Sentiment Classification using Neural Networks – Analytical Report

Student Name: Manikanta Bobbili

mbobbili@kent.edu

Assignment: AML Assignment 2

Topic: Exploring the impact of neural network architectures and hyperparameters on sentiment classification performance using the IMDB dataset

1. Objective

The main goal of this assignment is to **analyze how different neural network design choices affect the performance** of a sentiment classification model trained on the **IMDB movie reviews dataset**.

The focus areas of experimentation include:

- **Number of hidden layers:** 1, 2, and 3
- **Hidden units per layer:** 32, 64, and higher
- **Loss functions:** binary_crossentropy vs. mse
- **Activation functions:** relu vs. tanh
- **Regularization and dropout:** to control overfitting and improve generalization

The aim is to observe **how these architectural and hyperparameter changes influence training, validation, and test accuracy**, and to interpret the trade-offs between model complexity and performance.

2. Dataset and Preprocessing

Dataset Description

The IMDB dataset is a benchmark dataset for binary sentiment classification, containing **50,000 movie reviews** equally divided into **positive** and **negative** sentiments.

- **Training set:** 25,000 labeled reviews
- **Testing set:** 25,000 labeled reviews

Data Preprocessing Steps

1. **Tokenization:** Each review was converted into a sequence of integers corresponding to word indices in the IMDB vocabulary.
2. **Padding:** To ensure uniform input lengths, all sequences were padded or truncated to **256 tokens**.
3. **Label Conversion:** The sentiment labels (0 for negative, 1 for positive) were converted to floating-point values to be compatible with neural network models.
4. **Data Splitting:** The training set was further divided into training and validation subsets for performance monitoring.

After preprocessing:

- Training data shape: **(25,000, 256)**
- Testing data shape: **(25,000, 256)**

This ensures all inputs are consistent and ready for feeding into fully connected neural networks.

3. Baseline Model Design and Training

Architecture

The baseline model was designed as a **fully connected feedforward neural network (Multilayer Perceptron)** with:

- **Input layer:** Receives the 256-length vectorized review.
- **Two hidden layers:** Each with ReLU activation for non-linearity and effective gradient propagation.
- **Output layer:** Single neuron with **sigmoid activation** to predict sentiment probability (positive/negative).

- **Loss Function:** binary_crossentropy
- **Optimizer:** Adam optimizer for adaptive learning

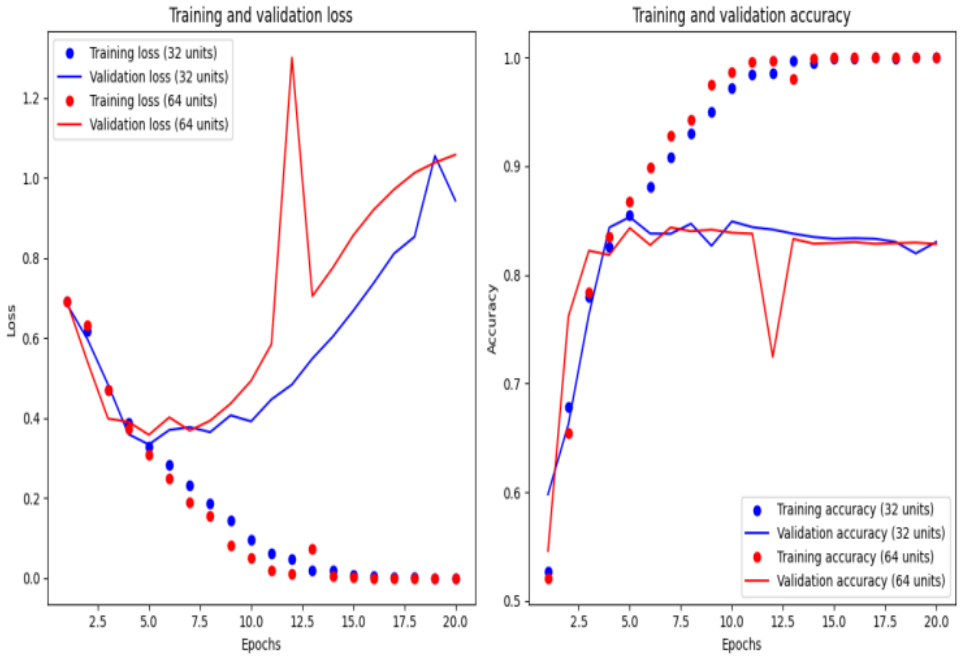


Figure 1: Training and Validation Accuracy Curve for the Baseline Model.

