Mamba–the First Linear-time Sequence Model that Achieves Transformer-quality Performance

TEAM FLASTRY

Xiaoke Song (xsong40)
Kejing Yan (kyan28)
Ruoyun Yang (ryang104)

GITHUB

https://github.com/anyfruit/mamba-sentiment-dataset.git

Paper: Gu, A., Dao, T., Ermon, S., R´e, C., & Rudra, A. (2023). Mamba: Linear-Time Sequence Modeling with Selective State Spaces. arXiv:2312.00752. https://arxiv.org/abs/2312.00752

01. Introduction

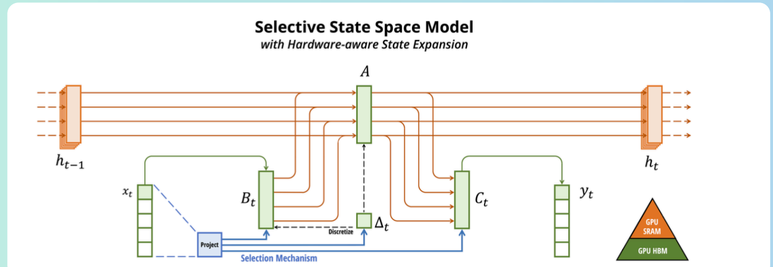
- **Objective**: Re-implement the Mamba model using TensorFlow and evaluate its performance on a sentiment classification task using the Sentiment140 dataset (1.6M labeled tweets).
- **Motivation**: Sentiment analysis requires understanding context, sarcasm, and nuance, making it a strong test case for sequential models like Mamba.
- **Why Mamba?**
  - Combines state space models (SSMs) with an **input-dependent selection mechanism**.
  - Achieves linear-time sequence modeling through **hardware-aware scanning operations**, avoiding the quadratic scaling issues of Transformers.
  - **Improves over earlier SSMs** by better handling information-dense, discrete data like text.
- **Research Goals**:
  - Assess Mamba's ability to model textual data efficiently and effectively.
  - Compare Mamba to Transformer and other models in terms of accuracy and inference efficiency.
  - Analyze trade-offs and implications for deploying lightweight NLP models in practice.

02. Data & Preprocessing

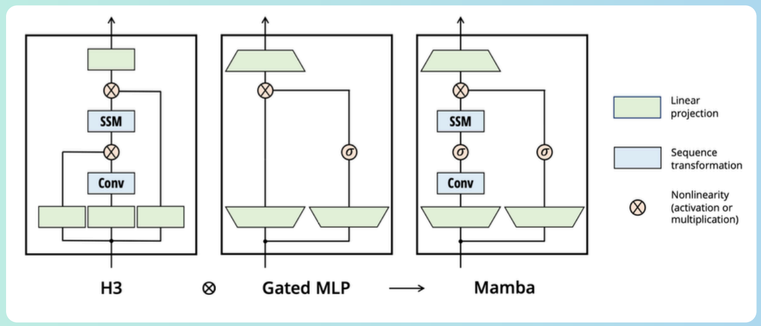
- **Dataset**:
  - Used the Sentiment140 dataset with ~1.6M tweets labeled as positive, neutral, or negative.
  - Original labels (0, 2, 4) were remapped to (0, 1, 2) for simplicity.
- **Data Collection Background**:
  - Tweets were collected via the Twitter API using emoticons, making it an early example of distant supervision in sentiment analysis.
- **Preprocessing Steps**:
  - Cleaned tweets by removing URLs, mentions, hashtags, and special characters; lowercase.
  - Tokenized text using BERT tokenizer (bert-base-uncased) with a maximum sequence length of 64 tokens.
  - Split the dataset into 98% training, 1% validation, and 1% testing, preserving label balance.
  - Saved processed datasets as CSVs and loaded them into TensorFlow datasets for training.
- **Standardization**:
  - Preprocessing pipeline was applied consistently across SSM, LSTM, Transformer, and Mamba models.

