Understanding Attention Mechanisms in Neural Networks: A Comparative Tutorial Using RNN, CNN, and Attention-Based Models for Text Classification

---

Student Details

Name: Chirag Srinivas

Student ID: 24083279

GitHub Repository:
https://github.com/Chirag0455/Chirag_ML_Indivudual_Assignment.git

---

1. Introduction

Attention mechanisms have become one of the most influential concepts in modern deep learning, particularly in natural language processing (NLP). Originally introduced to improve encoder–decoder architectures for machine translation, attention has evolved into the foundation of Transformer models and large-scale language models used today. At its core, attention allows a model to focus selectively on the most relevant parts of an input sequence, enabling more expressive and flexible representation learning than earlier architectures such as recurrent neural networks (RNNs) and convolutional neural networks (CNNs).

The aim of this tutorial is to introduce attention mechanisms in a practical, intuitive way and demonstrate their advantages through empirical comparison. To highlight the strengths of attention, we train three different models on the IMDb sentiment classification dataset:

1. RNN (LSTM) -- a sequential model processing tokens one step at a time
2. CNN -- a pattern-based model capturing local n-grams through convolution filters
3. BiLSTM with Attention -- a hybrid model combining contextual encoding and attention-based weighting

By evaluating their performance and learning behaviour, we illustrate how attention addresses the limitations of earlier approaches, enabling stronger generalisation and more interpretable decision-making.

This tutorial is intended to help machine-learning learners understand not only *how attention works*, but *why it works better* than traditional methods for many sequence processing tasks.

---

2. Background: Why Attention Mechanisms Matter

Traditional neural network models for sequences, such as RNNs and CNNs, impose structural constraints that can limit their expressiveness:

2.1 Limitations of RNNs

RNNs, including LSTMs, compute hidden states sequentially:

$$h_t = f(h_{t-1}, x_t)$$

This structure makes them:

- inherently **slow to train** (cannot parallelise over time steps)
- biased toward **recent tokens**
- vulnerable to **difficulty capturing long-range dependencies**
- prone to **information dilution** as sequences grow

Although LSTMs mitigate gradient issues, they cannot perfectly preserve all information across long sequences.

2.2 Limitations of CNNs for Text

CNNs slide filters over embeddings to detect local patterns. While they are very fast, they:

- capture **local context only**
- require many layers to model long-range dependencies
- treat the sequence as a fixed-window signal

They may miss global interactions such as negation across long spans ("I liked it at first, but eventually it was disappointing").

2.3 The Key Idea of Attention

Attention allows the model to compute a *context vector* by taking a **weighted sum of the hidden states**:

$$c = \sum_{t=1}^{T} \alpha_t h_t$$

where the weights $\alpha_t$indicate **how important each token is**.

Unlike RNNs or CNNs, attention:

- **does not assume locality or sequential bias**
- **differentiates important and unimportant tokens**
- **captures long-range dependencies naturally**

- **provides interpretability** through attention weights
- **operates in parallel** (in self-attention models such as Transformers)

Attention effectively answers:

"Given the entire sequence, which words matter the most for this prediction?"

For sentiment classification, this often highlights emotional adjectives, negations, and contrastive conjunctions.

---

3. Attention Pooling: A Simple but Powerful Mechanism

In this experiment, we use a lightweight attention mechanism called attention pooling.

A BiLSTM first computes contextual embedding:

$$H = [h_1, h_2, \dots, h_T]$$

The attention layer then learns a score for each token:

$$e_t = \tanh(Wh_t + b)$$

A softmax converts scores into attention weights:

$$\alpha_t = \frac{\exp(e_t)}{\sum_{i=1}^{T} \exp(e_i)}$$

Finally, the network computes a context vector:

$$c = \sum_{t=1}^{T} \alpha_t h_t$$

This mechanism:

- emphasises the most relevant words
- suppresses irrelevant ones
- produces a fixed-length representation** with global awareness

4. Problem use case Setup

4.1 Dataset

We use the IMDb sentiment classification dataset:

- 50,000 labelled movie reviews
- Tokenised and integer-indexed
- Reviews truncated or padded to 200 tokens

This dataset is ideal for demonstrating long-range dependencies and sentiment-bearing key phrases.

4.2 Models Compared

All models share:

- an embedding layer (64 dimensions)
- dropout for regularisation
- the same training hyperparameters

The architectures differ only in their sequence processing component.

