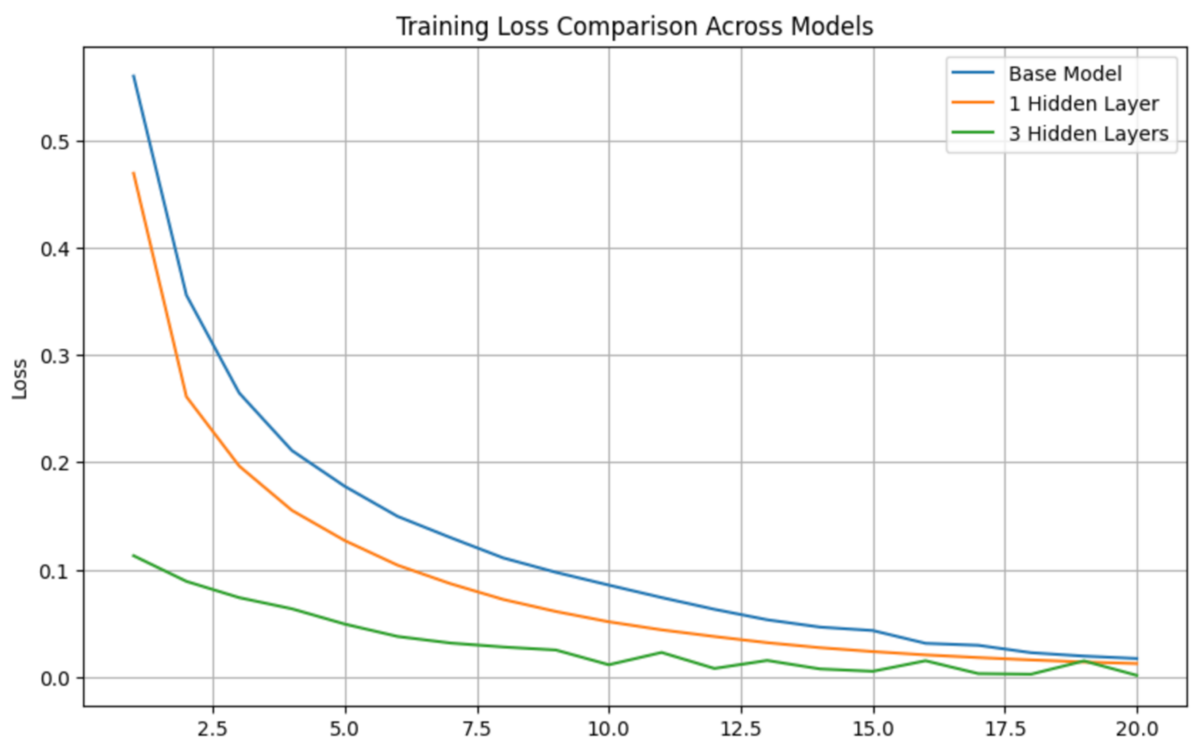


Validation Accuracy Comparison Across Models
Base Model
1 Hidden Layer
3 Hidden Layers
Val_accuracy
Epochs
0.96
0.94
0.92
0.90
0.88
0.86
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0



Validation Loss Comparison Across Models
Base Model
1 Hidden Layer
3 Hidden Layers
Val_loss
Epochs
0.6
0.5
0.4
0.3
0.2
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0


Training Accuracy Comparison Across Models
Accuracy
Epochs
1.00
0.95
0.90
0.85
0.80
0.75
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
Base Model
1 Hidden Layer
3 Hidden Layers


Training Loss Comparison Across Models
Loss
0.5
0.4
0.3
0.2
0.1
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
Base Model
1 Hidden Layer
3 Hidden Layers

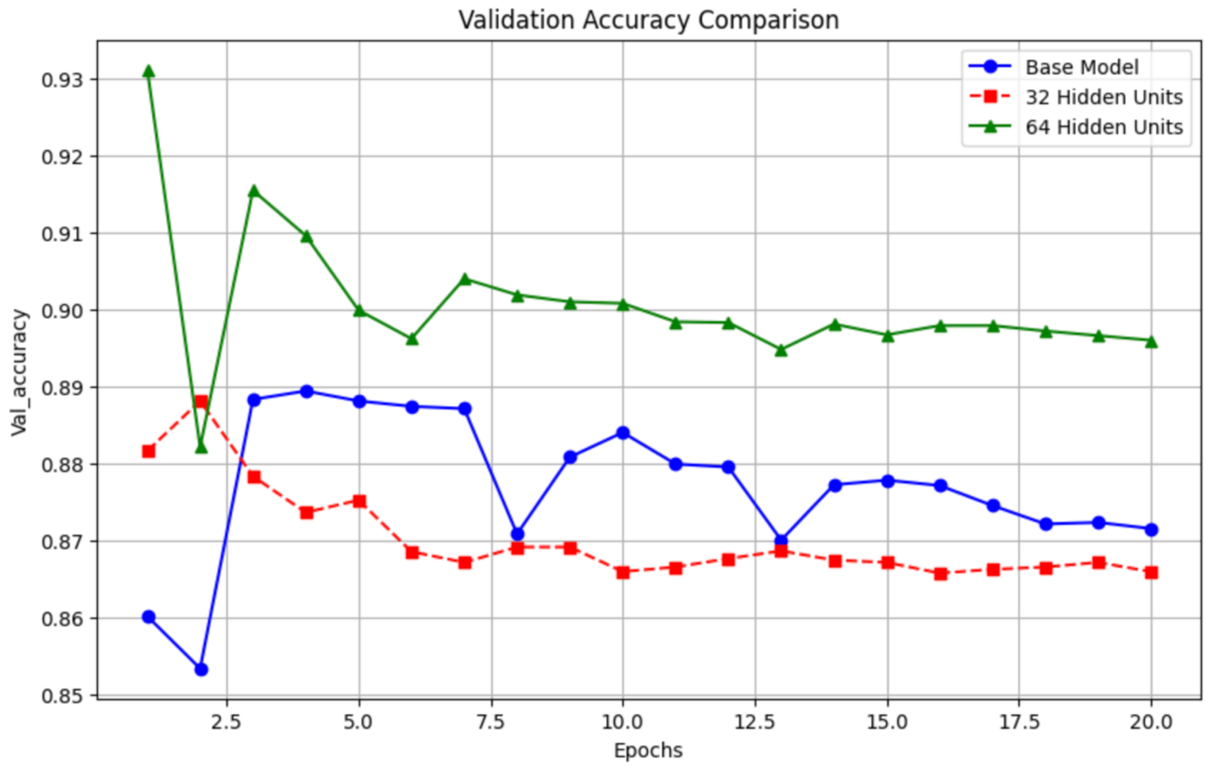
Experiment 2: Changing the Number of Hidden Units

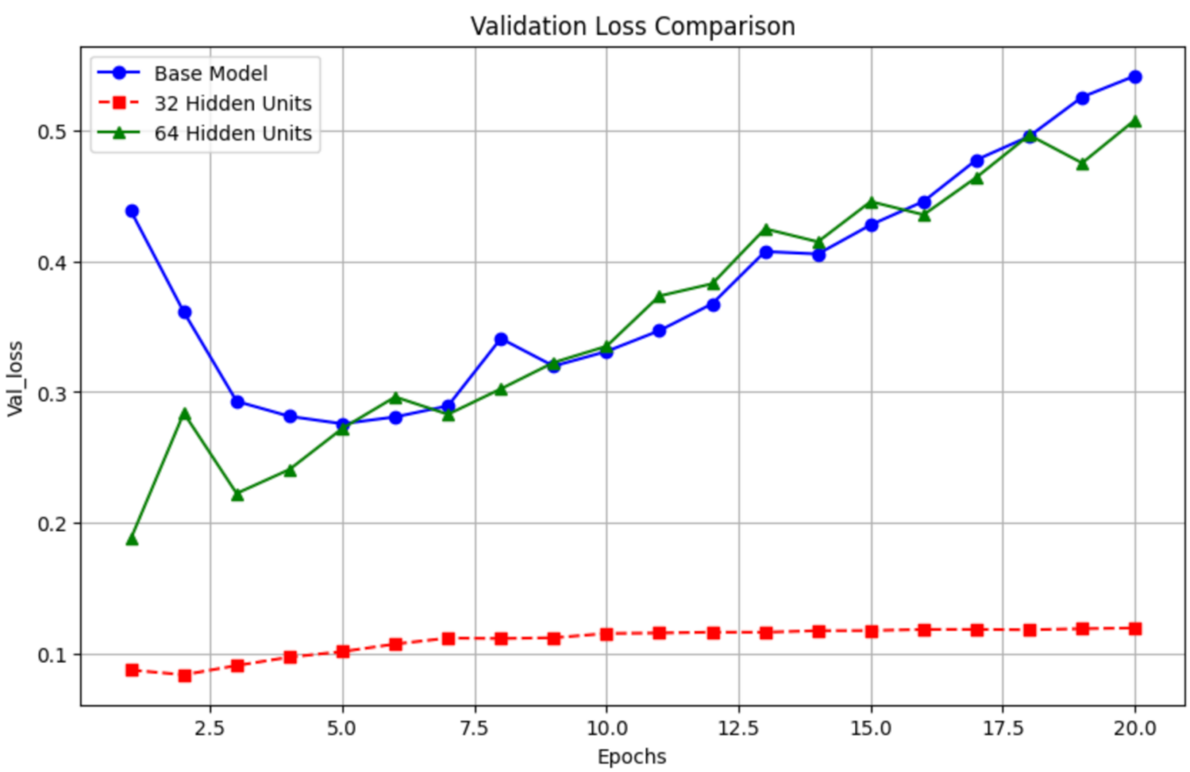
- When considering validation accuracy, the model with 32 hidden units surpasses both the 16-unit and 64-unit models. While the 64-unit model shows more variability and a lower validation accuracy, the 32-unit model performs similarly to the base 16-unit model.
- Although the 64-unit model achieves the lowest validation loss at later epochs, it exhibits signs of overfitting, indicating that increasing the number of hidden units alone does not necessarily enhance the model's ability to generalize well to unseen data.
- Training accuracy and loss remain relatively consistent across the 16, 32, and 64-unit models, with the base model showing slightly better overall performance.
- Increasing the number of hidden units from 16 to 32 to 64 does not lead to significant performance gains. Instead, it introduces greater volatility, which makes the base model, with its simpler architecture, a more stable and reliable choice.