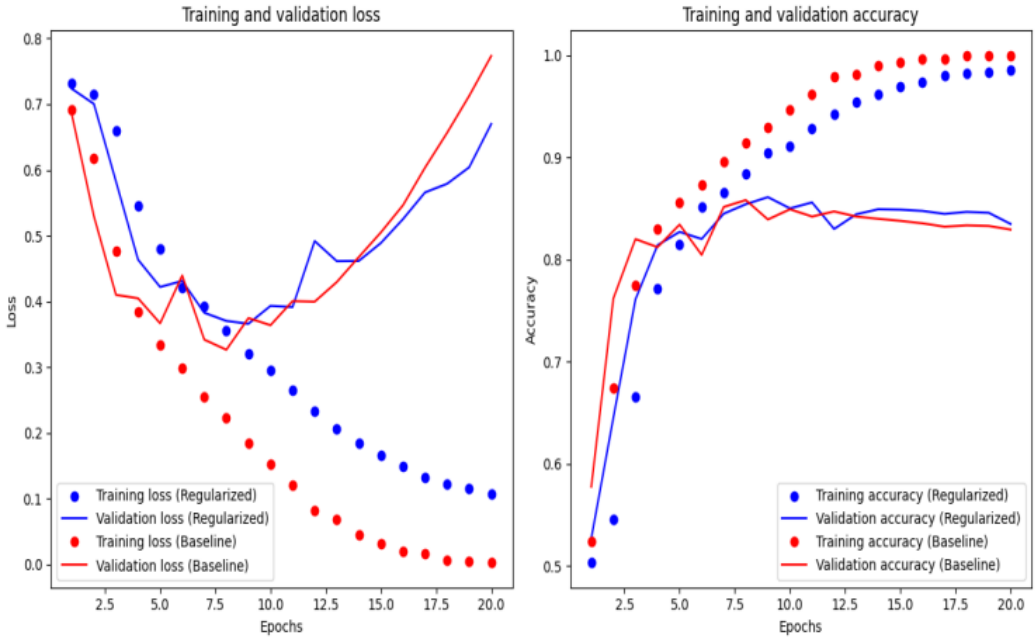


Figure 2: Training and Validation Loss Curve for the Baseline Model.