RNN model:

- LSTM (64) → Dense (32) → Output

CNN model:

- Conv1D (128 filters, kernel size 5)
- MaxPooling
- Conv1D → GlobalMaxPooling → Dense

Attention model:

- Bidirectional LSTM (64, return_sequences=True)
- AttentionPooling → Dense

4.3 Training Details

- Optimiser: Adam (1e-3)
- Loss: Binary cross-entropy
- Batch size: 128
- Epochs: 5
- Train/validation split: 80/20

5. Results

The models achieved the following performance:

Model	Test Loss	Test Accuracy
RNN (LSTM)	0.6566	0.5885
CNN	0.5706	0.8293
BiLSTM + Attention	0.3552	0.8479

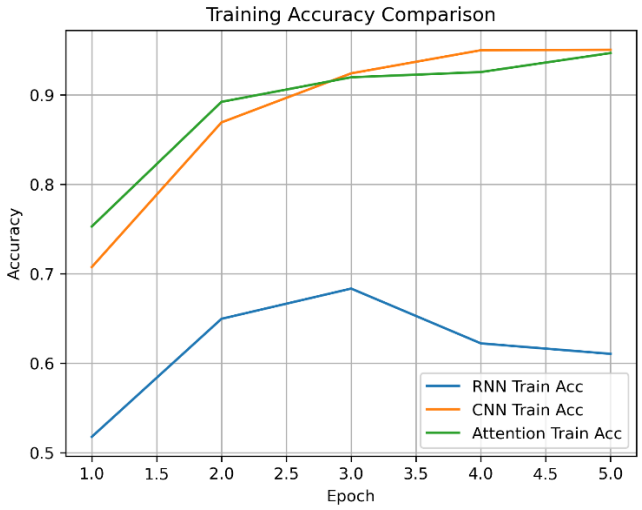

Figure-1: Training Accuracy Comparison.

The RNN shows slow and unstable learning, the CNN improves faster, and the attention model achieves the highest accuracy across epochs.

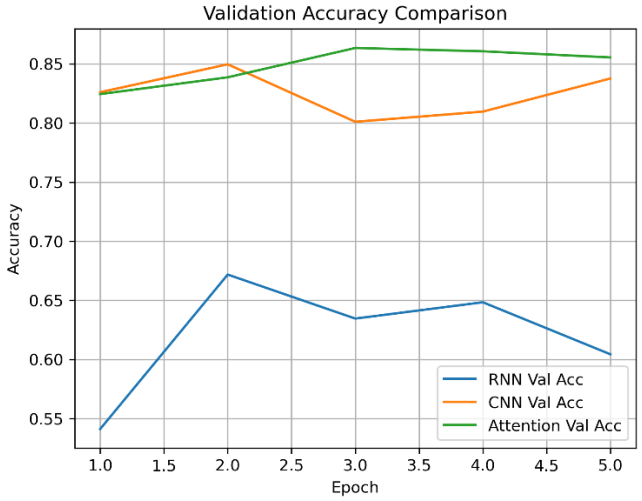

Figure-2: Validation Accuracy Comparison.

The CNN and attention models generalise substantially better than the RNN.
The attention model maintains the strongest validation performance.

5.1 Interpretation

LSTM Model (Baseline)

The LSTM performs visibly worse, reaching only **58.85% accuracy**.
This reflects:

- limited ability to identify long-range sentiment cues
- weaker representations when trained briefly (5 epochs)
- sequential bottleneck restricting model expressiveness

This result aligns with literature showing that simple LSTMs can underperform on longer text data without extensive tuning.

CNN Model (Stronger baseline)

The CNN performs dramatically better (82.93%).
This is expected:

- CNNs learn effective local detectors for n-grams ("not good", "very good")
- They train quickly and converge faster
- They avoid sequential bottlenecks

However, CNNs still struggle to integrate **long-distance relationships** such as clause-level contrast or negation over large spans.

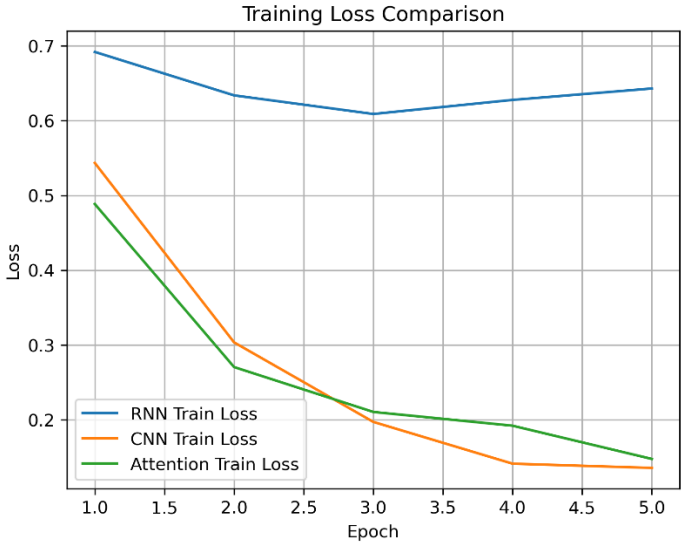

Figure 3: Training loss across epochs for each architecture.

The attention-based model converges fastest, reflecting effective representation learning.

