Random Quantum Circuits as 3D Convolutional Kernels for 3D SAS ATR

Chris Su
Carnegie Mellon University
Pittsburgh, PA
chrissu@andrew.cmu.edu

Greg Vetaw
Arizona State University
Tempe, AZ
gvetaw@asu.edu

Suren Jayasuriya
Arizona State University
Tempe, AZ
sjayasur@asu.edu

Glen Uehara
SenSIP Center
Arizona State University
Tempe, AZ
guehara@asu.edu

Andreas Spanias
SenSIP Center
Arizona State University
Tempe, AZ
aspanias@asu.edu

***Abstract*—We propose the applications of quantum circuits for 3D synthetic aperture sonar (SAS) data classification. Our work focuses on reducing model parameters and increasing inference efficiency. Specifically, our work proposes quantum circuits as a first-layer replacement of classical 3D convolutional layers in 3D SAS classification. We investigate our method with volumetric SAS data cubes from the Sediment Volume Search Sonar (SVSS) dataset.**

***Index Terms*—synethetic aperture sonar, cnn, quantum computing, quantum computer vision**

I. Introduction

Quantum machine learning (QML) has emerged as a promising field that integrates quantum computing and machine learning to address complex data processing challenges. With increasing quantum fidelity in recent years and the entering of a NISQ era, quantum machine learning algorithms that can exploit quantum supremacy with limited qubits see increasing relevance. Henderson et. al introduced random 2D quantum kernels as convolution layers in quantum-classical hybrid architectures [1]. Their method tested and validated on the MNIST digits dataset. Our work focuses on generalizing this approach to 3D data cubes. 3D data problems sees increasing relevance in robotics, augmented reality, virtual reality, and sonar/radar applications. The goal of this paper is to: (1) propose a 3D quantum kernel, and (2) analyze trade-offs between accuracy metrics (AUC-PR) and efficiency for CNN-based ATR, particularly for 3D SAS classification. The Sediment Volume Search Sonar (SVSS) dataset is a particularly challenging SAS dataset, due to targets being buried in the sub-bottom, which often return weaker acoustic signals compared to the clutter within the sediment.

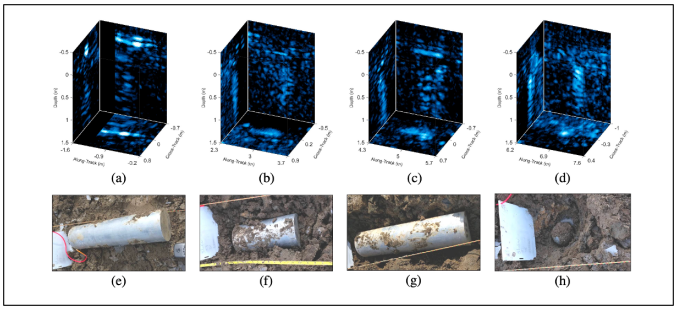

Fig. 1. SVSS 3-D Data Cubes, *Brown et. al.* [3]

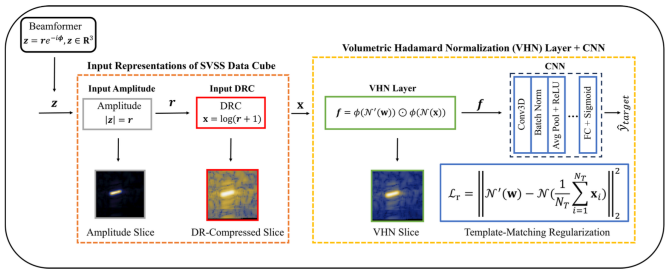

Fig. 2. Preprocessing with SOTA CNN, *Vetaw et. al.* [2]

Furthermore, the SVSS dataset is heavily imbalanced so a high performing model would have to be able to generalize well across different imbalance ratios. The data cubes also exhibit high dynamic range, that cause common dynamic range compressing techniques such as thresholding or log compression to lose important contrasting features. All these factors pose significant and unique challenges towards the development of efficient and high performing machine learning algorithms [2].

Brown et. al. showed that convolutional neural network architectures achieved competitive performance compared to a preescreening detection algorithm, additionally emphasizing the importance and utility of memory and computationally efficient classification of 3D SAS data cubes [6]. Vetaw et al. introduced a novel learnable preprocessing layer that tone mapped 3D data cubes and showed that this layer could be

Identify applicable funding agency here. If none, delete this.

added to the CNN architecture introduced by Brown et. al. to improve performance [2].

