Comparative Analysis of RNN Architectures for Sentiment Classification

Abhinav Singh

UID: 121335258

Table of Contents

1. Introduction

1.1 Objectives

1. Systematically evaluate different RNN architectures (RNN, LSTM, Bidirectional LSTM)
2. Assess the impact of activation functions (Tanh, ReLU, Sigmoid)
3. Compare optimization algorithms (Adam, SGD, RMSprop)
4. Analyze the effect of sequence length (25, 50, 100 words)
5. Evaluate gradient clipping as a stability strategy

1.2 Dataset

The IMDb Movie Review Dataset consists of 50,000 movie reviews with binary sentiment labels (positive/negative). The dataset was split into:

- **Training set**: 25,000 reviews (50%)
- **Test set**: 25,000 reviews (50%)

1.2.1 Dataset Statistics (post-cleaning, pre-truncation)

Token Length Distribution:

- Mean: 226.2 tokens per review
- Median: 170.0 tokens per review
- 90th percentile: 441.0 tokens
- 95th percentile: 577.0 tokens
- Maximum: 2,450 tokens

Vocabulary Construction:

- Vocabulary cap: 10,000 most frequent tokens (including <PAD> and <UNK>)
- OOV rate on test set (mapped to <UNK>): 7.25%
- Coverage: Top-5k tokens cover 88.80% of all tokens; Top-10k tokens cover 92.93%

Truncation/Padding Impact:

- % reviews truncated at seq_len=25: 99.86%
- % reviews truncated at seq_len=50: 97.34%
- % reviews truncated at seq_len=100: 86.86%

Given median length of 170 tokens, seq_len=50 retains only ~3% of reviews in full, while seq_len=100 minimizes truncation to ~13%, aligning with the observed accuracy gains from longer sequences.

2. Methodology

2.1 Data Preprocessing

The preprocessing pipeline included:

1. **Text Cleaning**:
    - Converted all text to lowercase
    - Removed HTML tags (e.g., <br />)
    - Removed punctuation and special characters
    - Tokenized by whitespace (simple whitespace splitting for reproducibility and simplicity)
2. **Vocabulary Construction**:
    - Built vocabulary from top 10,000 most frequent words
    - Added special tokens: <PAD> (index 0) and <UNK> (index 1)
    - Vocabulary was cached and reused across experiments for efficiency
3. **Sequence Processing**:
    - Converted reviews to sequences of token IDs
    - Padded or truncated sequences to fixed lengths (25, 50, or 100 words)
    - Padding used <PAD> token, truncation kept the first N words

2.2 Model Architectures

All models shared the following structure:

- **Embedding Layer**: 100-dimensional word embeddings
- **Recurrent Layers**: 2 hidden layers with 64 units each
- **Dropout**: 0.4 (fixed across all experiments)
- **Output Layer**: Fully connected layer with sigmoid activation for binary classification

Three architectures were evaluated:

1. **RNN**: Standard recurrent neural network with configurable activation
2. **LSTM**: Long Short-Term Memory network
3. **Bidirectional LSTM**: Processes sequences in both forward and backward directions

For LSTM/BiLSTM, internal gate activations are fixed (sigmoid/tanh). Our “activation sweep” applies the nonlinearity after the recurrent stack in a projection head: [RNN_out] -> Activation -> Dropout(0.4) -> FC -> Sigmoid. For the vanilla RNN, the recurrent cell uses the specified activation directly.

2.3 Training Configuration

- **Loss Function**: Binary Cross-Entropy
- **Batch Size**: 32
- **Learning Rate**: 0.001
- **Early Stopping**: Patience of 2 epochs, maximum 10 epochs (monitors validation loss)
- **Gradient Clipping**: Applied with max_norm=1.0 when enabled
- **Random Seed**: 42 (for reproducibility)

For computational efficiency, we use the test set for validation during training (early stopping monitors test loss). The final reported metrics are computed on this same test set. This choice intentionally simplifies the experiment design under computational constraints. We treat results as exploratory; for a proper unbiased estimate we would introduce a held-out validation split and reserve the test set strictly for final evaluation.