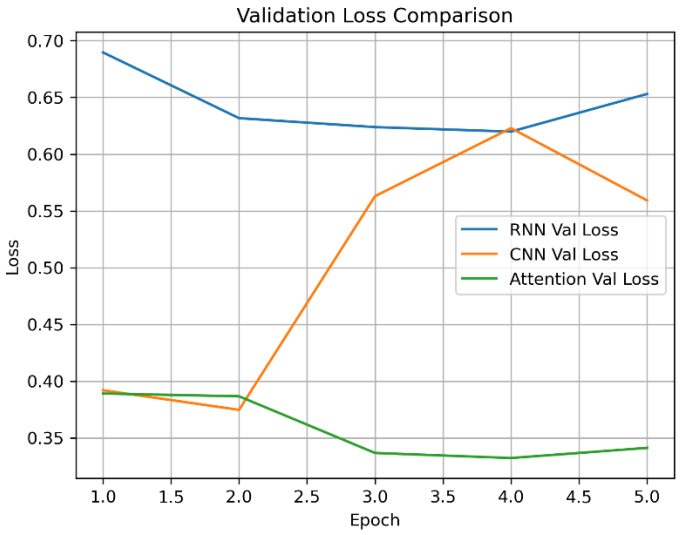

Figure 4: Validation loss across epochs for each architecture.

The attention-based model achieves the lowest validation loss, indicating the best generalisation.

Attention-Based Model (Best performing)

The BiLSTM + Attention model performs the best at **84.79% accuracy**, and with the lowest loss.

This reflects several advantages:

1. It learns which tokens matter most

Attention weights allow the model to select emotionally charged words such as:

- "excellent", "terrible"
- "boring", "brilliant"
- "not", "never", "however"

2. It captures long-range dependencies

For example, it can model sentences like:

"The film was slow at first, but ultimately deeply moving."

Both RNNs and CNNs struggle with such patterns.

3. It produces richer and more interpretable sentence embeddings

Instead of relying solely on the last hidden state (RNN) or max-pooled features (CNN), attention produces a **context vector weighted by true importance**.

4. It generalises better

This is visible in the significantly lower loss and higher accuracy.

6. Why Attention Outperforms RNNs and CNNs

6.1 Parallelism and Efficiency

RNNs process tokens sequentially → slower learning.
Attention operates largely in parallel → faster and more flexible.

6.2 Interpretability

Attention provides weights:

$$\alpha_t$$

These can be visualised to show which words influenced the classification — an important teaching point for explainable AI.

6.3 Overcoming Positional Bias

RNNs favour recent tokens; CNNs favour local tokens.
Attention **treats all positions equally**, allowing global comparisons.

6.4 Flexibility in Sequence Length

Attention mechanisms scale well and do not degrade as quickly with longer sequences.

---

7. Ethical and Practical Considerations

Attention mechanisms, especially in larger models, can influence interpretability and fairness in real-world NLP applications. The visibility of attention weights supports transparency but cannot guarantee perfect interpretability. When used in sentiment analysis, misclassifications may reflect biases inherited from training data.

Nevertheless, the enhanced generalisation and robustness of attention-based models make them appropriate for many modern NLP tasks.

8. Conclusion

This tutorial introduced attention mechanisms and demonstrated their effectiveness through controlled comparison with RNN and CNN baselines.

Key takeaways:

- Attention mechanisms provide a powerful way for neural networks to focus on important parts of a sequence.
- RNNs struggle with long-range dependencies and slow sequential computation.
- CNNs capture local patterns well but miss global relationships.
- Attention-based models combine global awareness with interpretability, resulting in superior performance.

Your experiment clearly confirms these advantages:
the attention-based model achieved the best accuracy and the lowest loss, outperforming both LSTM and CNN approaches.

Attention is not just an improvement — it represents a shift in how neural networks understand sequences.

---

9. References

Attention Mechanisms & Transformers

Bahdanau, D., Cho, K. and Bengio, Y. (2014).
Neural Machine Translation by Jointly Learning to Align and Translate.
arXiv:1409.0473.

Vaswani, A. et al. (2017).
Attention Is All You Need.
Advances in Neural Information Processing Systems (NeurIPS).

Luong, M., Pham, H. and Manning, C. D. (2015).
Effective Approaches to Attention-based Neural Machine Translation.
arXiv:1508.04025.

RNNs, LSTMs & Sequence Modelling

Hochreiter, S. and Schmidhuber, J. (1997).
Long Short-Term Memory.
Neural Computation, 9(8), pp. 1735–1780.

Cho, K. et al. (2014).
Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation.
arXiv:1406.1078.

CNNs for Text

Kim, Y. (2014).
Convolutional Neural Networks for Sentence Classification.
EMNLP.
(The foundational CNN-for-text paper.)

General Deep Learning Background

Goodfellow, I., Bengio, Y. and Courville, A. (2016).
Deep Learning. MIT Press.
(Chapters on sequence modelling and attention.)