Our work proposes 3D quantum preprocessing convolution kernels to act as a replacement first layer for classical convolutions in CNN's, thereby reducing the dimensionality of the data. Our specific contributions are: (1) introduce a classical-quantum hybrid CNN architecture that offloads a computationally expensive initial step to quantum computers; and (2) show that 3D quantum kernels can replace a first layer of the SOTA CNN to increase inference efficiency and reduce model parameter count. These methods were validated on a subset of the SVSS dataset. The hope of this research is to introduce quantum computing as a new computational resource in computer vision tasks and to encourage future research in this area.

II. Methods

In this section, we discuss the overall pipeline, the implementation of quantum kernels, the training details, and the SVSS dataset.

A. CNN Architectures

Several CNN architectures were implemented to validate our method. A 3D SOTA SAS binary classification CNN model from Brown et. al [6], which we refer to as CNN A; A volumetric hadamard normalization layer [3] was added to CNN A and referred to as CNN B. We also propose our own CNN architecture to work with the quantum convolution layer, which is referred to as CNN Q; A CNN architecture that implements a less classically computationally expensive approach to simulate quantum kernels, CNN QS; And lastly, CNN C, the purely classical counterpart of CNN Q and CNN QS that uses no quantum kernels, but instead resizes the SAS data cube to fit the input shape.

CNN A	CNN B	CNN C	CNN Q	CNN QS
CNN [6]	VHN Layer [2] with CNN A	Purely classical CNN	3D Quantum Kernels with CNN C	3D Sparse Quantum Kernels with CNN C

Fig. 3. A useful appendix / summary of the CNN architectures used in this paper.

B. Quantum Kernels

The same methodology was applied as in Henderson et. al [1] for the implementation of the random quantum kernels. Similar to classical kernels, one can vary the stride size of quantum kernels. For a perfect kernel of size $S = (k, k, k) \in \mathbb{N}$, we require k^3 qubits. For classical simulations of quantum circuits, this usually becomes intractible at around $(5, 5, 5)$, or 125 qubits. Our approach was tested on quantum kernels of size $S = (2, 2, 2)$, with stride $(2, 2, 2)$.

C. Sparse Quantum Kernels

Due to the computational intractability of large kernel sizes for classical simulations of quantum hardware, we also experimented with a "lattice" shaped kernel that used $\lfloor S/2 \rfloor$ qubits. Hence, for our purposes, classical simulations of quantum hardware only required 4 or 8 classical qubits. The

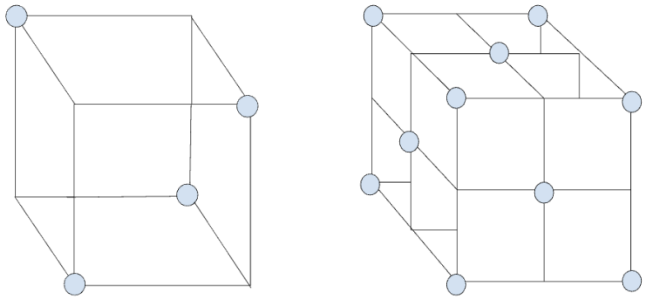

Fig. 4. Sparse Kernels, k = 2 (left), k = 3 (right)

implementation approach remained the same as the above non-sparse quantum kernels.

D. CNN C Implementation Details and Pipeline

Our base model to be used with quantum kernels were adapted from Brown et. al. to fit the smaller data cube dimension. Mainly, 3 convolution layers and 1 fully connected layer was used. Kernel sizes were $(3, 3, 3)$, with stride $(1, 2, 2)$, and padding $(0, 1, 1)$ for all three convolution layers. Average pooling was used after the first and second convolution layers, with kernel size $(6, 2, 2)$, stride $(1, 1, 1)$ and no padding. The ReLU activation function was used. All implementation was done in the Python PyTorch deep learning library. The cubes

Layers (In Order)	3D Conv1	Avg Pool	3D Conv2	Avg Pool	3D Conv3	FC 1	Output
Kernel Size	(3, 3, 3)	(6, 2, 2)	(3, 3, 3)	(6, 2, 2)	(3, 3, 3)	N/A	N/A
Stride	(1, 2, 2)	(1, 1, 1)	(1, 2, 2)	(1, 1, 1)	(1, 2, 2)	N/A	N/A
Padding	(0, 1, 1)	(0, 0, 0)	(0, 1, 1)	(0, 0, 0)	(0, 1, 1)	N/A	N/A

Fig. 5. CNN C Architecture (PyTorch)

were initially dynamically range compressed using log-DRC, then min-max normalized such that the entries were between 0 and 1. Specifically, the min-max normalization function $\mathcal{N}(\mathbf{x}) = \frac{\mathbf{x} - min(\mathbf{x})}{max(\mathbf{x}) - min(\mathbf{x})}$ was used for the data cubes.

