Quantum Recurrent Architectures for Text Classification

Wenduan Xu, Stephen Clark, Douglas Brown, Gabriel Matos,
and **Konstantinos Meichanetzidis**
Quantinuum
17 Beaumont Street, Oxford, UK
{wenduan.xu,steve.clark,douglas.brown,gabriel.matos,kmei}@quantinuum.com

Abstract

We develop quantum RNNs with cells based on Parametrised Quantum Circuits (PQCs). PQCs can provide a form of hybrid quantum-classical computation where the input and the output is in the form of classical data. The previous "hidden" state is the quantum state from the previous time-step, and an angle encoding is used to define a (non-linear) mapping from a classical word embedding into the quantum Hilbert space. Measurements of the quantum state provide classical statistics which are used for classification. We report results which are competitive with various RNN baselines on the Rotten Tomatoes dataset, as well as emulator results which demonstrate the feasibility of running such models on quantum hardware.

1 Introduction

Recurrent neural networks (RNNs) were transformative in the early stages of neural NLP (Sutskever et al., 2014), and still offer competitive performance over more recent architectures such as Transformers (Orvieto et al., 2024). Now that quantum computing is also emerging as a potentially transformative technology (Preskill, 2018), it is natural to consider quantum versions of NLP models, such as RNNs, and ask whether they exhibit any advantages over their classical counterparts. Here we develop quantum RNNs with cells based on Parametrised Quantum Circuits (PQCs). PQCs can be used to provide a form of hybrid quantum-classical computation where the input and the output is in the form of classical data, and a set of parameters which controls the PQC's computation is classically optimised (Benedetti et al., 2019).

The excitement around quantum computing comes from the expectation that it will enable us to solve problems, or run models, efficiently which cannot be run efficiently on a classical computer (Nielsen and Chuang, 2000). The rapidly advancing state of quantum hardware means that we may soon cross the "quantum advantage" threshold (Preskill, 2012; Moses et al., 2023). Moreover, there are good reasons for exploring quantum models even when classically simulating them. Since PQCs implement unitary (norm-preserving) transformations, they are well-suited to modelling long-range dependencies and mitigating the problem of vanishing and exploding gradients (Arjovsky et al., 2016; Orvieto et al., 2024).

We propose quantum RNNs where the previous "hidden" state is the quantum state from the previous time-step, and the state is updated using one of two methods. In the first method, the current word is encoded as a quantum state using an angle encoding, and entangled with the previous state. In the second method, the word embedding is also transformed into a set of angles, but applied directly to a PQC which simply updates the previous state. Measurements are applied at the end of the sequence, and define a mapping from the quantum state to classical statistics which can be used for classification. Our models are hybrid in that words are first encoded as classical embeddings, and then mapped (non-linearly) into a quantum Hilbert space, via the angles of the PQC. Quantum phenomenon such as entanglement can then be exploited to manipulate the quantum state representations.

Here, we train the models in simulation, exploiting the fact that quantum computation can be simulated using linear algebra, and hence is amenable to optimisation with standard loss functions and backpropagation. We show that the quantum recurrent architectures can perform as well on a realistic NLP task as the classical RNN, GRU and LSTM baselines. Our results on the Rotten Tomatoes dataset are obtained with only 4 qubits, corresponding to a 16-d (2^4) complex Hilbert space, compared to the 200-d hidden states of the classical models. Since the models can be run in simulation, our main motivation is to begin exploring quantum NLP models in anticipation of more powerful quantum hardware

Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, pages 18020–18027
November 12-16, 2024 ©2024 Association for Computational Linguistics

in the future, as well as to provide a comparison with existing models in the classical case. However, in order to demonstrate the feasibility of running such models on quantum hardware even today, we also run the models at test time on a quantum computer *emulator*, obtaining results close to those of the exact, noise-free simulation.