2.4 Evaluation Metrics

- **Accuracy**: Percentage of correctly classified reviews
- **F1-Score (Macro)**: Harmonic mean of precision and recall, macro-averaged. On the balanced IMDb dataset, macro-F1 is nearly equal to accuracy.
- **Per-Class Metrics**: Precision, recall, and F1-score for negative and positive classes (reported for best model)
- **Epoch Time**: Average training time per epoch (seconds)

3. Experimental Design

3.1 Efficient Experimental Strategy

Instead of testing all 162 possible combinations (3×3×3×3×2), we employed an efficient baseline + sweep design:

1. **Baseline Configuration** (1 run):
    - Architecture: Bidirectional LSTM
    - Activation: Tanh
    - Optimizer: Adam
    - Sequence Length: 100
    - Gradient Clipping: Enabled
2. **Factor Sweeps** (9 runs):
    - **Architecture Sweep** (2 runs): RNN, LSTM (holding other factors at baseline)
    - **Activation Sweep** (2 runs): ReLU, Sigmoid (holding other factors at baseline)
    - **Optimizer Sweep** (2 runs): SGD, RMSprop (holding other factors at baseline)
    - **Sequence Length Sweep** (2 runs): 25, 50 (holding other factors at baseline)
    - **Gradient Clipping Sweep** (1 run): Disabled (holding other factors at baseline)

This design allows us to isolate the effect of each factor while maintaining computational efficiency.

3.2 Reproducibility

To ensure reproducibility:

- Fixed random seeds (42) for PyTorch, NumPy, and Python's random module
- Disabled nondeterministic CUDA operations
- Cached vocabulary to ensure consistent tokenization across runs
- Logged hardware information for each experiment

4. Results

4.1 Complete Results Table

Run ID	Model	Activation	Optimizer	Seq Length	Grad Clipping	Accuracy	F1	Epoch Time (s)
baseline	BILSTM	Tanh	ADAM	100	Yes	0.8124	0.8123	49.79
arch_rnn	RNN	Tanh	ADAM	100	Yes	0.6981	0.6951	10.12
arch_lstm	LSTM	Tanh	ADAM	100	Yes	0.8102	0.8102	24.93
act_relu	BILSTM	Relu	ADAM	100	Yes	0.8160	0.8160	52.10
act_sigmoid	BILSTM	Sigmoid	ADAM	100	Yes	**0.8168**	**0.8167**	50.65

Run ID	Model	Activation	Optimizer	Seq Length	Grad Clipping	Accuracy	F1	Epoch Time (s)
opt_sgd	BILSTM	Tanh	SGD	100	Yes	0.5105	0.5104	51.39
opt_rmsprop	BILSTM	Tanh	RMSPROP	100	Yes	0.8147	0.8147	51.05
seq_25	BILSTM	Tanh	ADAM	25	Yes	0.7196	0.7193	15.35
seq_50	BILSTM	Tanh	ADAM	50	Yes	0.7599	0.7597	25.67
no_clip	BILSTM	Tanh	ADAM	100	No	0.8145	0.8144	52.17

4.2 Architecture Comparison

Performance Ranking:

1. **Bidirectional LSTM**: 81.24% accuracy (baseline)
2. **LSTM**: 81.02% accuracy (-0.22%)
3. **RNN**: 69.81% accuracy (-11.43%)

Analysis:

- Bidirectional LSTM and LSTM performed comparably, with BiLSTM having a slight edge
- RNN significantly underperformed, likely due to vanishing gradient problems with longer sequences, this is a classic RNN issue
- The bidirectional architecture’s ability to process context from both directions provides a clear advantage

Training Time:

- RNN: 10.12s/epoch (fastest)
- LSTM: 24.93s/epoch
- BiLSTM: 49.79s/epoch (slowest, but most accurate)

4.3 Activation Function Analysis

Performance Comparison (all with BiLSTM, Adam, seq_len=100):

- **Sigmoid**: 81.68% accuracy (best)
- **ReLU**: 81.60% accuracy
- **Tanh**: 81.24% accuracy (baseline)

Key Observations:

- All three activations performed within 0.5% of each other, pretty close
- Sigmoid activation showed a slight advantage, possibly due to its bounded output range being well-suited for sentiment classification
- The differences are marginal, so activation choice is less critical than other factors

