Comparative Analysis of RNN Architectures for Sentiment Classification

Author: Rohan Jain | **ID:** 121012081 | **Date:** November 12, 2025
Hardware: Google Colab GPU (T4), Python 3.10, TensorFlow 2.15

GitHub Link: https://github.com/Rohanjain2312/rnn-sentiment-analysis

---

1. Dataset Summary

The IMDb dataset contains 50,000 movie reviews (25,000 train/25,000 test) with balanced positive/negative labels.

Preprocessing:

- Lowercased text and removed punctuation using regex
- Tokenized using IMDb word index, keeping top 10,000 words
- Padded/truncated sequences to lengths: 25, 50, 100 words

Statistics:

- Vocabulary: 10,000 words
- Average review length: ~130 words (pre-padding)
- Class distribution: 50% positive, 50% negative

---

2. Model Configuration

Architecture:

- Embedding layer: dimension 100, vocabulary 10,000
- Recurrent layer: 64 units (128 for Bidirectional)
- Dropout: 0.5
- Output: Dense layer with sigmoid activation

Training Parameters:

- Epochs: 5, Batch size: 32, Learning rate: 0.001
- Loss: Binary cross-entropy, Validation split: 20%

Experimental Variations (243 total):

- Models: RNN, LSTM, Bidirectional LSTM
- Activations: Sigmoid, ReLU, Tanh
- Optimizers: Adam, SGD, RMSprop
- Sequence lengths: 25, 50, 100

- Gradient clipping: None, Clipnorm (1.0), Clipvalue (0.5)

---

3. Comparative Analysis

Top 5 Configurations

Model	Activation	Optimizer	Seq Length	Grad Clip	Accuracy	F1-Score	Time (s)
LSTM	Tanh	Adam	100	None	0.7650	0.7670	58.2
LSTM	ReLU	RMSprop	100	Clipvalue	0.7642	0.7658	59.1
LSTM	Tanh	RMSprop	100	Clipnorm	0.7638	0.7655	59.3
Bidir LSTM	Tanh	Adam	100	Clipnorm	0.7625	0.7648	77.5
Bidir LSTM	ReLU	RMSprop	100	None	0.7618	0.7642	78.8

Performance by Architecture

Architecture	Avg Accuracy	Avg Epoch Time (s)
LSTM	71.2%	58.7
Bidirectional LSTM	70.8%	78.2
Simple RNN	63.5%	42.3

Finding: LSTM outperforms RNN by 7.7% while Bidirectional LSTM adds only 0.4% improvement with 33% more computation.

Impact of Sequence Length

Model	Length 25	Length 50	Length 100	Change
LSTM	66.2%	71.3%	76.5%	+10.3%
Bidir LSTM	66.0%	70.8%	74.9%	+8.9%
RNN	63.8%	65.7%	62.4%	-1.4%

Finding: LSTM benefits strongly from longer sequences (+10.3%) while RNN performance degrades (-1.4%) due to vanishing gradients.

Impact of Optimizer

Optimizer	Avg Accuracy	Performance
Adam	69.8%	Best
RMSprop	69.5%	-0.3%
SGD	65.2%	-4.6%

Finding: Adam outperforms SGD by 4.6%. Adaptive learning rates are essential for RNN training.

Impact of Activation & Gradient Clipping

Activation Functions: Tanh (68.5%) > ReLU (68.2%) > Sigmoid (68.1%) - minimal difference (<1%)

Gradient Clipping: None (68.3%), Clipnorm (68.5%), Clipvalue (68.4%) - minimal impact (<0.2%)

---

4. Discussion

Best Configuration

LSTM + Tanh + Adam + Length 100 achieved 76.5% accuracy and 0.767 F1-score. LSTM's gating mechanisms handle long-term dependencies effectively, while Adam's adaptive learning rates enable efficient convergence.

Sequence Length Impact

Sequence length had the strongest effect. LSTM improved 10.3% from length 25→100 as longer sequences provide more context. Conversely, RNN declined 1.4% due to vanishing gradients over longer sequences.

Optimizer Impact

Adam achieved 4.6% higher accuracy than SGD. Adaptive learning rates (Adam, RMSprop) handle RNN's complex loss landscape better than fixed-rate SGD. This makes adaptive optimizers essential for RNN training.

Gradient Clipping Impact

Gradient clipping showed minimal impact (<0.2% difference) and no instability was observed without clipping. Modern initialization (Glorot) and adaptive optimizers (Adam) inherently control gradient magnitudes, making explicit clipping optional for this shallow architecture.

---

5. Conclusion

Optimal Configuration for CPU Constraints

Recommended Setup:

- Model: LSTM
- Activation: Tanh
- Optimizer: Adam (lr=0.001)
- Sequence Length: **50 words**
- Gradient Clipping: None
- Dropout: 0.5, Batch: 32

Expected Performance: 71.3% accuracy, ~0.710 F1-score, ~6-8 min/epoch (CPU)

Justification

Why LSTM over Bidirectional: Bidirectional LSTM requires 33% more computation for only 0.4% accuracy gain—not worthwhile under CPU constraints.

Why Length 50 over 100: Length 50 achieves 71.3% accuracy (93% of maximum) with 40% less training time than length 100. This provides the best accuracy/speed trade-off for CPU environments.

Why Adam: Converges faster than SGD (4.6% better accuracy) without manual learning rate tuning.

Why No Clipping: No stability issues observed; Adam inherently controls gradients.

Key Takeaways

1. LSTM essential for sentiment classification (7.7% better than RNN)
2. Longer sequences dramatically improve LSTM performance (+10.3%)
3. Adaptive optimizers (Adam/RMSprop) critical for RNN training
4. Bidirectional adds minimal value for this task (0.4%)
5. Gradient clipping optional for shallow architectures with Adam

Final Recommendation: Deploy LSTM with sequence length 50 for optimal CPU performance, delivering 71.3% accuracy with reasonable training time.