2 Quantum Circuits

A quantum circuit consists of a sequence of quantum gates operating on *qubits* (Nielsen and Chuang, 2000). Qubits are quantum units of information which, unlike classical bits, can be in a *superposition* of 0 and 1. Mathematically the state $|\psi\rangle$ of a qubit is represented as a vector in a 2-d complex Hilbert space:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle = \begin{bmatrix}\alpha \\ \beta\end{bmatrix} \in \mathbb{C}^2. \tag{1}$$

The vectors $|0\rangle = \begin{bmatrix}1 & 0\end{bmatrix}^\top$ and $|1\rangle = \begin{bmatrix}0 & 1\end{bmatrix}^\top$ (written in Dirac notation on the left) form the *computational basis* for the Hilbert space. The *amplitudes* α and β are complex numbers satisfying $|\alpha|^2 + |\beta|^2 = 1$. The only way to access a quantum state is via a *measurement*, which is a probabilistic operation which returns a particular basis state. For a *measurement in the computational basis*, $|\alpha|^2$ is the probability of obtaining $|0\rangle$ and $|\beta|^2$ that of obtaining $|1\rangle$.

The information in qubits is carried by horizontal wires in the diagrammatic circuit notation, and multiple qubits are represented by stacking wires vertically (see Fig. 1). For a circuit with 2 qubits, measuring each qubit results in one of 4 possible outcomes: $|00\rangle$, $|01\rangle$, $|10\rangle$, or $|11\rangle$. Hence a quantum state $|\psi\rangle$ from 2 qubits is represented by a vector in a 4-d complex Hilbert space:

$$\begin{aligned}|\psi\rangle &= \alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|10\rangle + \alpha_{11}|11\rangle \\ &= \begin{bmatrix}\alpha_{00} & \alpha_{01} & \alpha_{10} & \alpha_{11}\end{bmatrix}^\top, \end{aligned}\tag{2}$$

where $\sum_{x\in\{0,1\}^2} |\alpha_x|^2 = 1$. The size of the Hilbert space is 2^n for n qubits and hence grows *exponentially* with the number of qubits.

Computation in a quantum circuit consists of transformations of the qubits. Such transformations have to produce state vectors with unit norm, and so have to be *unitary* transformations, with the property that $\mathbf{U}\mathbf{U}^\dagger = \mathbb{1}$ (where $\mathbf{U}^\dagger$ is the conjugate transpose of $\mathbf{U}$). An example of a single-qubit unitary is the quantum $\mathbf{NOT}$ or $\mathbf{X}$ gate, which has the matrix shown below (assuming the computational basis). Just as in the classical case, it flips the $|0\rangle$ state to $|1\rangle$ and vice versa, and for a state $|\psi\rangle$ in superposition it acts linearly:

$$\mathbf{X}|\psi\rangle = \begin{bmatrix}0 & 1 \\ 1 & 0\end{bmatrix}\begin{bmatrix}\alpha \\ \beta\end{bmatrix} = \begin{bmatrix}\beta \\ \alpha\end{bmatrix}. \tag{3}$$

Unlike $\mathbf{X}$, many other gates are parametric and contain parameters which can be learned via an external optimisation procedure. An example is the $\mathbf{RX}$ *rotation* gate with the unitary matrix:

$$\mathbf{RX} = \begin{bmatrix}\cos\left(\frac{\theta}{2}\right) & -i\sin\left(\frac{\theta}{2}\right) \\ -i\sin\left(\frac{\theta}{2}\right) & \cos\left(\frac{\theta}{2}\right)\end{bmatrix}, \tag{4}$$

with θ the degree of rotation about the x-axis (for the *Bloch sphere* representation of single-qubits).[1]

Quantum circuits are typically initialised with a simple, easy-to-prepare state such as the all-zero state (see Fig. 1). This state is a *product state*, meaning that the measurement outcomes on each wire are independent. Gates operating on more than one wire allow the creation of correlations between measurement outcomes, i.e. the creation of *entangled states*. Entanglement is necessary to fully exploit the exponentially-sized Hilbert space, and can be achieved through the use of *controlled gates*, consisting of a *control* qubit and a *target* qubit. A standard example is a controlled $\mathbf{X}$ gate ($\mathbf{CX}$), which acts as the identity on the computational basis states $|00\rangle$ and $|01\rangle$, but flips the second qubit when acting on $|10\rangle$ and $|11\rangle$:

$$\begin{aligned}&\alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|10\rangle + \alpha_{11}|11\rangle \xmapsto{\mathbf{CX}} \\ &\alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|11\rangle + \alpha_{11}|10\rangle\end{aligned}$$

The PQC we use in Fig. 1 has examples of controlled rotation gates ($\mathbf{RX}$ and $\mathbf{RY}$), where if the control is $|0\rangle$ then nothing happens to the target, and if the control is $|1\rangle$ then the rotation is applied; and for a state in superposition, the gates act linearly on the basis states.

