Quantum-Enhanced Part-of-Speech Tagger(QLSTM) Using Quantum Classical Hybrid Approach

Haris Hussain (21K-3357), Maaz Khan (21K-3365), Areeb Ul Haq (21K-3346)
Department of Computer Science and Engineering,
FAST-NUCES, Karachi, Pakistan
k213357@nu.edu.pk, k213365@nu.edu.pk, k213346@nu.edu.pk

***Abstract*—This project investigates integrating quantum computing into natural language processing tasks through a novel Quantum Long Short-Term Memory (QLSTM) model. The approach combines parameterized quantum circuits with classical LSTM architectures to enhance part-of-speech (POS) tagging performance. Implemented using PennyLane, Qiskit, and PyTorch, the QLSTM leverages quantum computation for expressive feature representation and computational efficiency. Comparative evaluations with a classical LSTM baseline demonstrate the QLSTM's potential advantages in learning and generalization. The results underscore the promise of quantum-enhanced natural language processing (QNLP) in advancing sequence modeling tasks.**

***Index Terms*—Quantum Computing, Qiskit, PennyLane, Long Short-Term Memory, Natural Language Processing, Part-of-Speech Tagging.**

I. Literature Review

Part-of-speech (POS) tagging is a foundational task in natural language processing (NLP), aiming to classify words within a sentence into their grammatical roles. Classical Long Short-Term Memory (LSTM) networks have proven effective in this domain due to their ability to model sequential dependencies. However, their performance is constrained by limitations in feature representation and computational demands.

The emerging field of quantum computing introduces new paradigms in machine learning, with quantum circuits offering unique advantages such as entanglement and superposition. Quantum-enhanced models, including Variational Quantum Circuits (VQCs), have demonstrated potential for tasks requiring complex feature extraction and efficient computation [1,2]. Building on the Quantum LSTM (QLSTM) framework proposed by Chen et al. [3], this project explores its application in POS tagging, integrating Qiskit and PennyLane for quantum circuit representation and optimization.

II. Methodology

A. Data Preparation

A tagged corpus of sentences serves as the dataset. Preprocessing steps include tokenization and creation of word-to-index and tag-to-index mappings for efficient numerical encoding. The dataset is divided into training and testing sets.

B. Model Architecture

Classical LSTM: Implemented using PyTorch, this model acts as the baseline for comparative analysis.

Quantum LSTM (QLSTM):

- *Quantum Circuit Representation:* Qiskit is utilized to design and visualize parameterized quantum circuits for feature extraction. PennyLane facilitates hybrid quantum-classical execution.
- *Integration:* The quantum circuit replaces neural components in the LSTM cell, enabling quantum-enhanced operations.

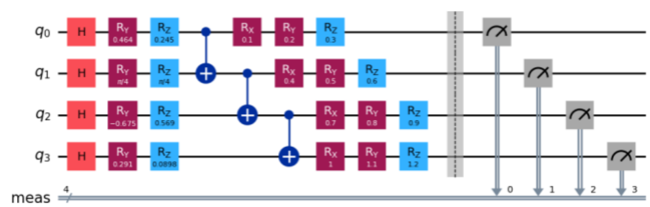

Fig. 1. Overview of the Quantum-Classical Hybrid Architecture used in the QLSTM model.

C. Training and Evaluation

Both models are trained using stochastic gradient descent (SGD) with a negative log-likelihood loss function. Accuracy metrics are monitored during training. Performance is evaluated on unseen data using accuracy and convergence time as metrics.

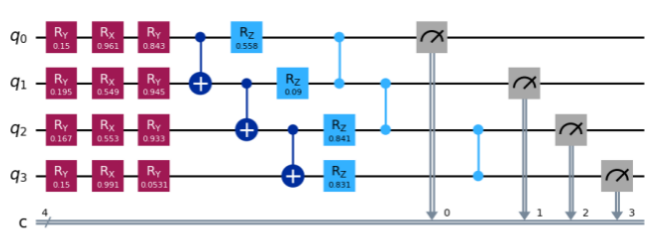

Fig. 2. Training and evaluation process for the Classical and Quantum LSTM models.

III. Results

Training results reveal the following:

- **Accuracy:** The QLSTM demonstrates competitive performance compared to the classical LSTM, showcasing enhanced ability to generalize grammatical patterns.

- **Training Dynamics:** While the QLSTM shows potential, it exhibits instability in accuracy and loss trends during training, likely due to noise in quantum computations.

Visualizations of loss and accuracy trends confirm these observations.

A. Accuracy Comparison

Table I compares the initial and final accuracy of both models, highlighting the QLSTM's instability in accuracy during training.

TABLE I
ACCURACY COMPARISON

Model	Initial Accuracy	Final Accuracy	Stability
Classical LSTM	0.2	1.0	Stable
Quantum LSTM	0.2	0.8	Unstable

B. Loss Comparison

Table II provides a comparison of the initial and final loss values for both models, emphasizing the instability in the QLSTM loss trend after 80 epochs.

TABLE II
LOSS COMPARISON

Model	Initial Loss	Final Loss	Stability
Classical LSTM	1.8	0.2	Stable
Quantum LSTM	1.8	0.3	Unstable

C. Conclusion from Tables

From the tables, it is evident that the classical LSTM outperforms the quantum LSTM in terms of accuracy stability and convergence, although the quantum LSTM achieves competitive loss values. Further refinements are needed for quantum-enhanced methods to match or exceed classical benchmarks consistently.

IV. CONCLUSION

This project successfully applies the QLSTM framework to POS tagging, integrating Qiskit for quantum circuit representation and PennyLane for hybrid computations. The Classical LSTM outperforms the Quantum LSTM in terms of accuracy and loss stabilization, while the Quantum LSTM, despite showing potential, suffers from instability, likely due to hyperparameter choices, training methodology, or inherent quantum noise. To improve stability and performance, further tuning of the Quantum LSTM is recommended, including adjustments to the learning rate, noise mitigation, and more robust quantum feature extraction. While the Classical LSTM provides reliable results and could be preferred for production use, the Quantum LSTM may become a viable alternative if it demonstrates significant improvements in future iterations.

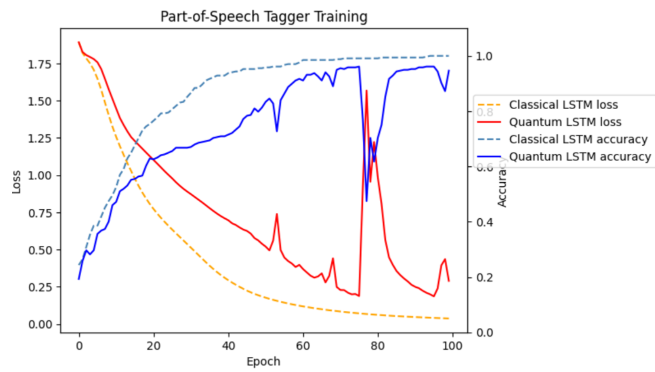

Fig. 3. Graphical summary of results and future directions for QLSTM research.

REFERENCES

[1] S. Y.-C. Chen, S. Yoo, and Y.-L. L. Fang, "Quantum Long Short-Term Memory," arXiv preprint arXiv:2009.01783, 2020.

[2] V. Havlíček *et al.*, "Supervised learning with quantum-enhanced feature spaces," Nature, vol. 567, no. 7747, pp. 209–212, 2019.

[3] P. Schuld and F. Petruccione, "Machine Learning with Quantum Computers," Springer, 2021.

[4] Qiskit Developers, "Qiskit: An Open-Source Framework for Quantum Computing," Qiskit Documentation, 2023.