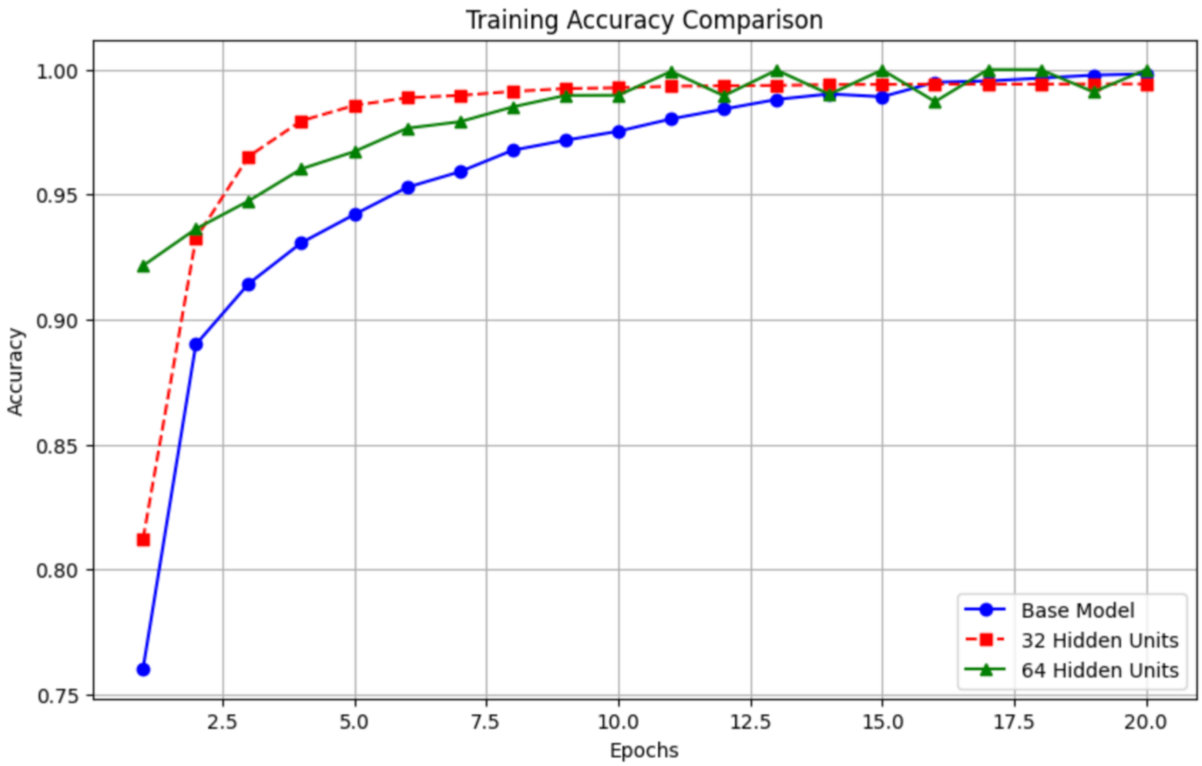
The evaluation of different model architectures, based on varying numbers of hidden units, highlights that the 32-hidden-unit model offers the best validation accuracy compared to the 16-unit and 64-unit models. This suggests that a moderate increase in hidden units can improve performance, but excessive complexity, as seen in the 64-unit model, leads to greater variability and a decline in validation accuracy. Notably, while the 32-unit model's performance is similar to the 16-unit base model, it does show an advantage in validation accuracy, making it a more reliable choice in this context.

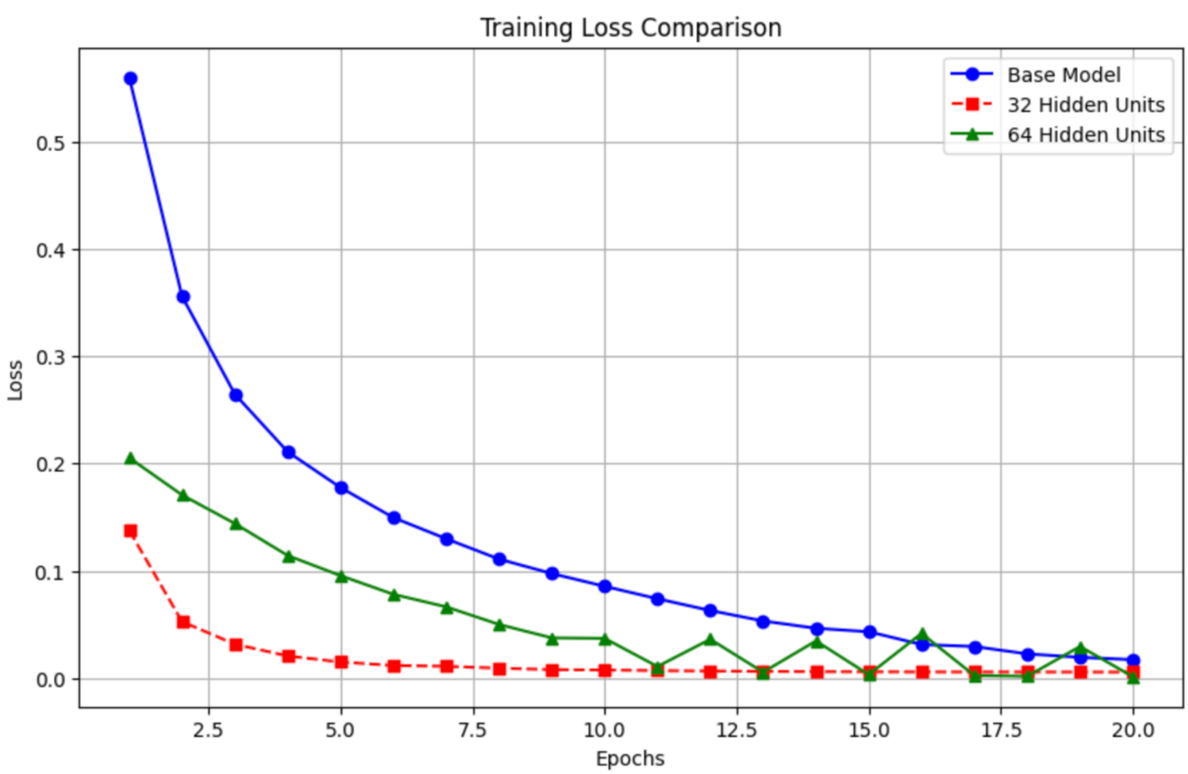
On the other hand, the 64-unit model, despite achieving the lowest validation loss in later epochs, exhibits signs of overfitting. This means that the model is likely memorizing the training data rather than learning generalizable patterns, which is a common issue when a model becomes too complex. This observation reinforces the idea that simply increasing the number of hidden units does not guarantee better generalization or improved results on unseen data. It also highlights the importance of balancing model complexity to prevent overfitting and ensure the model’s ability to generalize well.

Training accuracy and loss across the models with 16, 32, and 64 units remain largely similar, with only minor differences between them. However, the base model, with its simpler configuration, performs slightly better overall. This suggests that a less complex model can still perform well and may be more stable and easier to train.


Validation Accuracy Comparison
Base Model
32 Hidden Units
64 Hidden Units
Val_accuracy
Epochs
0.93
0.92
0.91
0.90
0.89
0.88
0.87
0.86
0.85
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0


Validation Loss Comparison
Base Model
32 Hidden Units
64 Hidden Units
Val_loss
Epochs
0.5
0.4
0.3
0.2
0.1
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0


Training Accuracy Comparison
Accuracy
Epochs
Base Model
32 Hidden Units
64 Hidden Units