A quantum state can be measured using a specific basis, such as the Pauli-$\mathbf{Z}$ measurement in the computational basis, yielding a scalar value (e.g. for a single qubit, +1 can be associated with the outcome $|0\rangle$ and -1 with $|1\rangle$). On actual quantum hardware, because the measurement outcome is

[1] To describe general single-qubit unitary transformations, it is useful to consider the Bloch sphere (Nielsen and Chuang, 2000) which can be used to represent single-qubit gates in terms of rotations of the sphere.

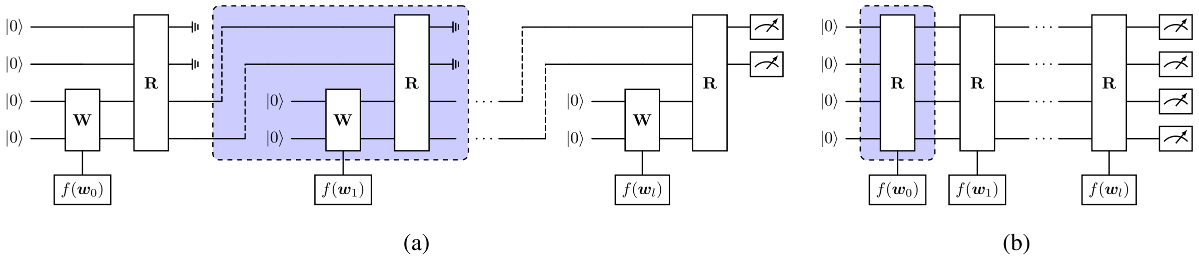


Figure 1: The quantum recurrent models. f is a classical affine transformation; $\mathbf{W}$ in (a) is the word ansatz (with $\mathbf{RX}$, $\mathbf{RZ}$, and $\mathbf{RX}$ rotation gates on each wire); $\mathbf{R}$ denotes the ansatz shown in Fig. 2 in both models; $l+1$ is the sentence length. Output from measuring 2 wires shown in (a) and the possibility of measuring all 4 wires in (b).

probabilistic, the circuit is run many times and an average is taken (where each run is referred to as a *shot*), giving an output in $[-1, 1]$. This output can then form part of a classical ML objective function.

3 Quantum Recurrent Models

The first model performs recurrent computation as shown in Fig. 1a. At each time-step, the word is encoded using a word *ansatz* $\mathbf{W}$,[2] using two of the wires, always starting with the all-zero state. More concretely, each word is represented as a classical word embedding $\boldsymbol{w}_i \in \mathbb{R}^m$, which gets mapped via an affine transformation f into the parametrised angles required by the rotation gates in $\mathbf{W}$:

$$f : \boldsymbol{w}_i \mapsto \boldsymbol{\theta}_i, \tag{5}$$

with $\boldsymbol{\theta}_i \in \mathbb{R}^n$ and n the number of angles. This produces a state on the two wires, which is combined with the previous state using a tensor product, resulting in a product state. All 4 wires are then put through another ansatz $\mathbf{R}$, which also has a set of trainable and parametrised angles, and produces an entangled state. The nature of a unitary transformation means that the number of wires going into each ansatz has to be the same as the number coming out, so 2 wires are "*ignored*", or "*discarded*", shown with the ground symbol.

The second model is simpler, and results from dispensing with $\mathbf{W}$ (Fig. 1b). Since there is no separate input state for each word, there is no need for discarding. The ansatz at each time-step still has its parameters supplied by a classical transformation of a word embedding. Measurements are taken of the final state and post-processed to give a classical output. Here we measure two qubits and use the scalar values as logits for a softmax (Fig. 1a).[3]

There are two ways in which quantum computation can be simulated on a classical computer. The first is to perform shot-based probabilistic computing, possibly including the modelling of noise from the imperfect nature of current quantum devices. Such simulators are often referred to as *emulators*. The second approach is to perform exact simulation, by calculating the measurement outcomes analytically (Nielsen and Chuang, 2000). Here we use exact, noiseless simulation for training, making use of automatic differentiation tools. At test time we use exact simulation and in addition perform runs on a quantum computer emulator.

