Transformer and LSTM Implementations for English-Urdu Machine Translation

SYED MUHAMMAD MURTAZA KAZMI
Department of Computer Science
National University of Computer & Emerging Sciences
Islamabad, Pakistan
i210685@nu.edu.pk

***Abstract*—This report presents the implementation of Transformer and Long Short-Term Memory (LSTM) models for machine translation from English to Urdu using the UMC005 Parallel Corpus. The study encompasses data preprocessing, model architecture design, hyperparameter tuning, training, and evaluation using BLEU and ROUGE metrics. Comparative analysis highlights the performance differences between the two models in terms of translation accuracy, training time, memory usage, and inference speed.**

***Index Terms*—Machine Translation, Transformer, LSTM, English-Urdu, BLEU Score, ROUGE, Deep Learning**

I. Introduction

Machine Translation (MT) plays a vital role in breaking linguistic barriers, enabling effective communication between diverse language speakers. Traditional approaches such as rule-based systems and statistical machine translation have largely been replaced by neural machine translation (NMT), driven by deep learning advancements. Among NMT models, the Transformer architecture has established itself as the state-of-the-art, leveraging self-attention mechanisms to capture global dependencies in sentences.

This report explores the implementation and evaluation of both Transformer and Long Short-Term Memory (LSTM) models for English-to-Urdu translation. The UMC005 Parallel Corpus serves as the primary dataset, providing parallel texts in English and Urdu. The study aims to compare the performance of these architectures in terms of translation quality, computational efficiency, and overall practicality.

II. Methodology

A. Dataset

The UMC005: English-Urdu Parallel Corpus includes parallel translations from religious texts, such as the Bible and the Quran. It provides a balanced dataset for machine translation tasks, containing:

- **Bible Subset**: Aligned English and Urdu sentences from the Bible.
- **Quran Subset**: Aligned English and Urdu sentences from the Quran.

The dataset is divided into training, validation, and testing sets to ensure effective model training and evaluation.

B. Data Preprocessing

Effective preprocessing is crucial for the success of NMT models. The following steps were applied:

- **Text Cleaning**:
  - Converted all text to lowercase to maintain uniformity.
  - Removed special characters, punctuation, and digits.
  - Eliminated extra spaces and trimmed unnecessary whitespace.
- **Tokenization**:
  - Employed whitespace-based tokenization for both English and Urdu.
  - Experimented with subword tokenization techniques like Byte Pair Encoding (BPE) to handle rich morphological features in Urdu.
- **Vocabulary Construction**:
  - Created token-to-index mappings, including special tokens (`<PAD>`, `<UNK>`, `<SOS>`, `<EOS>`).
  - Set a minimum token frequency threshold to filter rare words.
- **Sequence Indexing**:
  - Converted tokenized sentences into indexed sequences.
  - Added `<SOS>` and `<EOS>` tokens to target sequences for proper sequence generation.
- **Data Splitting**:
  - Split the dataset into training (90%) and validation (10%) sets.

C. Model Architectures

1) Transformer Model: The Transformer architecture, introduced by Vaswani et al. [1], is entirely attention-based and avoids recurrence. Key components include:

- **Encoder**: Composed of multi-head self-attention layers and feedforward layers, with positional encoding to capture token order.
- **Decoder**: Combines self-attention and encoder-decoder attention to generate translations.
- **Hyperparameters**: Tuned parameters include $d_{model} = 512$, $n_{head} = 8$, 3 encoder and decoder layers, dropout rate = 0.1, and learning rate = 0.0005.

2) LSTM Model: The LSTM-based Seq2Seq model captures temporal dependencies in sequential data. Its components are:

- **Encoder**: Encodes input sentences into a fixed-length context vector.
- **Decoder**: Decodes the context vector to generate target translations.
- **Attention Mechanism**: Not implemented but could improve alignment and translation quality.
- **Hyperparameters**: Embedding size = 256, hidden size = 512, 2 layers, dropout rate = 0.1, learning rate = 0.0005.

D. Training Procedure

Both models were trained using the Adam optimizer and cross-entropy loss, ignoring padding tokens. Key training strategies included:

- Learning rate scheduling with a decay factor of 0.95.
- Early stopping to prevent overfitting.
- Batch size of 32 for efficient training.
- Training over 100 epochs.