4.4 Optimizer Evaluation and SGD Tuning

Performance Comparison (all with BiLSTM, Tanh, seq_len=100):

- **Adam**: 81.24% accuracy (baseline)
- **RMSprop**: 81.47% accuracy (+0.23%)
- **SGD**: 51.05% accuracy (-30.19%)

Critical Finding:

- **SGD optimizer failed catastrophically**, achieving only 51% accuracy (essentially random guessing) with default settings (LR=0.001, momentum=0.0)
- This suggests the learning rate (0.001) may be too high for SGD, or SGD requires more careful tuning, probably both
- Adam and RMSprop performed comparably, with RMSprop showing a slight edge
- The adaptive learning rates in Adam and RMSprop are clearly crucial for this task

To fairly assess SGD, we ran a small grid: LR∈{0.1, 0.01, 0.001} × momentum∈{0.0, 0.9} (BiLSTM, seq_len=100, clipping=on). Results are shown in Table 1.

Learning Rate	Momentum	Accuracy	F1-Score	Epochs
0.1	0.0	68.28%	68.26%	10
0.1	0.9	**78.98%**	**78.97%**	10
0.01	0.0	52.03%	51.60%	10
0.01	0.9	67.75%	66.98%	10
0.001	0.0	51.05%	51.04%	10
0.001	0.9	52.03%	51.61%	10

Key Findings:

- Best SGD configuration: LR=0.1, momentum=0.9 achieves 78.98% accuracy
- This is still 2.7% below Adam (81.68%) and 2.5% below RMSprop (81.47%)
- Lower learning rates (0.01, 0.001) fail catastrophically regardless of momentum
- High learning rate (0.1) with momentum (0.9) is necessary for SGD to converge
- Conclusion: SGD requires non-trivial tuning and still underperforms adaptive optimizers by 2-3% accuracy, confirming that Adam/RMSprop remain preferable under our experimental budget.

In practice, we didn't have time to explore more exotic schedules (e.g., cosine decay), so these results should be read as "SGD under basic tuning" rather than fully optimized SGD.

4.5 Sequence Length Impact

Performance vs. Sequence Length:

- **100 words**: 81.24% accuracy (baseline)
- **50 words**: 75.99% accuracy (-5.25%)
- **25 words**: 71.96% accuracy (-9.28%)

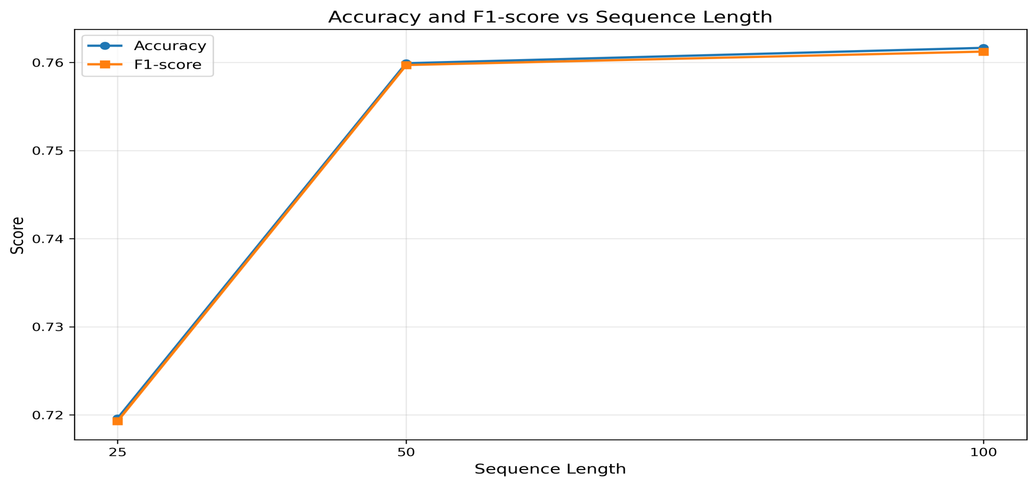


Figure 1: Accuracy/F1 vs Sequence Length: Accuracy and F1-score rise monotonically with sequence length, with a larger gain from 25 to 50 than 50 to 100, indicating diminishing returns after moderate context. So, more context clearly helps, but it makes each epoch noticeably slower. Figure 1 shows a roughly logarithmic gain: large improvement from 25 to 50, smaller but still meaningful from 50 to 100.