4 Experiments

We use `TorchQuantum` (Wang et al., 2022) for analytical simulation – a PyTorch-based toolkit for developing hybrid classical-quantum models with batching and automatic differentiation support. The dataset is Rotten Tomatoes, a binary classification sentiment analysis dataset.[4] As baselines we include results for an RNN, GRU and LSTM. Some modifications were needed for `TorchQuantum`, for example the addition of density matrices and some related operations such as the partial trace (Nielsen and Chuang, 2000) to allow discarding.[5]

Here we provide the most important architectural details; hyperparameters are listed in Appendix B. We use 4 qubits for all models and, for the model with discarding, each word is encoded with a sequence of parameterised $\mathbf{RX}, \mathbf{RZ}, \mathbf{RX}$ gates on each of the bottom two wires (Fig. 1a), and so $\boldsymbol{\theta}_i \in \mathbb{R}^6$. We use the ansatz shown in Fig. 2

[2] An ansatz is a sequence of gates applied to specific wires.

[3] There are other options, e.g. measuring all 4 qubits (Fig. 1b) and transforming those 4 classical values into logits.

[4] https://huggingface.co/datasets/rotten_tomatoes. A balanced dataset of 8,530 sentences for training, 1,066 for dev and 1,066 for test.

[5] Our code is available at https://github.com/CQCL/QRNN-Sentiment.

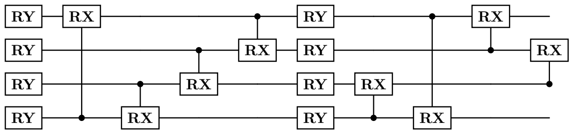


Figure 2: The choice of ansatz applied at each time-step – ansatz 14 from Sim et al. (2019).

($\boldsymbol{\theta}_i \in \mathbb{R}^{16}$) for $\mathbf{R}$ (in both the model with and without discarding), taken from Sim et al. (2019) which demonstrates experimentally that this ansatz is expressive, meaning that it has high entangling capability and can represent a large part of the Hilbert space. Note that for all rotation gates in $\mathbf{W}$ and $\mathbf{R}$, each gate has one classical parameter θ associated with it (4).

For the model with discarding, we found that using the classical word embedding to provide the parameters for the $\mathbf{R}$ as well as the $\mathbf{W}$ ansatze, via a separate mapping g, resulted in higher accuracies. Hence the models have trainable parameters associated with the following components: word embeddings (randomly initialised); the mapping f from word embeddings to angles for word encoding; the mapping g from word embeddings to angles for recurrent encoding; and the output classification function. For the f and g functions we use a single-layer affine transformation, and for the output function a 2-class softmax. Dropout is applied to the input embeddings (before f and g), and the loss function is binary cross-entropy.

Experiments were run on a single A100/RTX-6000 GPU. Table 1 shows the results, including internal and published baselines. Here we chose QRNN models which performed best on the development data which were then appplied to the test data. Overall we found little difference between the discard and no-discard models, with the no-discard model performing slightly better with the experimental configurations used here. The QRNN models are performing competitively compared with the classical baselines. What is particularly striking is the difference between the numbers of parameters used in the classical vs. the quantum models.

Appendix. A contains some additional results showing how the accuracy varies across different training runs, and also learning curves for the various models. The QRNN models are less stable than the classical baselines, showing more variation across runs, but convergence is much faster.

For the emulator results, we use the cloud-based native `H1-1E` simulator of the Quantinuum H1 quantum processor, including a realistic noise model, accessed through Quantinuum Nexus.[6] We took two of the best-performing QRNN and QRNN$_d$ models and obtained 80.1% and 77.2% test accuracies on the emulator, respectively, with 1,000 shots used for measurements. In comparison, the classical results for these particular models were 79.5% for both. Given the highly accurate nature of the emulator, these results are encouraging: they indicate that, even with the nascent state of quantum computing technology today, we already have hardware that can run well-performing NLP models at a reasonable scale.