Training Loss Comparison
Loss
Epochs
Base Model
32 Hidden Units
64 Hidden Units

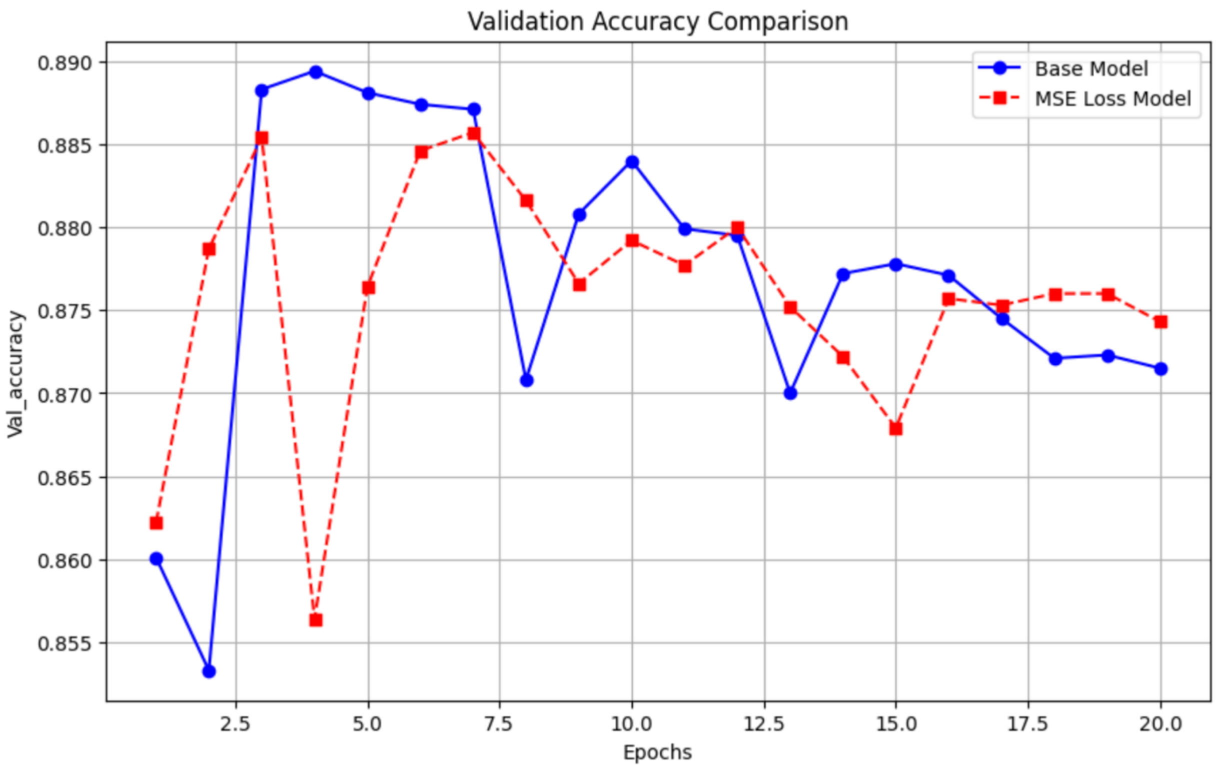
Experiment 3: Changing the Loss Function

- Although Binary Cross-Entropy (BCE) is typically preferred for binary classification problems, in this case, Mean Squared Error (MSE) significantly outperforms BCE.
- MSE shows a clear advantage in validation loss, where it achieves a lower validation loss compared to the base model. This indicates that the MSE model is more effective at minimizing the error between predicted and actual values.
- The MSE model also delivers better results with fewer training epochs, highlighting its efficiency in learning the patterns in the data.
- Overall, the MSE model proves to be a more effective choice than the BCE model for building this specific model.

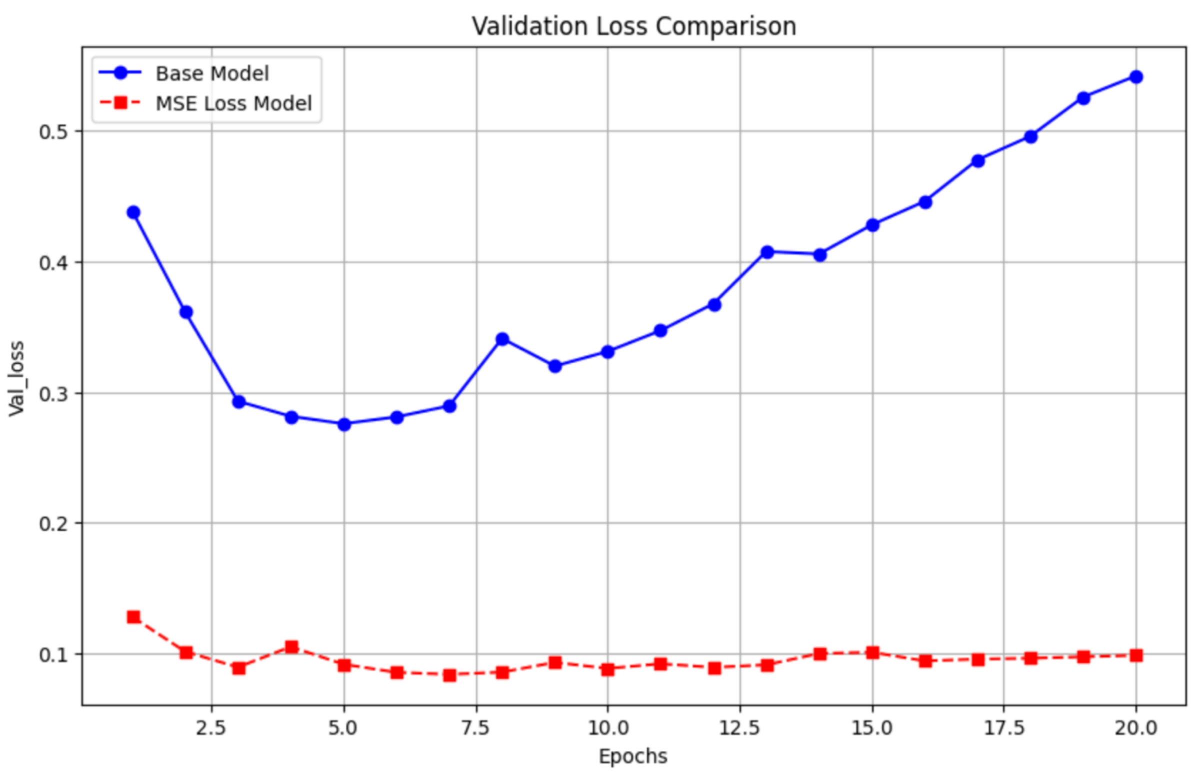
In this analysis, while Binary Cross-Entropy (BCE) is generally the preferred loss function for binary classification tasks, the Mean Squared Error (MSE) approach shows superior performance in this particular case. Despite BCE's widespread use in binary classification, the MSE model demonstrates a clear edge in both validation loss and overall accuracy. Specifically, the MSE model achieves a lower validation loss, meaning it is better at approximating the true values and minimizing prediction errors. This suggests that the MSE loss function helps the model generalize more effectively compared to BCE.

Moreover, the MSE model not only outperforms the BCE model in terms of accuracy and validation loss, but it also does so with fewer epochs of training. This efficiency in training implies that the MSE model requires less computational time to achieve similar or better results, which is a significant advantage when resources or time are limited.

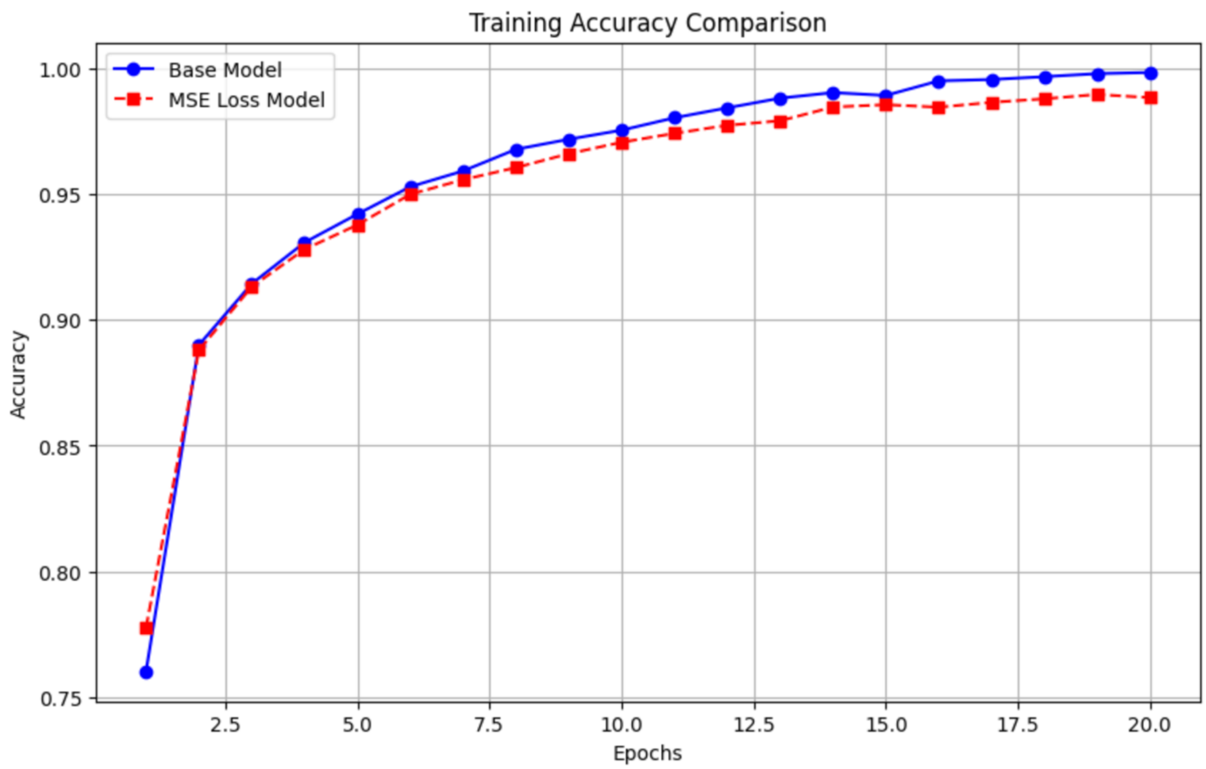
Thus, for this particular model-building task, the results indicate that the MSE loss function is a more optimal choice than BCE. The ability of MSE to provide lower validation loss with less training time makes it a more reliable and efficient option for this scenario.