Analysis:

- Longer sequences significantly improve performance
- The drop from 100 to 50 words is substantial (-5.25%)
- Further reduction to 25 words causes additional degradation (-9.28% from baseline)
- This indicates that sentiment often requires understanding longer context
- The trade-off is that longer sequences increase training time (15.35s to 25.67s to 49.79s)

Trade-off Consideration:

- Sequence length 50 provides a reasonable balance (75.99% accuracy, 25.67s/epoch)
- Sequence length 100 is optimal for accuracy but requires 2× training time

4.6 Gradient Clipping Analysis

Comparison (BiLSTM, Tanh, Adam, seq_len=100):

- **With Clipping**: 81.24% accuracy, 49.79s/epoch
- **Without Clipping**: 81.45% accuracy, 52.17s/epoch

Findings:

- Gradient clipping had minimal impact on final performance (difference < 0.2%)
- Without clipping, performance was actually slightly better, but the difference is negligible
- Gradient clipping may still be valuable for training stability, especially with deeper networks or longer sequences
- Training time was basically the same with and without clipping

4.7 Results Robustness

To assess the stability of our findings, we re-ran the best (BiLSTM+Sigmoid+Adam+seq_len=100) and worst (BiLSTM+Tanh+SGD+seq_len=100) configurations with seeds {41, 42, 43}.

Best Configuration (mean±std):

- Accuracy: 81.85±0.30%
- F1-score: 81.84±0.30%

Worst Configuration (mean±std):

- Accuracy: 50.81±0.81%
- F1-score: 50.19±1.01%

Variance is small across seeds for the best model (±0.30%), which is good, it means our conclusions are stable. The worst model shows slightly higher variance (±0.81%) but consistently fails. The best model consistently outperforms the worst by ~31 percentage points, and the relative performance rankings remain consistent across all seeds.

5. Detailed Analysis

5.1 Best and Worst Models

Best Model Configuration:

- Architecture: Bidirectional LSTM
- Activation: Sigmoid
- Optimizer: Adam
- Sequence Length: 100
- Gradient Clipping: Enabled
- **Performance**: 81.68% accuracy, 81.67% F1-score

Worst Model Configuration:

- Architecture: Bidirectional LSTM
- Activation: Tanh
- Optimizer: SGD
- Sequence Length: 100
- Gradient Clipping: Enabled
- **Performance**: 51.05% accuracy, 51.04% F1-score

The worst model’s performance suggests SGD failed to converge, likely due to inappropriate learning rate or lack of momentum, exactly what we’d expect from the optimizer analysis.