Model	Dev	Test	$\|h\|$	$\|e\|$	$\|\theta\|$
QRNN	76.9	78.7	16	100	1.6K
QRNN$_d$	76.5	78.3	16	100	2.2K
RNN	76.9	76.8	200	100	60K
GRU	78.1	77.2	200	100	180K
LSTM	78.0	78.5	200	100	240K
LSTM†	-	79.7	1,024	512	6M
LM-LSTM	-	78.3	-	-	-
SA-LSTM	-	79.7	-	-	-

Table 1: Classification accuracy on Rotten Tomatoes. $|h|$ = hidden state size; $|e|$ = embedding dimension; $|\theta|$ = parameter count; QRNN$_d$: with discarding. LSTM† is the LSTM with tuning and dropout, and LM-LSTM and SA-LSTM apply pretraining to LSTM†, all from Dai and Le (2015).

5 Related Work

A notable contribution to the development of QRNNs is the work of Bausch (2020), which provides a rigorous theoretical framework for quantum-enhanced sequential learning and highlights the need for quantum hardware to fully realize the potential of QRNNs. In this paper, Bausch explores the potential for quantum models to outperform classical RNNs, particularly in scenarios involving long-range dependencies and complex temporal correlations. Bausch's work introduces a novel class of QRNNs designed to leverage the key principles of quantum mechanics—such as superposition and entanglement—to process sequential data. Another key insight from Bausch's work is the demonstration that QRNNs can be trained using fewer iterations compared to classical RNNs, as we also demonstrate.

[6] https://nexus.quantinuum.com/

Another important development in the QRNN literature is the Quantum Long Short-Term Memory (QLSTM) model, which extends the classical LSTM architecture to the quantum domain. Proposed by Chen et al. (2022), the QLSTM model used quantum gates to represent the forget, input, and output gates of classical LSTMs, allowing a hybrid model while retaining the potential to address the vanishing gradient problem. This work demonstrated that QLSTMs could outperform classical LSTMs on certain benchmarks, achieving better accuracy and faster convergence.

The hybrid QRNN model of Yu et al. (2024) also explored a text classification task, although only in the low-resource setting. Other QRNNs have also been proposed (Li et al., 2023; Chen and Khaliq, 2024; Siemaszko et al., 2023). Li et al. also apply their model to a text categorization task, although our model architectures are different, and use novel ansatze and a discarding operation. The latter two works use different circuits and are particularly relevant for tasks in the continuous-variable domain.

NLP has also used quantum models (Wu et al., 2021; Widdows et al., 2024), often based on the observation that the mathematics of quantum Hilbert spaces shares similarities with vector space models of semantics (Clark, 2015). One focus has been on compositional distributional models of language (Coecke et al., 2010), with a recent demonstration of solving a toy NLP problem on quantum hardware (Lorenz et al., 2023).

Harvey et al. (2023) also investigate PQC-based architectures for sequence modelling, including some test runs on actual quantum hardware. Various architectures are considered, including some based on syntactic structure. They also report results based on the Rotten Tomatoes dataset, obtaining their best results using a convolution structure.

6 Conclusion

We have described quantum RNNs and simulation experiments, on a realistic NLP task, obtaining results that are competitive with some challenging classical RNN baselines. We believe this is the first paper published at an NLP conference which demonstrates a QNLP model being applied successfully to such a task and the running of such a model on a quantum emulator. One noteworthy feature of the models is the relatively small number of learnable parameters. Future work includes scaling up the number of qubits, and to larger datasets, with the potential to use anzatze which cannot be efficiently simulated on a classical computer (with the caveat that trainability could be an issue; see the Limitations section below).

Although the main motivation has been to investigate the potential of quantum models in anticipation of the development of (rapidly improving) quantum hardware, the models can also be thought of bringing advantages to existing classical models, such as the naturally bounded gradients from the unitary transformations (Arjovsky et al., 2016).

Author Contributions

WX designed and implemented the models and did all the simulation experiments. SC supervised the project. SC and WX wrote the paper. GM and DB implemented emulator code, and GM ran the emulator experiments for this paper. WX, DB and GM conceived of a basic version of the QRNN. KM helped initiate the project and conceived of the baseline discarding model. Everyone contributed to the discussions during the first phase of the project.

Acknowledgements

Thanks to Richie Yeung and Carys Harvey for sharing their JAX implementation of the "stairs" reader. Thanks also to the Oxford Compositional Intelligence team for feedback on this work.

Limitations

The training has been carried out using exact analytical simulation, with the models only being run on the emulator at test time. Optimisation on a quantum computer is non-trivial because of the lack of a quantum equivalent of the backpropagation algorithm (Benedetti et al., 2019). Also, there are theoretical results to suggest that, as the number of qubits grows, the loss landscape becomes flat almost everywhere resulting in "barren plateaus", which makes gradient-based optimisation extremely challenging (McClean et al., 2018). However, research in this area is ongoing (Larocca et al., 2024).