III. Results and Analysis

A. Training and Validation Loss

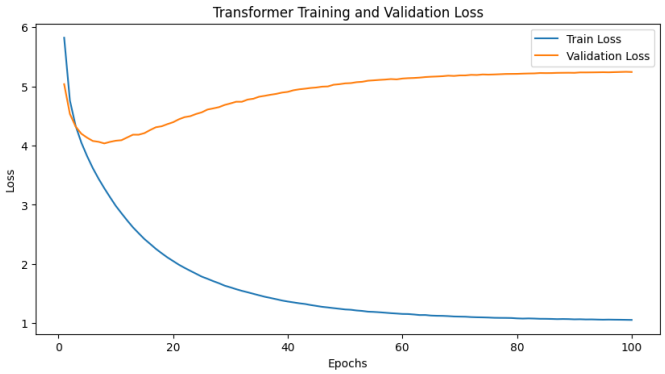

Fig. 1: Transformer Training and Validation Loss

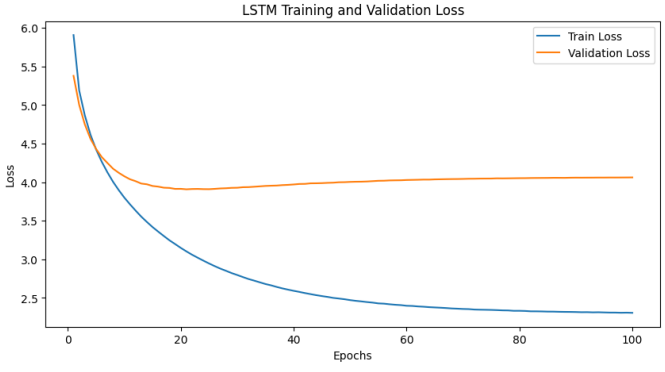

Fig. 2: LSTM Training and Validation Loss

B. Translation Examples

A sample translation comparison is illustrated in Figure 3.

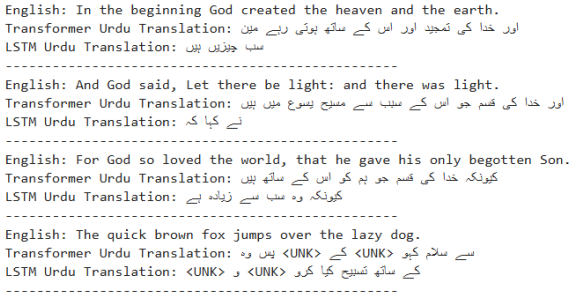

Fig. 3: Comparison of Translations: Transformer vs LSTM

C. Evaluation Metrics

- **BLEU Score**: Transformer = 1.77, LSTM = 0.27.
- **ROUGE-L Score**: Both models achieved negligible scores, reflecting challenges in alignment and structural coherence.
- **Perplexity**: Transformer = 56.67, LSTM = 49.81.
- **Inference Time**: Transformer = 0.0974 seconds/sentence, LSTM = 0.0088 seconds/sentence.

IV. Discussion

The Transformer model outperformed the LSTM in BLEU scores, indicating superior translation quality. However, both models struggled with low ROUGE-L scores, suggesting challenges in capturing linguistic nuances and maintaining sentence structure. The LSTM model's faster inference makes it suitable for resource-constrained environments.

A. Challenges

- Limited dataset size and linguistic complexity of Urdu.
- Presence of out-of-vocabulary (`<UNK>`) tokens.
- Insufficient structural alignment in translations.

B. Future Improvements

- Incorporating pre-trained multilingual models like mBERT or mBART.
- Using advanced tokenization techniques such as SentencePiece.
- Adding attention mechanisms to the LSTM model.

V. Conclusion

This study demonstrated the implementation of Transformer and LSTM models for English-Urdu translation using the UMC005 Parallel Corpus. While the Transformer achieved better translation quality, the LSTM provided faster inference. Future work will focus on enhancing data quality, incorporating pre-trained models, and experimenting with hybrid architectures.

References

[1] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., & Polosukhin, I. (2017). *Attention is All You Need*. Advances in Neural Information Processing Systems, 30.

[2] Hochreiter, S., & Schmidhuber, J. (1997). *Long Short-Term Memory*. Neural Computation, 9(8), 1735-1780.

[3] Papineni, K., Roukos, S., Ward, T., & Zhu, W.-J. (2002). *BLEU: a Method for Automatic Evaluation of Machine Translation*. Proceedings of the 40th Annual Meeting on Association for Computational Linguistics.
[4] Lin, C.-Y. (2004). *ROUGE: A Package for Automatic Evaluation of Summaries*. Text Summarization Branches Out.