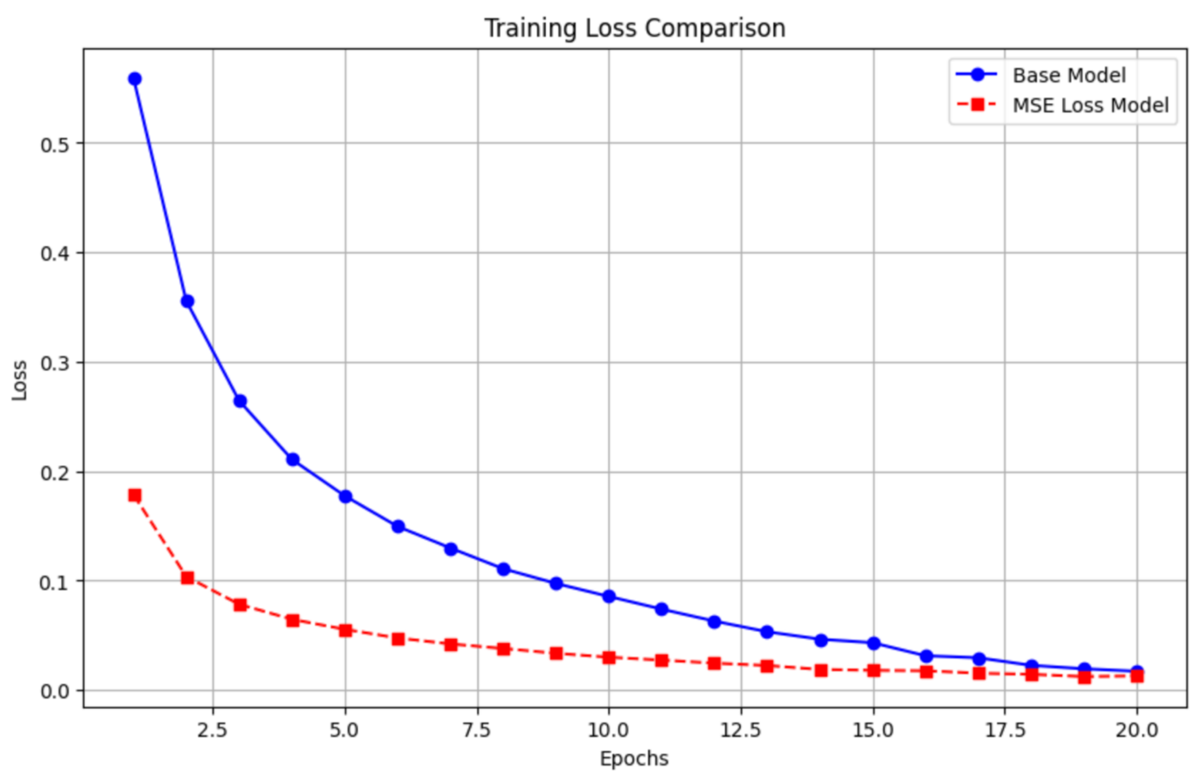
Validation Accuracy Comparison
Base Model
MSE Loss Model
Val_accuracy
Epochs
0.890
0.885
0.880
0.875
0.870
0.865
0.860
0.855
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0



Validation Loss Comparison
Base Model
MSE Loss Model
Val_loss
Epochs
0.5
0.4
0.3
0.2
0.1
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0



Training Accuracy Comparison
Base Model
MSE Loss Model
Accuracy
Epochs
1.00
0.95
0.90
0.85
0.80
0.75
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0



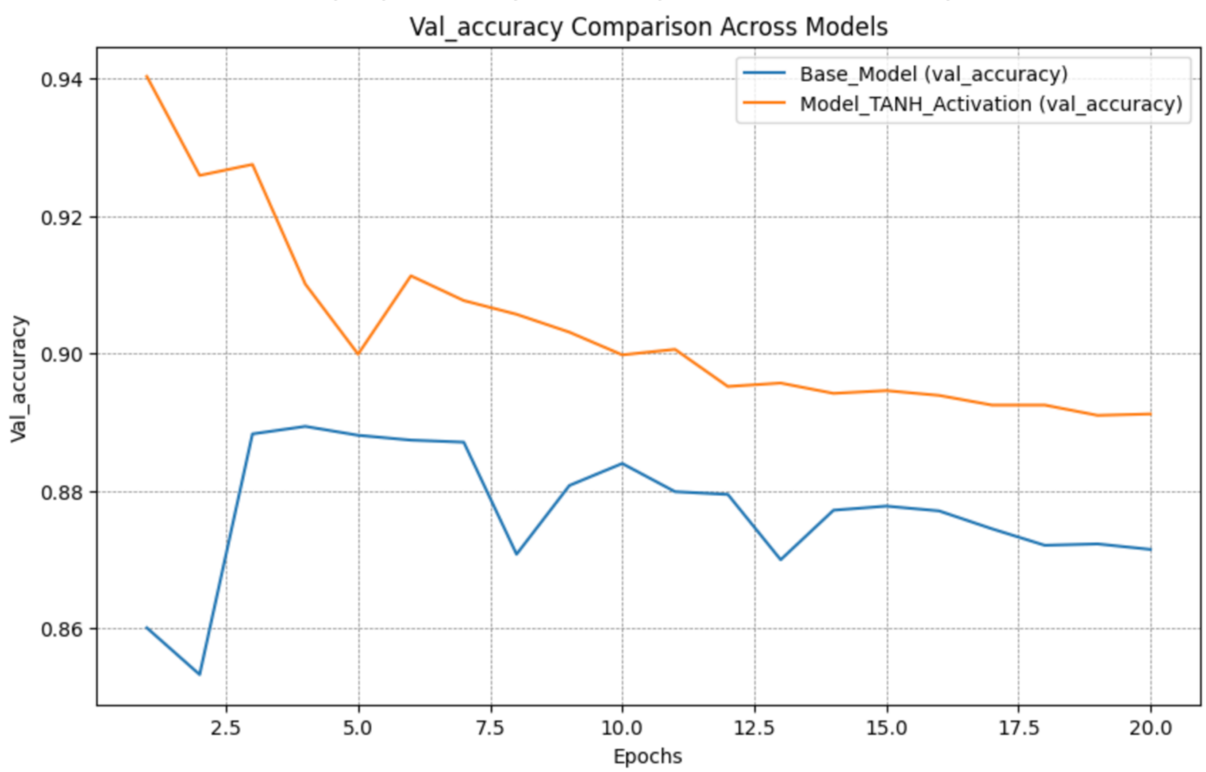
Training Loss Comparison
Base Model
MSE Loss Model
Loss
Epochs
0.5
0.4
0.3
0.2
0.1
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0

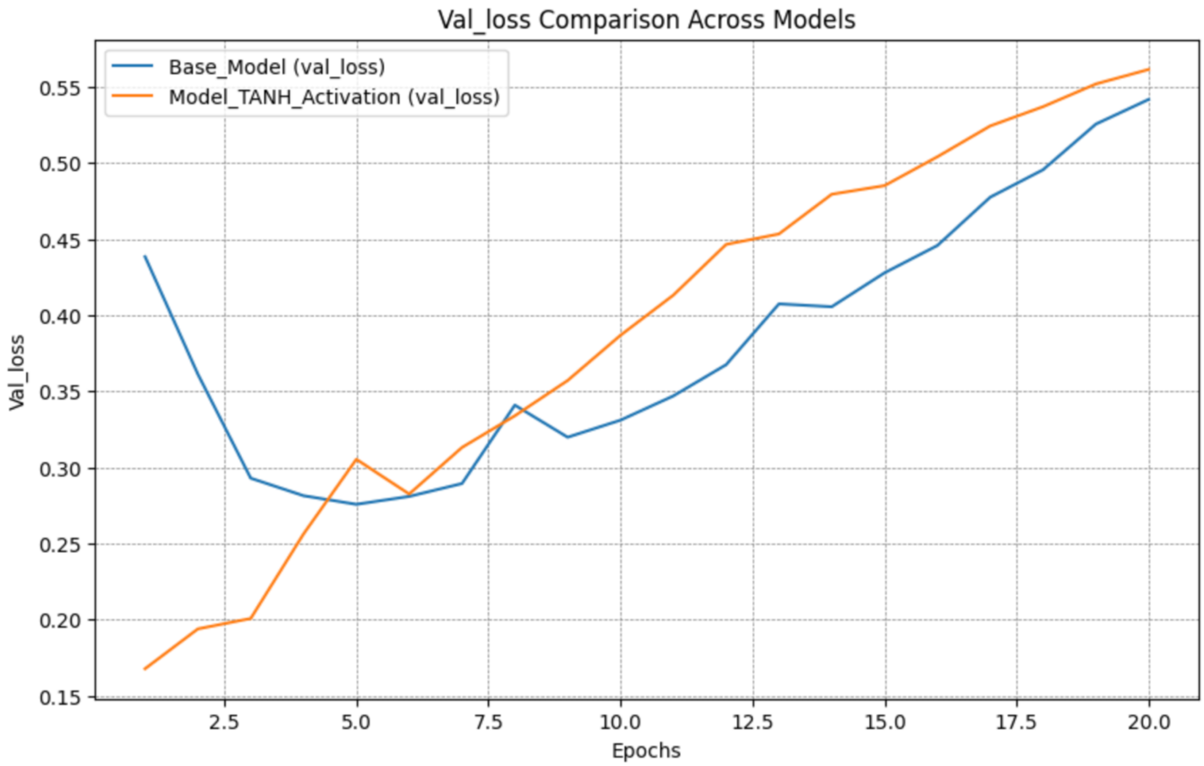
Experiment 4: Changing the Activation Function

- The ReLU activation function (used in the base model) outperforms the Tanh activation function, as it achieves higher accuracy.
- The ReLU model also demonstrates a lower loss compared to the Tanh model, indicating that ReLU is more effective in minimizing the error between predicted and actual values.
- Overall, ReLU is a superior choice, as it provides better accuracy and a lower loss compared to Tanh.