The results are not comparable to the state of the art in NLP, but we argue that this is perfectly reasonable given the nascent state of quantum computing. We have also only applied the model to one dataset in one language, but again this is reasonable given that the goal of the work is to demonstrate the feasibility of large-scale hybrid quantum-classical models in the future, where the models can also be adapted to other datasets and tasks, as well as the pre-training paradigm which has proved so successful in NLP (Peters et al., 2018).

References

Martin Arjovsky, Amar Shah, and Yoshua Bengio. 2016. Unitary evolution recurrent neural networks. In *ICML'16: Proceedings of the 33rd International Conference on International Conference on Machine Learning*, volume 48, page 1120–1128.

Johannes Bausch. 2020. Recurrent quantum neural networks. In *Advances in Neural Information Processing Systems*, volume 33, pages 1368–1379. Curran Associates, Inc.

Marcello Benedetti, Erika Lloyd, Stefan Sack, and Mattia Fiorentini. 2019. Parameterized quantum circuits as machine learning models. *Quantum Science and Technology*, 4(4).

Samuel Yen-Chi Chen, Shinjae Yoo, and Yao-Lung L. Fang. 2022. Quantum long short-term memory. In *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8622–8626.

Yuan Chen and Abdul Khaliq. 2024. Quantum recurrent neural networks: Predicting the dynamics of oscillatory and chaotic systems. *Algorithms*, 17(4).

Stephen Clark. 2015. Vector space models of lexical meaning. In *The Handbook of Contemporary Semantic Theory*, 2nd edition, chapter 16, pages 493–522. Wiley Blackwell.

Bob Coecke, Mehrnoosh Sadrzadeh, and Stephen Clark. 2010. Mathematical foundations for a compositional distributional model of meaning. *Linguistic Analysis*, 36(1-4):345–384.

Andrew M. Dai and Quoc V. Le. 2015. Semi-supervised sequence learning. In *Advances in Neural Information Processing Systems, NIPS*.

Carys Harvey, Richie Yeung, and Konstantinos Meichanetzidis. 2023. Sequence processing with quantum tensor networks. https://arxiv.org/abs/2308.07865.

Martin Larocca, Supanut Thanasilp, Samson Wang, Kunal Sharma, Jacob Biamonte, Patrick J. Coles, Lukasz Cincio, Jarrod R. McClean, Zoe Holmes, and M. Cerezo. 2024. A review of barren plateaus in variational quantum computing. https://arxiv.org/abs/2405.00781.

Yanan Li Li, Zhimin Wang, Rongbing Han, Shangshang Shi, Jiaxin Li, Ruimin Shang, Haiyong Zheng, Guoqiang Zhong, and Yongjian Gu. 2023. Quantum recurrent neural networks for sequential learning. *Neural Networks*, 166:148–161.

Robin Lorenz, Anna Pearson, Konstantinos Meichanetzidis, Dimitri Kartsaklis, and Bob Coecke. 2023. QNLP in practice: Running compositional models of meaning on a quantum computer. *Journal of Artificial Intelligence Research*, 76.

Jarrod R. McClean, Sergio Boixo, Vadim N. Smelyanskiy, Ryan Babbush, and Hartmut Neven. 2018. Barren plateaus in quantum neural network training landscapes. *Nature Communications*, 9(1).