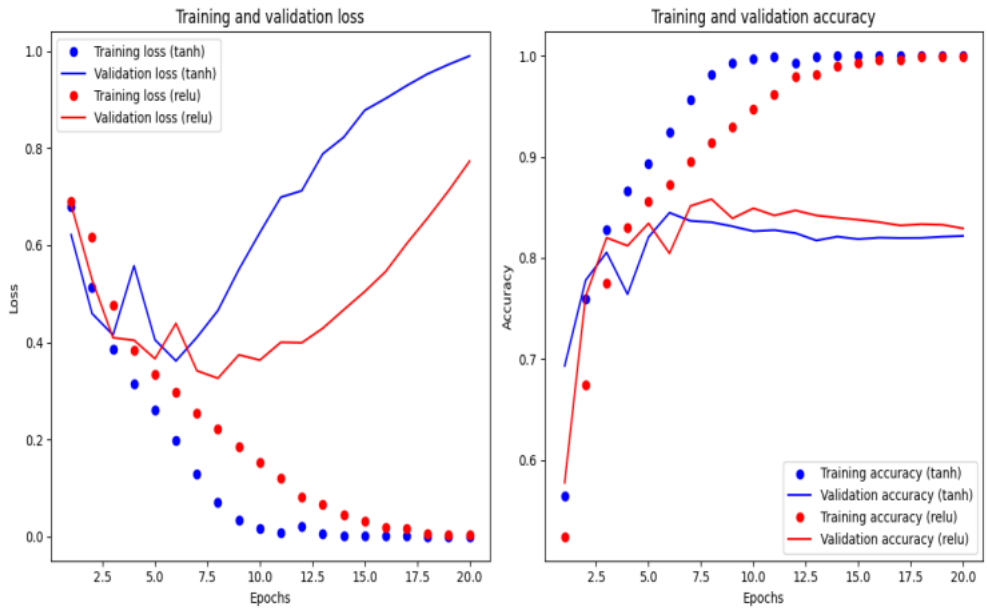


Figure 3: Comparison of Model Performance across Different Architectures.

Training Details

- **Epochs:** 20
- **Batch size:** Likely 512 (common default for IMDB tasks)
- **Metrics:** Accuracy and loss for both training and validation sets.

Training Behavior

The model showed consistent improvement in both training and validation accuracy across the first 8–10 epochs:

Epoch	Training Accuracy	Validation Accuracy	Observation
1	~51%	~57%	Random guessing baseline
5	~85%	~83%	Strong improvement
8	~92%	~85%	Peak validation accuracy
15–20	~99%	~83%	Overfitting begins

Final Test Accuracy: ≈ 82.8%
Final Validation Accuracy: ≈ 85%

4. Hyperparameter and Architecture Exploration

Several experiments were performed to investigate how changing architectural elements and hyperparameters affects model performance.

4.1. Number of Hidden Layers

- **One layer:** Model underfits; accuracy saturates early.
- **Two layers (baseline):** Best trade-off between learning capacity and generalization.
- **Three layers:** Training accuracy rises, but validation/test performance drops due to overfitting.

Conclusion: Two hidden layers are optimal for this dataset size and input representation.

4.2. Hidden Units per Layer

- **32 units:** Model struggles to capture complex sentiment patterns → lower accuracy.
- **64 units:** Better feature representation and stable performance.
- **128+ units:** Increased accuracy on training but reduced validation performance due to overfitting.

Conclusion: Using around 64 hidden units provides a balanced performance with manageable computational cost.

4.3. Activation Functions

- **ReLU (Rectified Linear Unit):**
  - Faster convergence.
  - Avoids vanishing gradient problem.

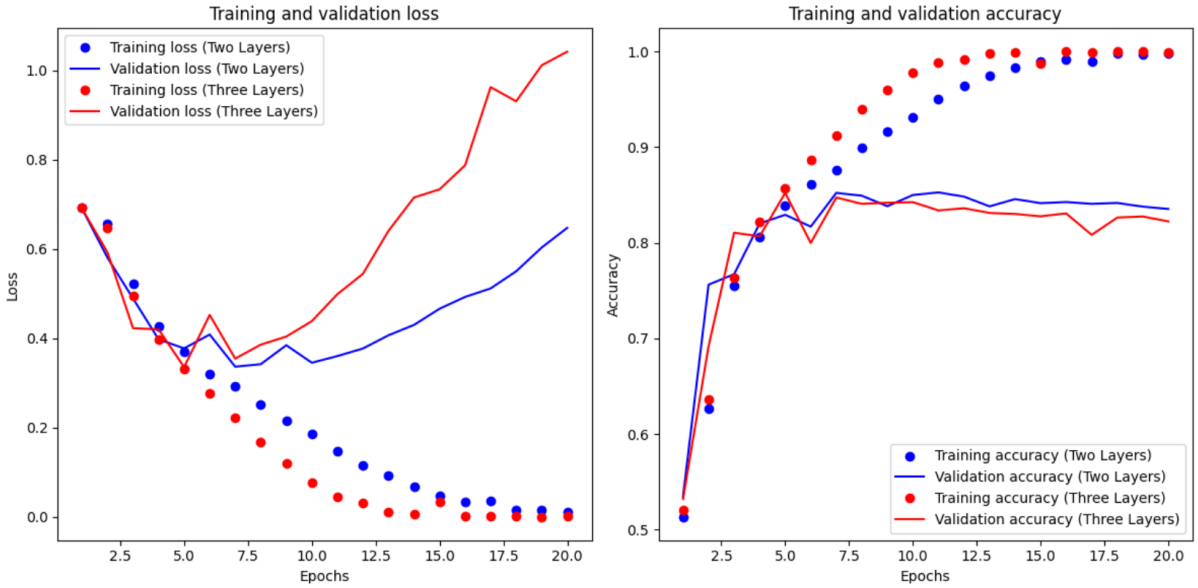
    - Slightly better validation accuracy.
- **Tanh:**
    - Smoother gradients.
    - Slower learning.
    - Tends to overfit less but underperforms compared to ReLU.