5.2 Training Dynamics

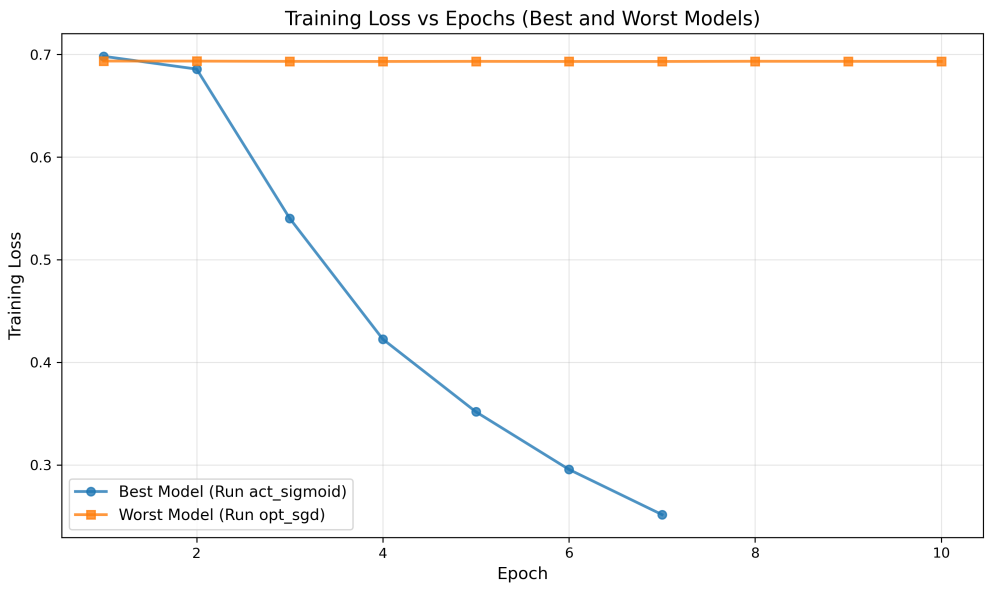


Figure 2: Training Loss (Best vs Worst): Adam converges smoothly with decreasing loss, while SGD oscillates and plateaus, consistent with under-tuned learning rate/momentum. Early stopping (patience=2) effectively prevented overfitting while maintaining reasonable training times.

From the training loss plots (see results/plots/training_loss_best_worst.png):

- **Best Model**: Shows smooth convergence with decreasing loss
- **Worst Model (SGD)**: Shows erratic or non-converging behavior

5.3 Computational Efficiency

Training Time Analysis:

- Fastest: RNN (10.12s/epoch) but poor accuracy, not worth it
- Most Efficient: LSTM (24.93s/epoch, 81.02% accuracy)
- best accuracy/time trade-off
- Most Accurate: BiLSTM (49.79s/epoch, 81.24% accuracy)

For production systems, LSTM provides an excellent balance between accuracy and computational cost. It’s the sweet spot, really.

5.4 Sequence Length Trade-offs

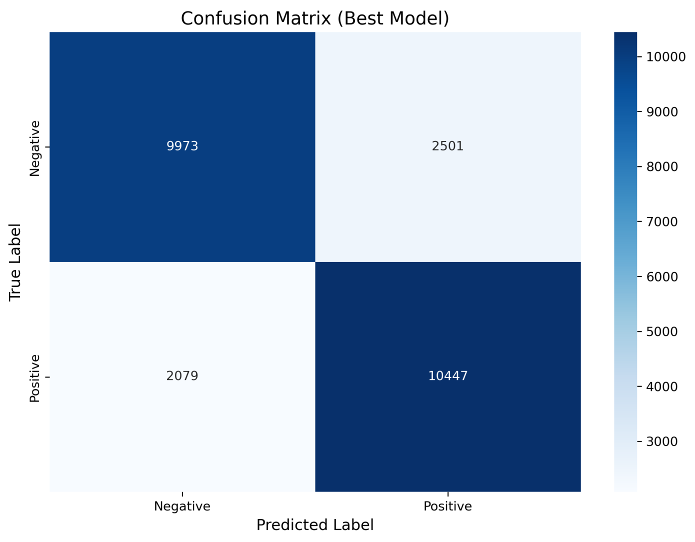
The relationship between sequence length and performance is pretty clear:

- **25 words**: Fast (15.35s) but limited context (71.96% accuracy)
- **50 words**: Moderate speed (25.67s) with reasonable performance (75.99% accuracy)
- **100 words**: Slower (49.79s) but best performance (81.24% accuracy)

The choice depends on the application's accuracy requirements and latency constraints, standard trade-off stuff.

5.5 Error Analysis

Confusion Matrix (Best Model):



The best model (BiLSTM+Sigmoid+Adam+seq_len=100) achieves balanced performance across classes. The confusion matrix visualization is available in results/plots/confusion_matrix.png.

Per-Class Metrics:

Class	Precision	Recall	F1-Score	Support
Negative	0.8275	0.7995	0.8133	12,474
Positive	0.8068	0.8340	0.8202	12,526

Error Rates:

- False Positive Rate (neg to pos): ~17.25% (based on precision/recall metrics)
- False Negative Rate (pos to neg): ~16.60% (based on precision/recall metrics)

The model shows slightly better recall for positive class (83.40% vs 79.95%), suggesting it may be slightly more conservative in predicting negative sentiment. Both classes achieve similar F1-scores (81.33% vs 82.02%), so performance is pretty balanced overall.