S. A. Moses, C. H. Baldwin, M. S. Allman, R. Ancona, L. Ascarrunz, C. Barnes, J. Bartolotta, B. Bjork, P. Blanchard, M. Bohn, J. G. Bohnet, N. C. Brown, N. Q. Burdick, W. C. Burton, S. L. Campbell, J. P. Campora III au2, C. Carron, J. Chambers, J. W. Chan, Y. H. Chen, A. Chernoguzov, E. Chertkov, J. Colina, J. P. Curtis, R. Daniel, M. DeCross, D. Deen, C. Delaney, J. M. Dreiling, C. T. Ertsgaard, J. Esposito, B. Estey, M. Fabrikant, C. Figgatt, C. Foltz, M. Foss-Feig, D. Francois, J. P. Gaebler, T. M. Gatterman, C. N. Gilbreth, J. Giles, E. Glynn, A. Hall, A. M. Hankin, A. Hansen, D. Hayes, B. Higashi, I. M. Hoffman, B. Horning, J. J. Hout, R. Jacobs, J. Johansen, L. Jones, J. Karcz, T. Klein, P. Lauria, P. Lee, D. Liefer, C. Lytle, S. T. Lu, D. Lucchetti, A. Malm, M. Matheny, B. Mathewson, K. Mayer, D. B. Miller, M. Mills, B. Neyenhuis, L. Nugent, S. Olson, J. Parks, G. N. Price, Z. Price, M. Pugh, A. Ransford, A. P. Reed, C. Roman, M. Rowe, C. Ryan-Anderson, S. Sanders, J. Sedlacek, P. Shevchuk, P. Siegfried, T. Skripka, B. Spaun, R. T. Sprenkle, R. P. Stutz, M. Swallows, R. I. Tobey, A. Tran, T. Tran, E. Vogt, C. Volin, J. Walker, A. M. Zolot, and J. M. Pino. 2023. A race track trapped-ion quantum processor. *Phys. Rev. X*, 13:041052.

Michael A. Nielsen and Isaac L. Chuang. 2000. *Quantum Computation and Quantum Information*. Cambridge University Press.

Antonio Orvieto, Samuel L. Smith, Albert Gu, Anushan Fernando, Caglar Gulcehre, Razvan Pascanu, and Soham De. 2024. Resurrecting recurrent neural networks for long sequences. https://arxiv.org/pdf/2303.06349.pdf.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

John Preskill. 2012. Quantum computing and the entanglement frontier. arXiv:1203.5813. Rapporteur talk at the 25th Solvay Conference on Physics - The Theory of the Quantum World.

John Preskill. 2018. Quantum Computing in the NISQ era and beyond. *Quantum*, 2:79.

Model	Max	Min	μ
QRNN	79.3	53.8	75.5
QRNN$_d$	79.6	58.2	72.1
RNN	79.1	70.7	76.9
GRU	80.0	76.1	78.1
LSTM	80.0	76.2	78.0

Table 2: The max, min and average (μ) classification accuracy on Rotten Tomatoes test set across 100 runs. QRNN$_d$: with discarding.

M. Siemaszko, A Buraczewski, B Le Saux, and M Stobinska. 2023. Rapid training of quantum recurrent neural networks. *Quantum Mach. Intell.*, 5(32).

Sukin Sim, Peter D. Johnson, and Alan Aspuru-Guzik. 2019. Expressibility and entangling capability of parameterized quantum circuits for hybrid quantum-classical algorithms. *Advanced Quantum Technologies*, 2(12).

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 27.

Hanrui Wang, Yongshan Ding, Jiaqi Gu, Zirui Li, Yujun Lin, David Z Pan, Frederic T Chong, and Song Han. 2022. QuantumNAS: Noise-adaptive search for robust quantum circuits. In *The 28th IEEE International Symposium on High-Performance Computer Architecture (HPCA-28)*.

Dominic Widdows, Willie Aboumrad, Dohun Kim, Sayonee Ray, and Jonathan Mei. 2024. Quantum natural language processing. https://arxiv.org/abs/2403.19758.

Sixuan Wu, Jian Li, Peng Zhang, and Yue Zhang. 2021. Natural language processing meets quantum physics: A survey and categorization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3172–3182, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Wenbin Yu, Lei Yin, Chengjun Zhang, Yadang Chen, and Alex Liu. 2024. Application of quantum recurrent neural network in low resource language text classification. *IEEE Transactions on Quantum Engineering*, 5.

A Additional Results

The results in Table 1 were obtained across 100 runs after the hyperparameters are tuned on the dev set. To account for random seeds, we also report here the min, max, and average accuracies on the test set in Table 2, for completeness.

Fig. 3 shows the learning curves on the dev set for the QRNN models and internal baselines in Table 1. As can be seen, the QRNNs learn relatively quickly, with only a small number of epochs required to converge, while the classical baselines typically needed at least an order of magnitude more epochs to converge, with the choice of hyperparameters shown below.