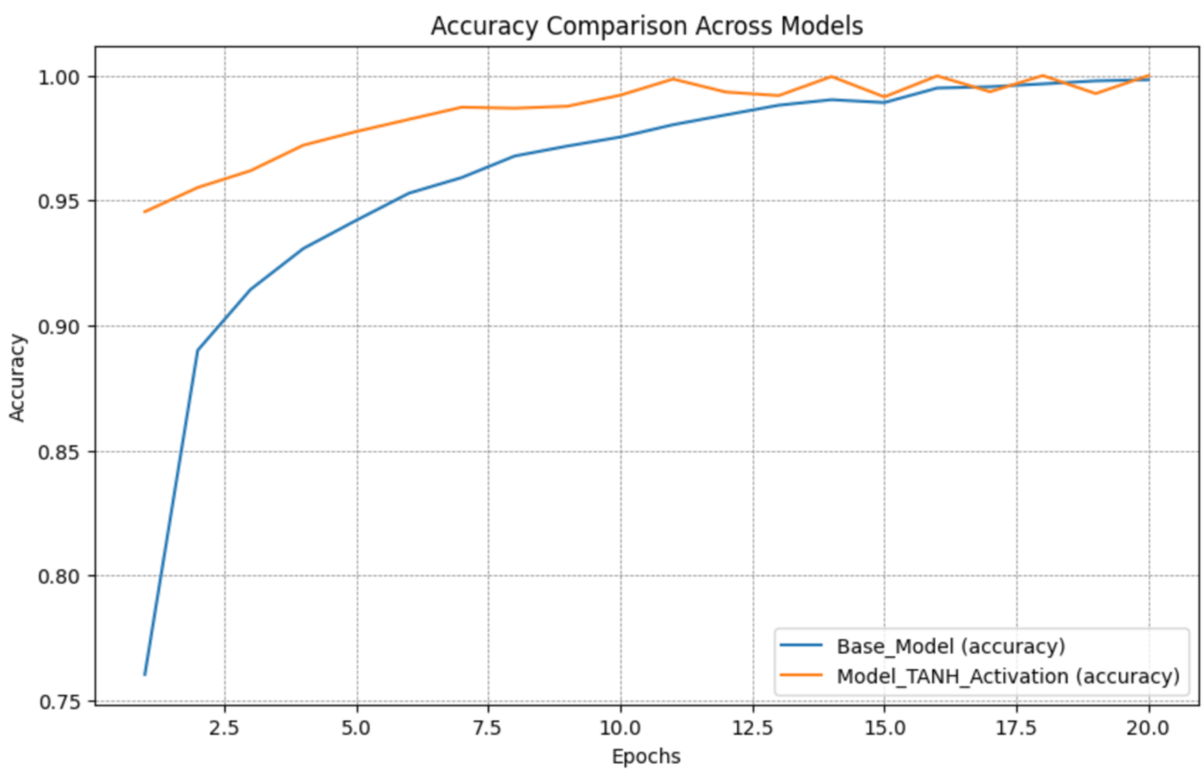
In this analysis, the ReLU (Rectified Linear Unit) activation function used in the baseline model proves to be more effective than the Tanh activation function. The ReLU model achieves higher accuracy, which indicates it performs better in making correct predictions on both training and validation data. Additionally, the ReLU model shows a lower loss compared to the Tanh model, suggesting that it is better at minimizing the difference between predicted and true values. This is a crucial factor in ensuring that the model is learning the correct patterns and generalizing well to unseen data.

ReLU activation functions are known for their ability to avoid some of the issues that Tanh activation can face, such as vanishing gradients, which often hinder learning in deeper networks. The ReLU model's ability to achieve both higher accuracy and lower loss highlights its effectiveness in optimizing model performance. Therefore, when comparing the two, ReLU emerges as the better option due to its superior ability to minimize errors and improve overall model performance.

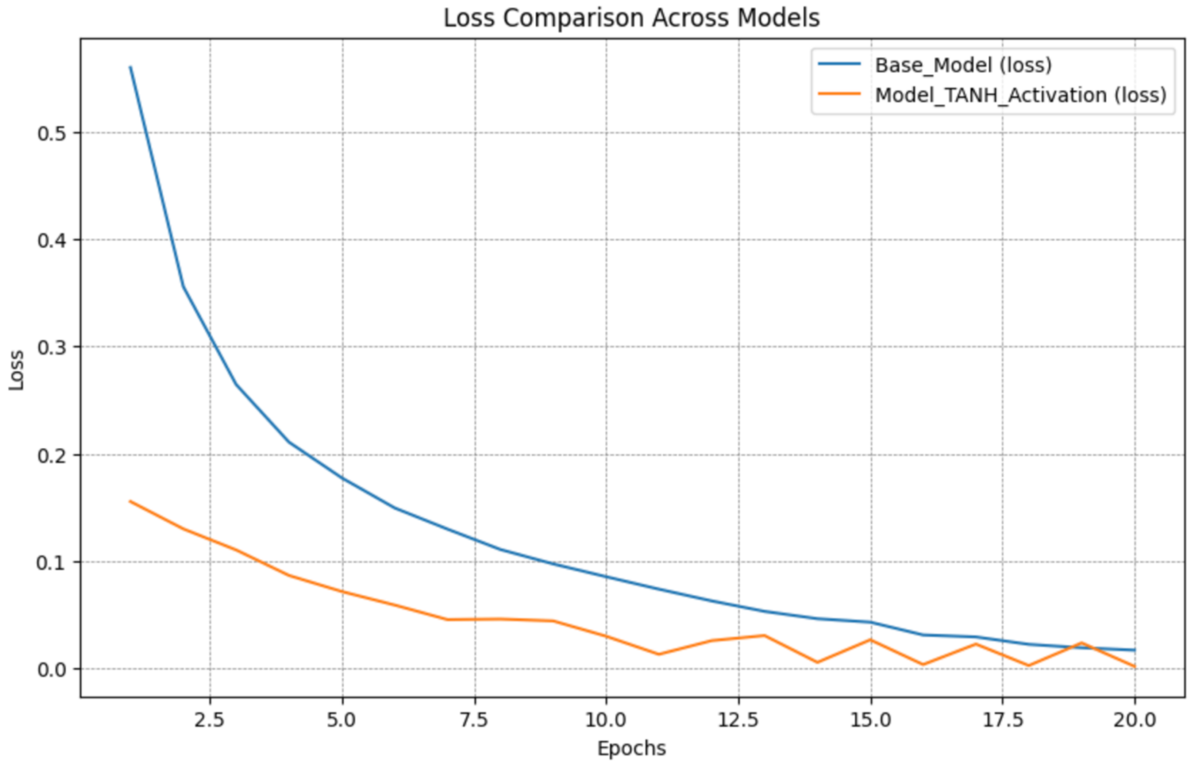




Val_loss Comparison Across Models
Base_Model (val_loss)
Model_TANH_Activation (val_loss)
Val_loss
Epochs
0.15
0.20
0.25
0.30
0.35
0.40
0.45
0.50
0.55
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0



Accuracy Comparison Across Models
Base_Model (accuracy)
Model_TANH_Activation (accuracy)
Accuracy
Epochs
0.75
0.80
0.85
0.90
0.95
1.00
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0