03. Methodology

- **SSMs--Structured State Space Models:**
  - SSMs offer a highly efficient alternative to Transformer-based architectures for sequential modeling by achieving linear time complexity O(L) in sequence length L, compared to the quadratic complexity O(L^2) of Transformer self-attention layers.
- **Mamba Re-implementation:**
  - We faithfully re-implemented the Mamba architecture in TensorFlow. Mamba's design leverages GPU memory hierarchy by minimizing costly memory transfers between fast and slow memory.
- **Model Structure:**
  - Mamba extends SSMs by introducing dynamic input-dependent parameters (Delta_t, B_t, C_t), allowing the model to selectively weight information and better handle discrete, information-dense text data. (See Figure Below)



- **Mamba Layer Components (5):**
  - Linear Upscaling: Project input embeddings to higher dimensions.
  - Local Convolution Mixing: Capture local patterns independently across channels.
  - Selective State Space Update: Apply a selective scan across token sequences.
  - Gated Multiplication: Modulate information flow dynamically.
  - Linear Downscaling: Return to original embedding size with residual connections.

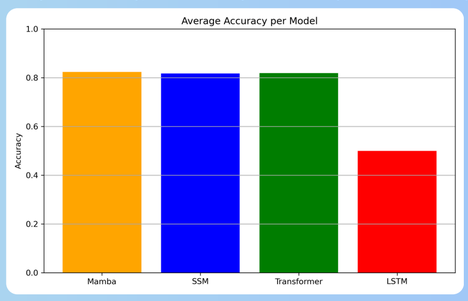


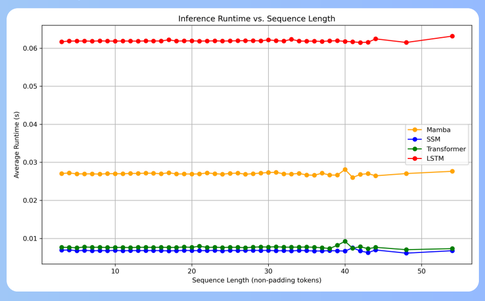
- **Comparison Models:**
  - For benchmarking, we implemented LSTM, Transformer, and a simple SSM baseline, matching model depth (2 layers), hidden sizes, and dropout rates as closely as possible to ensure fair comparison.

04.Results

We evaluated four models — Mamba, SSM, Transformer, and LSTM — on accuracy and inference efficiency.

- **Accuracy :**
  - Mamba achieved the highest accuracy (82.32%), followed by Transformer (81.86%) and SSM (81.76%). LSTM underperformed at 49.98%, likely due to sequential processing limitations.
- **Inference Runtime vs. Sequence Length:**
  - SSM and Transformer maintained the lowest and most stable runtimes. Mamba showed slightly higher but consistent runtimes, while LSTM remained the slowest regardless of sequence length. On average, SSM was fastest (0.0068s), followed by Transformer (0.00767s), Mamba (0.02704s), and LSTM (0.06191s).





The table below showcases qualitative examples by different models.

Tweet	Mamba	SSM	Transformer	LSTM	True Label
"relaxing.. "	Positive	Positive	Positive	Negative	Positive
"Haha, I agree, they rock and they are awesome."	Negative	Negative	Positive	Negative	Positive
"Nice one bruv! Hope you can make it along"	Positive	Positive	Positive	Negative	Positive

05. Challenges

- **Framework Translation:**
  - Re-implementing in TensorFlow without access to PyTorch's custom CUDA kernels for selective scan.
- **Manual Reconstruction:**
  - Built selective scan logic manually using tf.einsum, tf.cumsum, and custom reshaping.
- **Tensor Shape Mismatches:**
  - Frequent issues arose from handling multi-dimensional tensors (B, L, D, N), leading to silent computation errors.
- **Debugging Complexity:**
  - Diagnosing shape misalignments required detailed step-by-step inspection and insertion of intermediate assertions.

06. Discussion & Future Work

- **Lessons Learned:**
  - Re-implementing required careful translation of research algorithms.
  - Building standardized pipelines for fair model comparison was crucial for meaningful evaluation.
  - Hands-on debugging of dynamic tensor operations deepened our understanding of sequence modeling.
- **Limitations:**
  - Mamba's runtime advantage was less visible on short text sequences (tweets) compared to longer, more complex inputs.
- **Future Work:**
  - More epochs and tune hyperparameters
  - Conduct ablation studies on Mamba's input-dependent parameters to isolate their impact.
  - Explore model behavior across different demographics to assess fairness and generalization.
  - Investigate scaling models to longer sequences and more diverse NLP tasks.