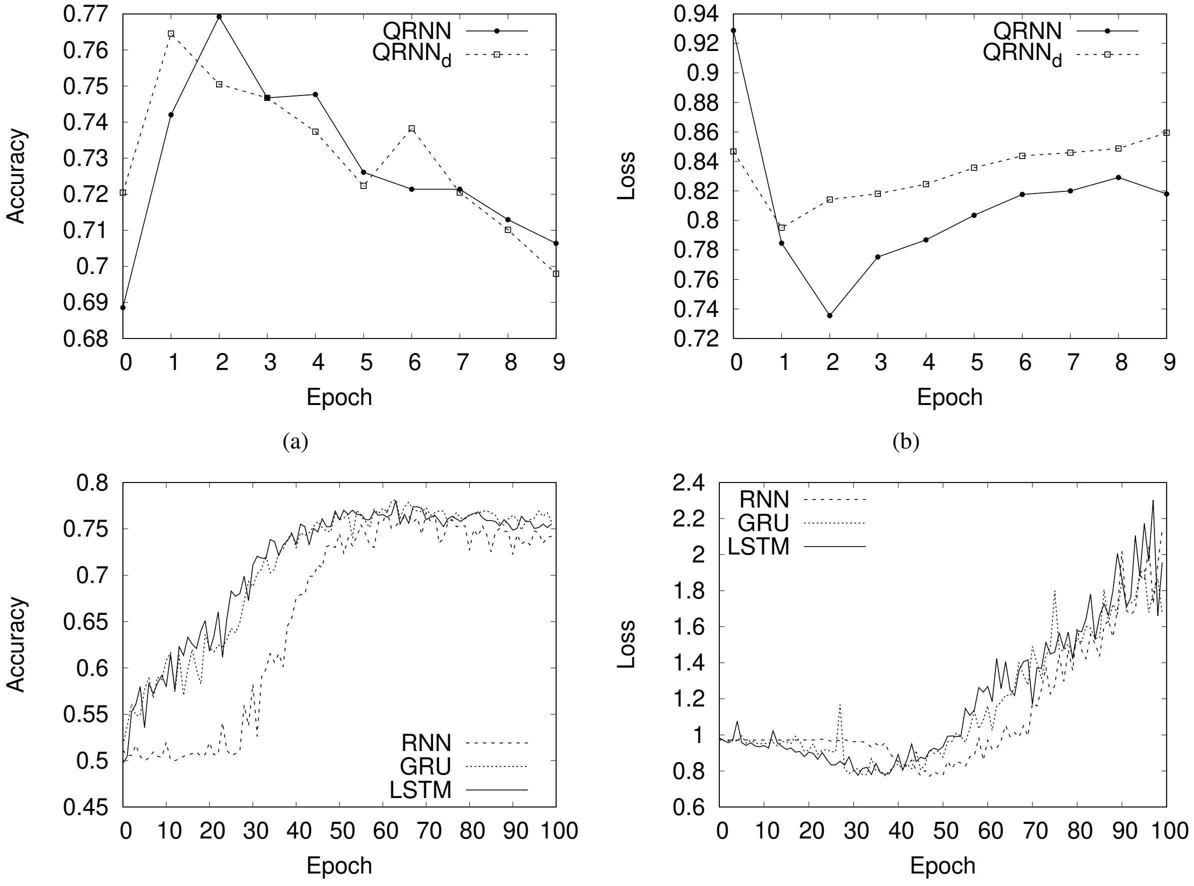
B Hyperparameters

The same hyperparameters are used for both the QRNN and QRNN$_\text{d}$ models:

Optimiser	Adam
Learning rate	$1e^{-3}$
Weight decay	$1e^{-4}$
ϵ	$1e^{-10}$
|Batch|	200
|Embedding|	100
Embedding Dropout	0.2
Embedding init	Xavier uniform $(gain = 1.0)$
Max grad norm	5.0

For the RNN, GRU and LSTM baselines, the hyperparameters are:

Optimiser	Adam
Learning rate	$1e^{-3}$
Weight decay	$1e^{-4}$
ϵ	$1e^{-10}$
|Batch|	200
|Embedding|	100
Embedding dropout	0.7
Embedding init	$\mathcal{N}(0,1)$
Max grad norm	1.0



(a)

(b)

(c)

(d)

Figure 3: Accuracy and loss curves (on the dev set) for the QRNN and QRNN$_d$ models and the internal baselines in Table 1.