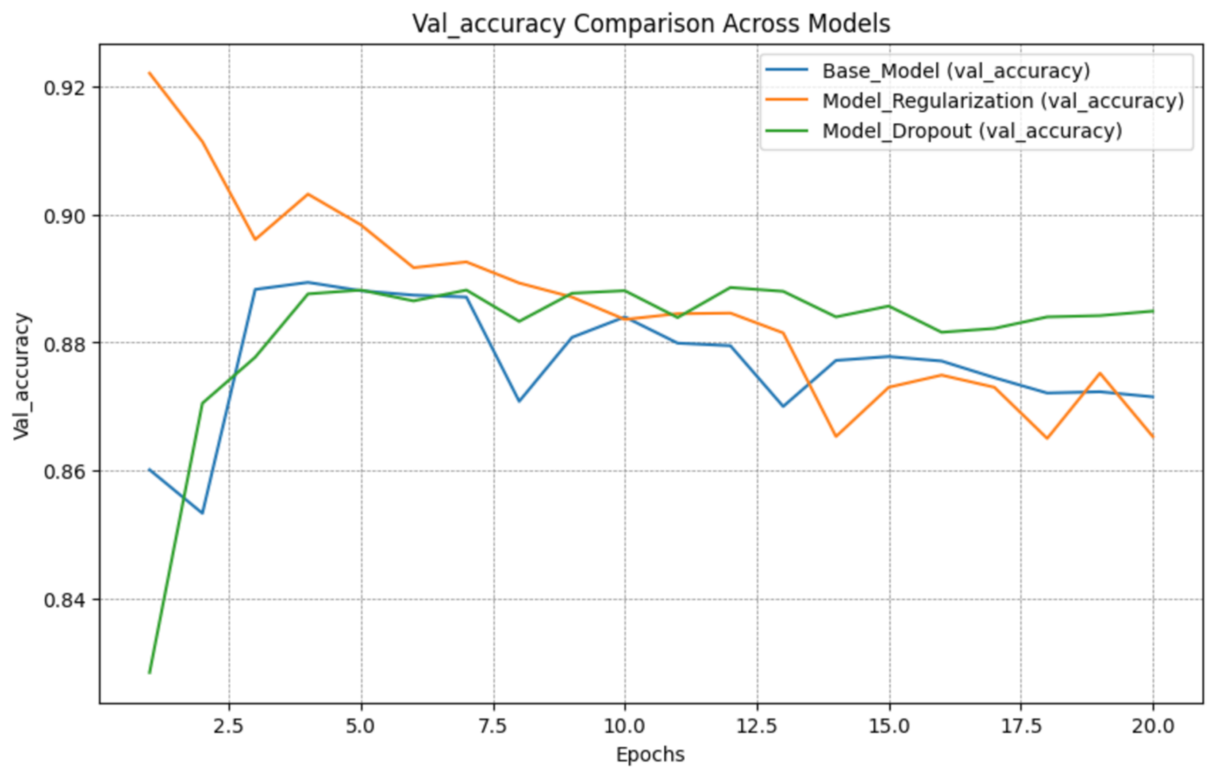


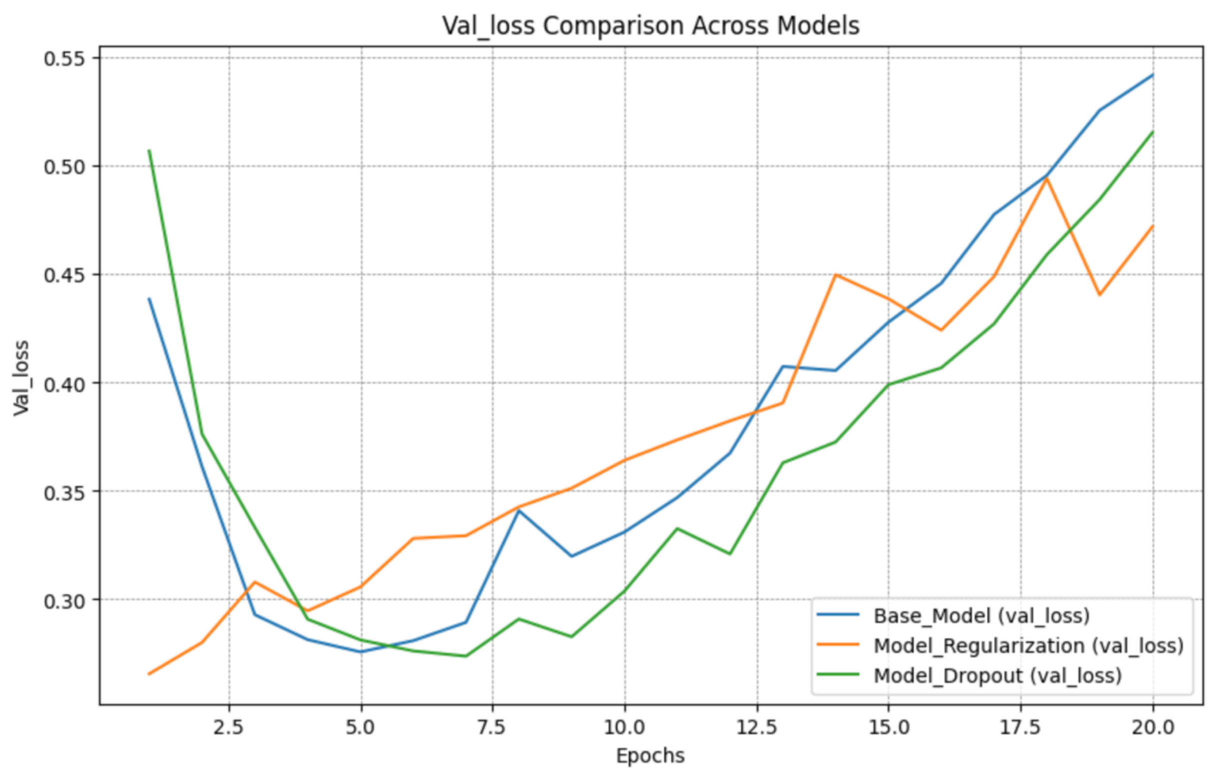
Experiment 5: Implementing Regularization Techniques

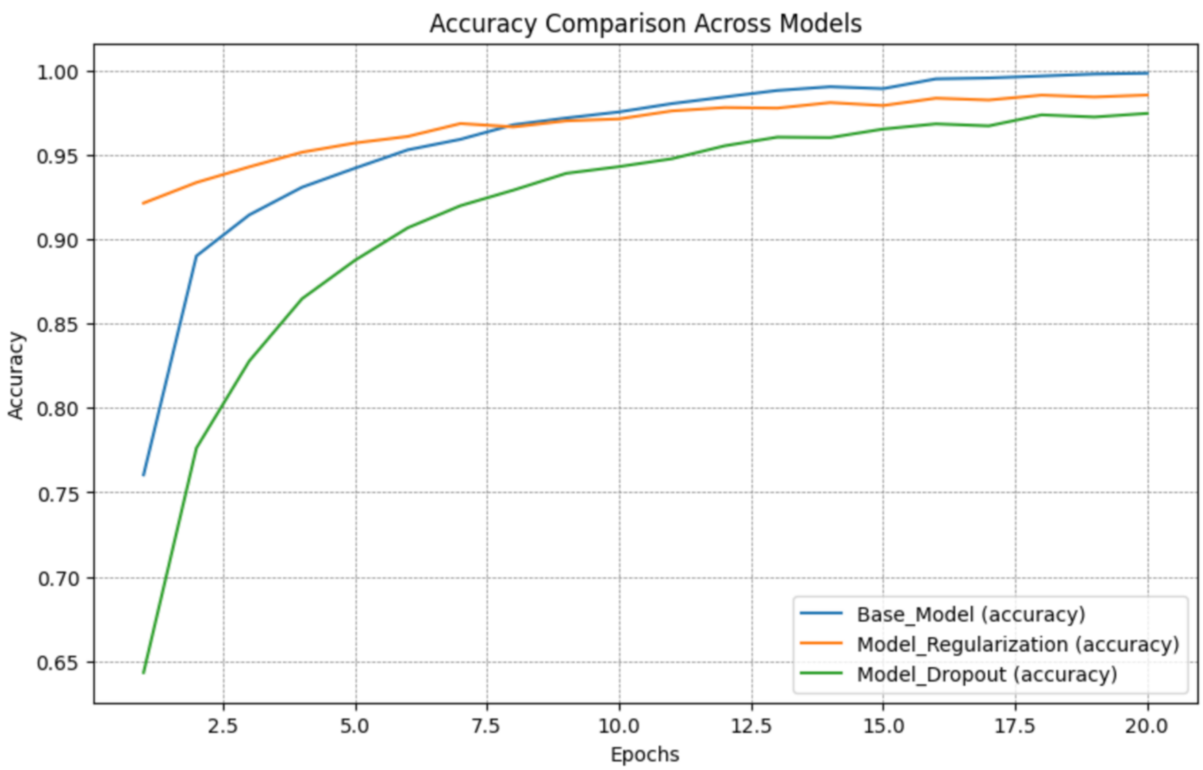
- Dropout optimization achieves the highest accuracy when compared to both the baseline model and L2 regularization.
- It also leads the models in terms of loss, showing the lowest error rate, with L2 regularization following closely behind, while the baseline model is the least effective.
- Overall, considering the performance across all metrics, dropout optimization emerges as the most effective method for improving this model.