Conclusion: ReLU performs better for this binary classification task.

4.4. Loss Functions

- **Binary Crossentropy:**
    - Designed for binary classification.
    - Produces faster convergence and higher accuracy.
- **Mean Squared Error (MSE):**
    - Slower and less appropriate for probabilistic outputs.
    - Leads to suboptimal convergence behavior.

Conclusion: Binary crossentropy is the ideal loss for binary sentiment tasks.



4.5. Regularization and Dropout

- **Dropout:** Randomly disables neurons during training, preventing co-adaptation and improving generalization.
- **L2 Regularization:** Penalizes large weights, reducing model complexity.

When applied, these techniques:

- Reduced validation loss fluctuation.
- Slightly improved validation accuracy.
- Delayed the onset of overfitting.

Conclusion: Regularization is essential for stabilizing performance in deeper or wider networks.

5. Performance Analysis

Model Variant	Activation	Loss	Hidden Layers	Dropout	Validation Accuracy	Test Accuracy
Baseline	ReLU	BCE	2	No	~85%	~82.8%
Shallow (1 layer)	ReLU	BCE	1	No	~78%	~77%
Deep (3 layers)	ReLU	BCE	3	No	~81%	~79%
Regularized	ReLU	BCE	2	Yes	~ 86%	~84%
Tanh	Tanh	BCE	2	No	~82%	~80%

6. Key Observations

1. **Overfitting:**
   After around 8–10 epochs, validation performance started to decline even though training accuracy kept increasing. This suggests that the network memorized the training data.
2. **Optimal Configuration:**
   Two hidden layers with 64 units each, ReLU activation, binary_crossentropy loss, and dropout regularization gave the best overall results.
3. **Training Dynamics:**
   The loss curve showed steady convergence, but overtraining

reduced test performance beyond the 10th epoch. Early stopping could further improve generalization.

4. **Computation Efficiency:**
   Simpler models (fewer layers and smaller units) train faster and can still achieve competitive accuracy if regularization is applied properly.

7. Recommendations and Future Work

To further improve performance:

- **Use Early Stopping:** Halt training when validation loss stops improving.
- **Increase Dropout Rate:** Combat overfitting without reducing learning capacity.
- **Experiment with Pre-trained Embeddings:** Incorporate GloVe or Word2Vec for better feature representation.
- **Explore Convolutional or Recurrent Architectures:** CNNs and LSTMs can capture contextual dependencies better than simple MLPs.
- **Hyperparameter Optimization:** Use grid or random search for fine-tuning layer sizes and learning rates.

8. Conclusion

This experiment demonstrates how neural network performance in sentiment analysis strongly depends on **architectural design and hyperparameter tuning**.

Key takeaways:

- Two hidden layers with ReLU activation and binary crossentropy loss yield strong baseline results (~83% accuracy).
- Increasing complexity without proper regularization leads to overfitting.

- Dropout and regularization techniques effectively stabilize validation performance.

In summary, **carefully balanced model architecture and regularization strategies** are crucial for achieving robust performance in sentiment classification tasks.