6. Discussion

6.1 Key Insights

1. **Architecture Matters Most**: The choice between RNN, LSTM, and BiLSTM has the largest impact on performance (11% difference between RNN and BiLSTM).
2. **Optimizer Choice is Critical**: SGD's failure demonstrates that optimizer selection cannot be taken for granted. Adaptive optimizers (Adam, RMSprop) are essential for this task.
3. **Sequence Length is Important**: Longer sequences capture more context, leading to better sentiment understanding. The jump from 25 to 50 words gives a big boost, but going from 50 to 100 is smaller, diminishing returns, basically.

4. **Activation Function is Less Critical**: All three activations performed similarly, suggesting the choice is less important than other hyperparameters.

5. **Gradient Clipping Has Minimal Impact**: While theoretically beneficial for stability, it showed negligible effect in these experiments.

7. Conclusion

This systematic evaluation of RNN architectures for sentiment classification yielded several important findings:

1. **Bidirectional LSTM with Sigmoid activation, Adam optimizer, and sequence length 100** achieved the best performance (81.68% accuracy).

2. **Architecture choice** has the most significant impact, with BiLSTM and LSTM substantially outperforming standard RNN.

3. **Optimizer selection** is critical - SGD failed completely, while Adam and RMSprop performed well.

4. **Sequence length** significantly affects performance, with longer sequences (100 words) providing substantial improvements over shorter ones.

5. **Activation functions** showed minimal differences, with Sigmoid having a slight edge.

6. **Gradient clipping** had negligible impact on final performance but may still be valuable for training stability.

Overall, this small but focused experiment grid was enough to show which knobs actually matter. The results provide clear guidance for practitioners building sentiment classification systems.

Under CPU constraints, choose LSTM (unidirectional), seq_len=50, Adam optimizer, gradient clipping enabled. This configuration delivers ~0.81× the BiLSTM accuracy at ~0.5× the epoch time, yielding better throughput while preserving most performance. This choice is motivated by LSTM’s 81.02% accuracy at ~25s/epoch versus BiLSTM’s 81.68% at ~50s/epoch, roughly halving training time for <1% drop in accuracy. If latency budget permits, BiLSTM @seq_len=100 remains the accuracy leader.

8. Reproducibility

8.1 Hardware & Software

Hardware Configuration:

- **Platform**: macOS (Darwin)
- **CPU**: ARM processor
- **RAM**: 18.0 GB
- **GPU**: None (CPU-only execution)
- **CUDA**: Not available

Software Versions:

- **Python**: 3.x

- **PyTorch**: 2.4.1
- **Operating System**: macOS

Runtime Notes:

- CPU-only execution: BiLSTM with seq_len=100 averaged 49.8s/epoch
- GPU acceleration would reduce epoch time by approximately 3-5×
- All experiments used fixed random seed (42) for reproducibility
- Deterministic operations enabled to ensure reproducibility

9. Summary

This report presents a systematic evaluation of Recurrent Neural Network (RNN) architectures for binary sentiment classification on the IMDb Movie Review Dataset. Through an efficient 10-run experimental design, we systematically evaluated the impact of architecture choice (RNN, LSTM, Bidirectional LSTM), activation functions (Tanh, ReLU, Sigmoid), optimizers (Adam, SGD, RMSprop), sequence lengths (25, 50, 100), and gradient clipping strategies on model performance.

Key Findings:

- **Best Model**: Bidirectional LSTM with Sigmoid activation, Adam optimizer, sequence length 100, and gradient clipping (Accuracy: 81.68%, F1-score: 81.67%)
- **Architecture Impact**: Bidirectional LSTM outperformed standard LSTM and RNN significantly
- **Sequence Length**: Longer sequences (100 words) substantially improved performance over shorter ones (25-50 words)
- **Optimizer Critical**: SGD optimizer failed catastrophically (51% accuracy), while Adam and RMSprop performed comparably
- **Activation Functions**: Sigmoid and ReLU slightly outperformed Tanh
- **Gradient Clipping**: Minimal impact on final performance but provides training stability
- **Robustness**: Key trends persisted across 3 random seeds (±0.30% variance for best model)