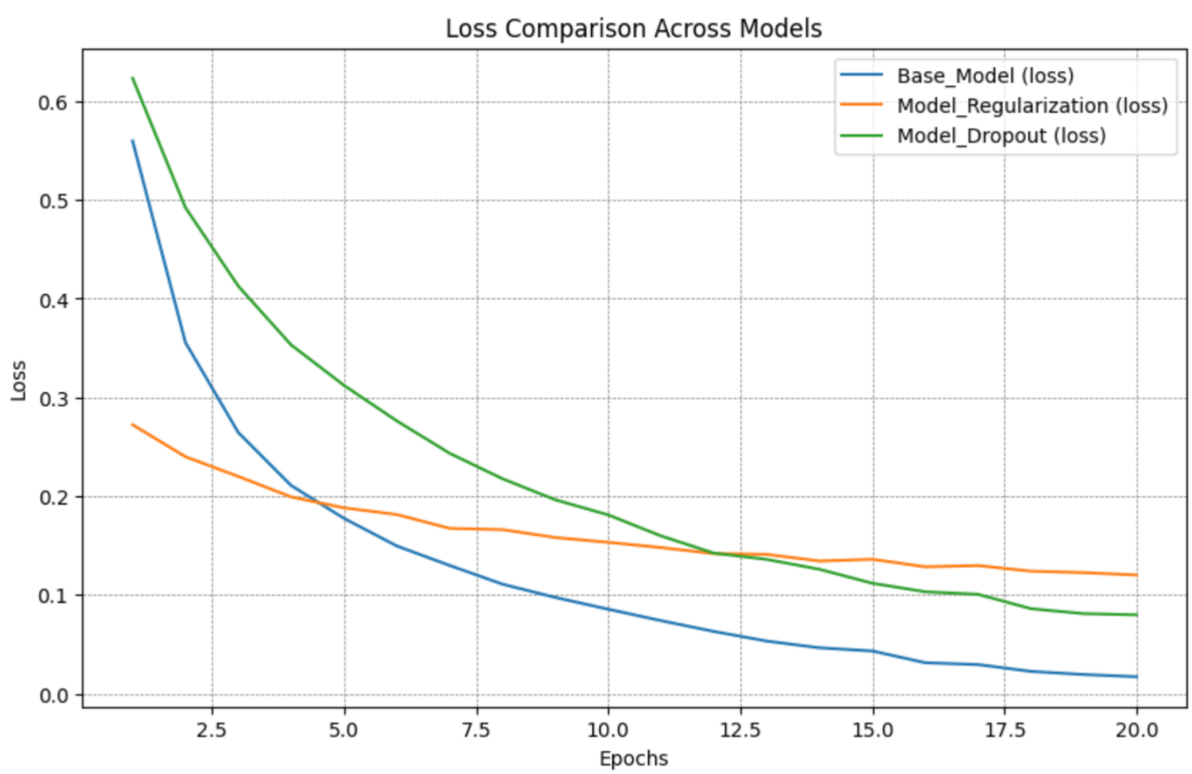
In this analysis, dropout optimization stands out as the top-performing method, outperforming both the baseline model and L2 regularization in terms of accuracy. Dropout helps prevent overfitting by randomly "dropping" units from the neural network during training, which forces the model to generalize better. As a result, it achieves the highest accuracy across the board, making it the most effective technique for enhancing model performance.

In terms of loss, dropout also performs the best, showing the lowest error rate compared to the other methods. This indicates that the model with dropout is better at minimizing the difference between predicted and actual values, which is critical for generalization. L2 regularization, while still effective, follows closely behind in both accuracy and loss, and the baseline model comes last in terms of both metrics.


Val_accuracy Comparison Across Models
Base_Model (val_accuracy)
Model_Regularization (val_accuracy)
Model_Dropout (val_accuracy)
Val_accuracy
0.92
0.90
0.88
0.86
0.84
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
Epochs


Val_loss Comparison Across Models
Base_Model (val_loss)
Model_Regularization (val_loss)
Model_Dropout (val_loss)
Val_loss
0.55
0.50
0.45
0.40
0.35
0.30
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
Epochs


Accuracy Comparison Across Models
Accuracy
Epochs
Base_Model (accuracy)
Model_Regularization (accuracy)
Model_Dropout (accuracy)
1.00
0.95
0.90
0.85
0.80
0.75
0.70
0.65
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0


Loss Comparison Across Models
Loss
Epochs
Base_Model (loss)
Model_Regularization (loss)
Model_Dropout (loss)
0.6
0.5
0.4
0.3
0.2
0.1
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0

Results & Observations (compare all models)

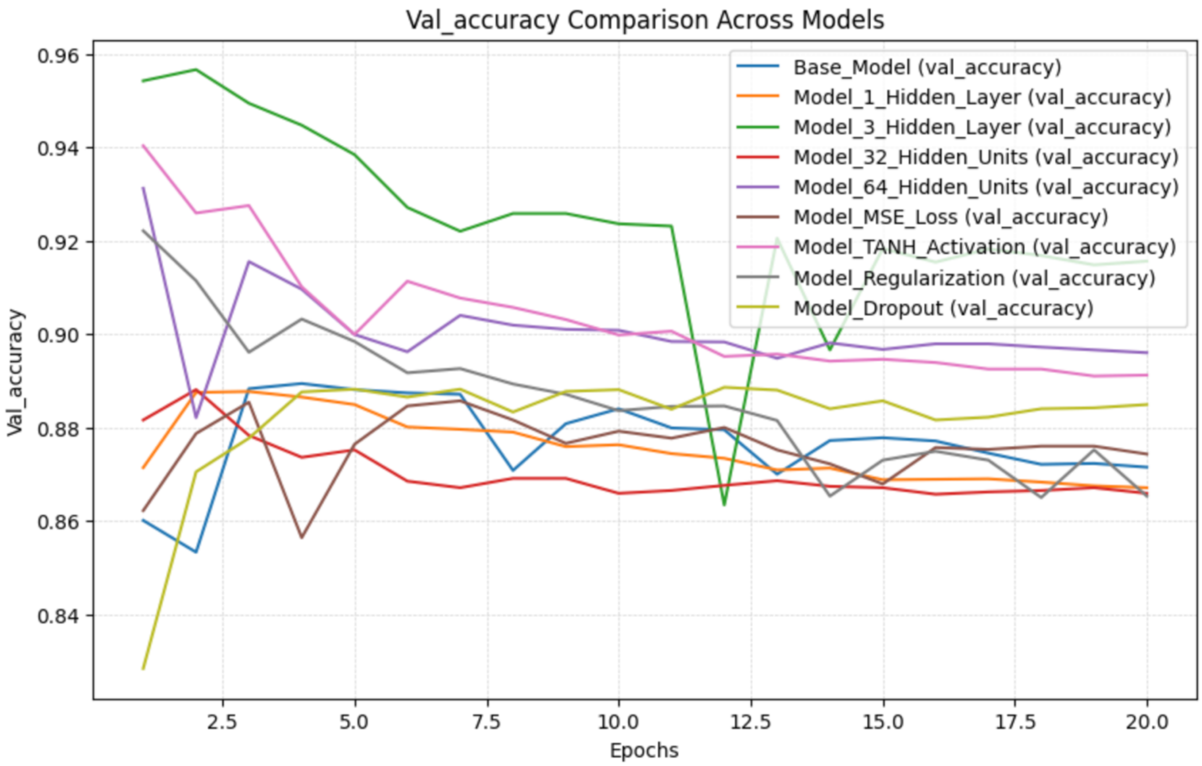
- Among all the models, the one with 32 hidden units achieved the highest validation accuracy.
- In terms of overall accuracy, the Tanh activation model and the baseline model performed similarly, making them the top performers in this comparison.
- When evaluating validation loss, the model with 64 hidden units showed the best results.
- Overall, the model with 64 hidden units demonstrated a good balance, delivering consistently strong performance across all metrics.

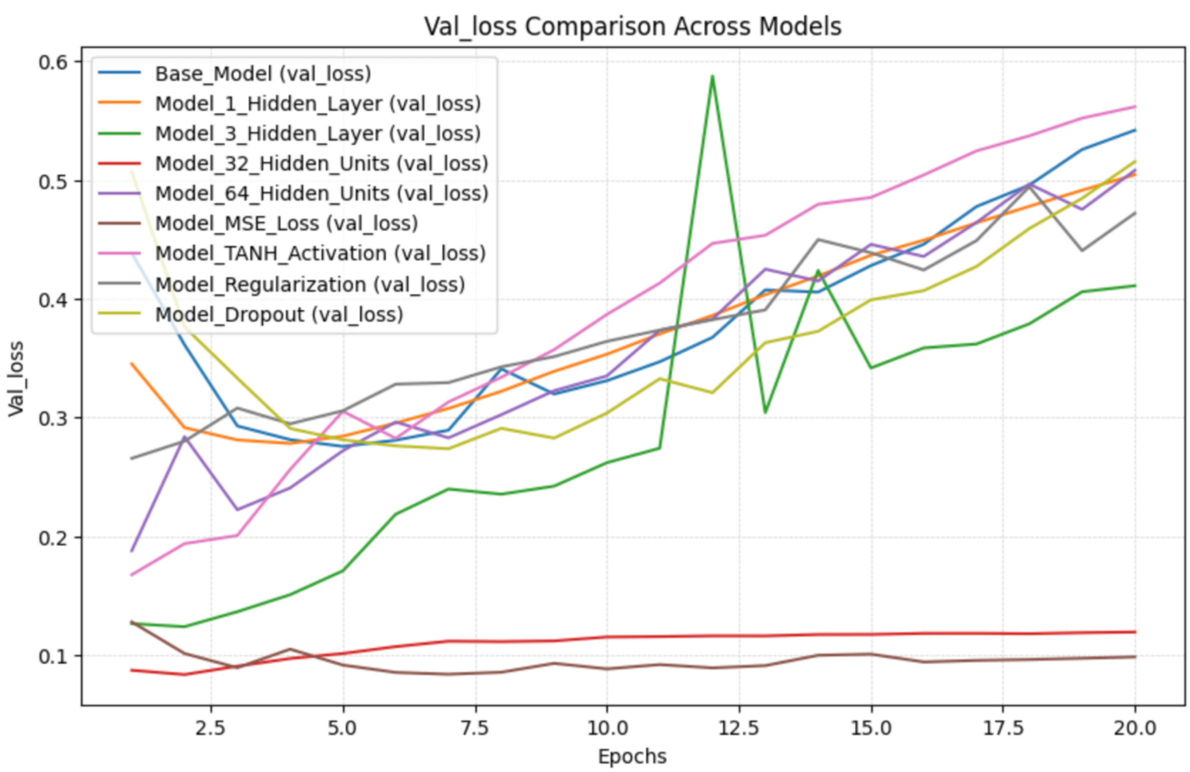
In this comparison of different model architectures, the model with 32 hidden units stands out by achieving the highest validation accuracy. This indicates that it was the most effective at correctly predicting unseen data during validation, suggesting that 32 hidden units might provide an optimal number for capturing patterns in the dataset.

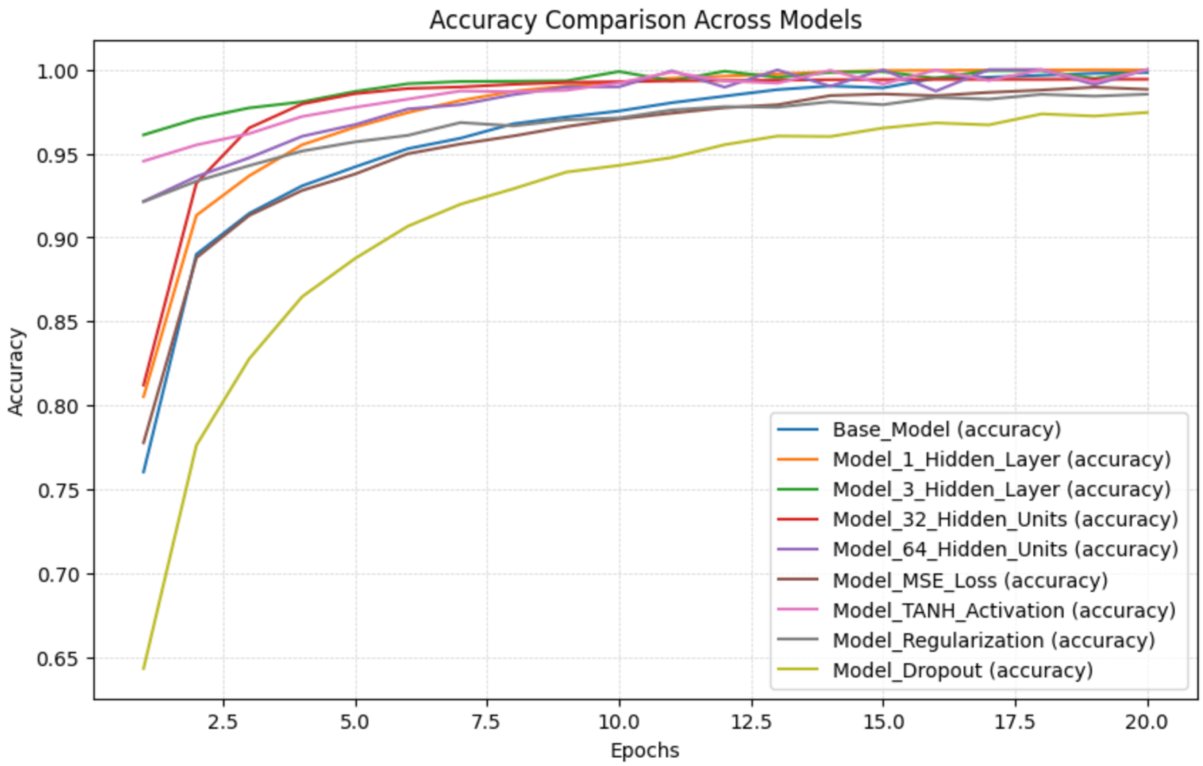
When evaluating overall accuracy, both the Tanh activation model and the baseline model performed similarly, both showing strong results. This suggests that the choice of activation function may not significantly impact overall accuracy in this case, as both models exhibit high performance.

In terms of validation loss, which reflects the model's ability to minimize the error between predicted and true values, the model with 64 hidden units performed the best. This highlights its capacity to generalize well and avoid overfitting, as lower validation loss is a good indicator of model effectiveness.

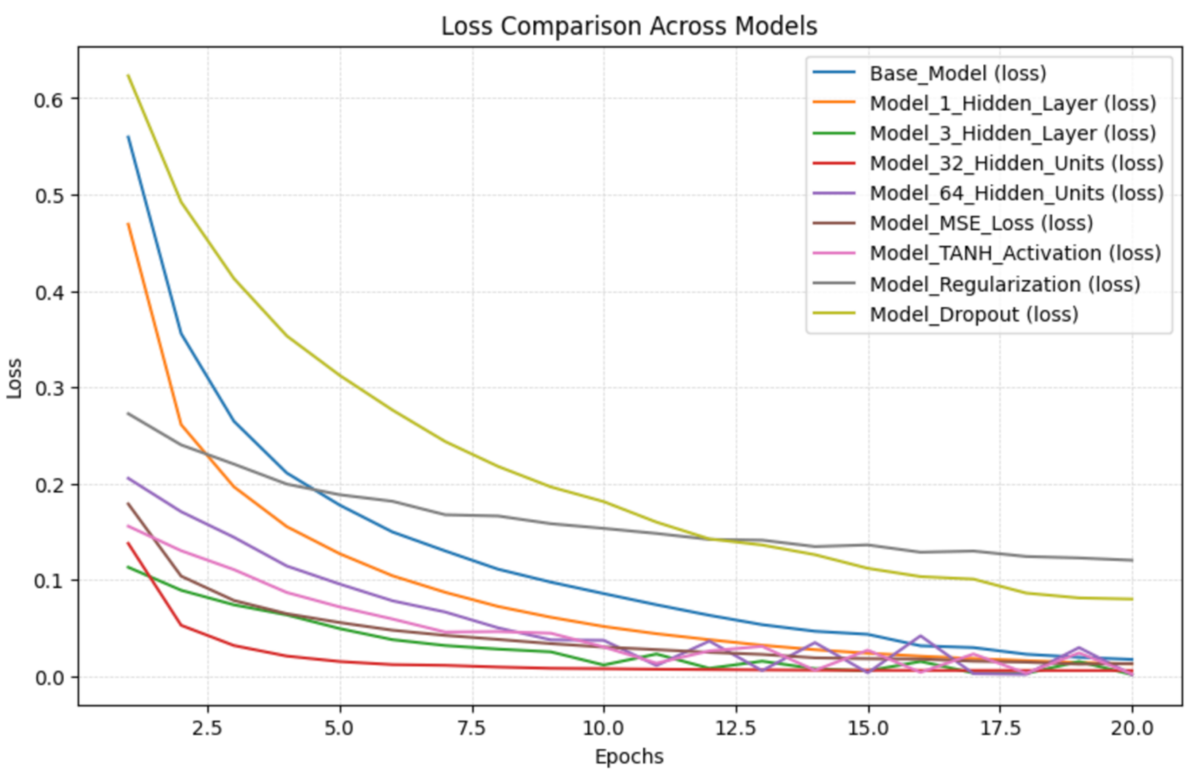
Ultimately, the model with 64 hidden units provided a solid balance between accuracy and loss, performing well across all metrics. It suggests that the additional complexity of 64 hidden units might provide a robust learning capacity, yielding stable and consistent performance throughout the training process.


Val_accuracy Comparison Across Models
Base_Model (val_accuracy)
Model_1_Hidden_Layer (val_accuracy)
Model_3_Hidden_Layer (val_accuracy)
Model_32_Hidden_Units (val_accuracy)
Model_64_Hidden_Units (val_accuracy)
Model_MSE_Loss (val_accuracy)
Model_TANH_Activation (val_accuracy)
Model_Regularization (val_accuracy)
Model_Dropout (val_accuracy)
Val_accuracy
Epochs


Val_loss Comparison Across Models
Base_Model (val_loss)
Model_1_Hidden_Layer (val_loss)
Model_3_Hidden_Layer (val_loss)
Model_32_Hidden_Units (val_loss)
Model_64_Hidden_Units (val_loss)
Model_MSE_Loss (val_loss)
Model_TANH_Activation (val_loss)
Model_Regularization (val_loss)
Model_Dropout (val_loss)
Val_loss
Epochs



Accuracy Comparison Across Models
Accuracy
Epochs
1.00
0.95
0.90
0.85
0.80
0.75
0.70
0.65
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
Base_Model (accuracy)
Model_1_Hidden_Layer (accuracy)
Model_3_Hidden_Layer (accuracy)
Model_32_Hidden_Units (accuracy)
Model_64_Hidden_Units (accuracy)
Model_MSE_Loss (accuracy)
Model_TANH_Activation (accuracy)
Model_Regularization (accuracy)
Model_Dropout (accuracy)



Loss Comparison Across Models
Loss
Epochs
0.6
0.5
0.4
0.3
0.2
0.1
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
Base_Model (loss)
Model_1_Hidden_Layer (loss)
Model_3_Hidden_Layer (loss)
Model_32_Hidden_Units (loss)
Model_64_Hidden_Units (loss)
Model_MSE_Loss (loss)
Model_TANH_Activation (loss)
Model_Regularization (loss)
Model_Dropout (loss)

Conclusion

The analysis of various model architectures, activation functions, loss functions, and optimization techniques revealed key insights into performance. The model with 32 hidden units achieved the highest validation accuracy, indicating that an optimal number of hidden units significantly enhances performance. The model with 64 hidden units, on the other hand, showed the best validation loss, suggesting that larger architectures can improve error minimization, but they also introduce more variability. In terms of activation functions, ReLU, used in the baseline model, outperformed Tanh in both accuracy and loss, making it the superior choice for this task. Regarding loss functions, the Mean Squared Error (MSE) loss function proved to be more effective than Binary Cross-Entropy (BCE), offering lower validation loss and quicker convergence. Finally, among regularization techniques, Dropout emerged as the most effective, outperforming both the baseline model and L2 regularization in terms of accuracy and loss. The combination of 32 hidden units, ReLU activation, MSE loss function, and dropout regularization provided the most balanced and effective results, ensuring optimal performance in terms of accuracy, stability, and generalization.