E. Summary of Training Details

Training was on an Apple M2 CPU with 8 GB memory. The CNN architectures were implemented using the Python PyTorch Deep Learning Library, version 2.3.1. The quantum kernels were implemented in the Python Pennylane library, version 0.36.0. Figure 6 outlines the various configurations for the relevant CNN architectures for this paper.

Configuration	CNN A	CNN B	CNN C	CNN Q	CNN QS
Num Epochs	200	200	200	200	200
Batch Size	20	20	20	20	20
Cube Size	64 x 64 x 101	64 x 64 x 101	32 x 32 x 50	64 x 64 x 101	64 x 64 x 101
Optimizer	Adam	Adam	Adam	Adam	Adam
Betas (Adam, B1, B2)	0.9, 0.999	0.9, 0.999	0.5, 0.9	0.5, 0.9	0.5, 0.9
Learning Rate	0.002	0.002	0.0005	0.0005	0.0005

Fig. 6. Training details of CNNs

F. SVSS Dataset

The Sediment Volume Search Sonar (SVSS) collects data at different seasons which results in different sediment compositions as well as depths [2]. Several past experiments used both depths in their research, so our method adapted that approach [2], [3], [7], [9]. Due to the computationally expensive nature of 3D data problems, and even more so for quantum simulations, our method used only a subset of the SVSS dataset. The training data composed of 1000 data cubes that were measured during November, and the test data composed of 240 data cubes that were measured during June. Since the data varied between November and June, if the model performance was within an acceptable range, we could hypothesize that the model was generalizing the distribution from November data cubes well. The data was balanced between targets and non-targets.

III. Results

A. Model Size Reduction and Inference Efficiency

In this section, we outline the various inference times and model size reductions for the various CNN classical and hybrid architectures.

Architecture	CNN A	CNN B	CNN C	CNN Q	CNN QS
MACs	4.8 x 10^8	4.9 x 10^8	4.9 x 10^7	2.3 x 10^8	1.2 x 10^8
Parameter Count	2,342	416,038	2,521	3277	2845

Fig. 7. Model Sizes and MAC count

We used an open source Python Library, torchprofile, to analyze the MACs (multiply-accumulate operations) for each CNN architecture. The least computationally expensive architecture was the base CNN C architecture, followed by CNN QS and CNN Q. As initially hypothesized, we found that the hybrid architectures required less MACs when compared to the SOTA (purely) classical architectures. This was reflected in experimental running times, which have not been included in the figure. The hybrid architectures, CNN Q and CNN QS, were on the same order of parameter size as classical architectures CNN A and CNN C. CNN B had a significant parameter size due to the initialization of leaernable weights that had equivalent dimensions to the input size, (64, 64, 101).

B. Architecture Performances

In this section, we outline the performances of the various CNN architectures outlined in this paper. Namely, we found that using 3D quantum kernels to replace the first convolution layer in classical architectures did not hinder the model performance. The models' performances were evaluated using

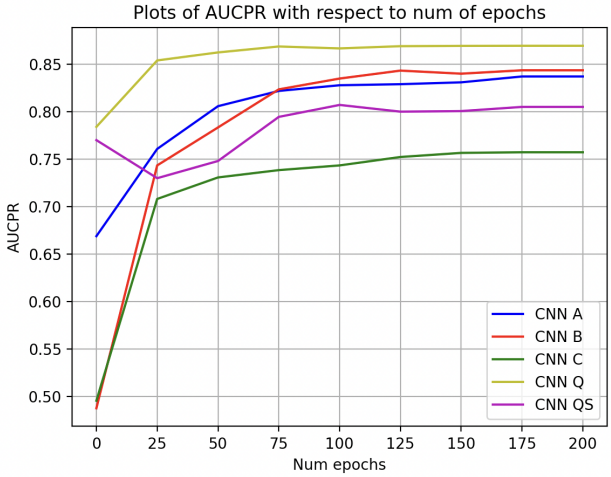

Fig. 8. AUCPR scores across the CNN architectures. Architectures were tested every 10 epochs during a training run of 200 epochs.

the Area-Under-Curve of Precision-Recall metric (AUC-PR). We used the scikit package for AUC-PR calculations [8]. The models were tested on 240 data cubes from a balanced June dataset. The hybrid architecture, CNN Q, had the best performance (AUC-PR = 0.868), followed by CNN B and CNN A (AUC-PR = 0.840, AUC-PR = 0.835, respectively). When analyzing the role of strictly the 3D quantum kernels in

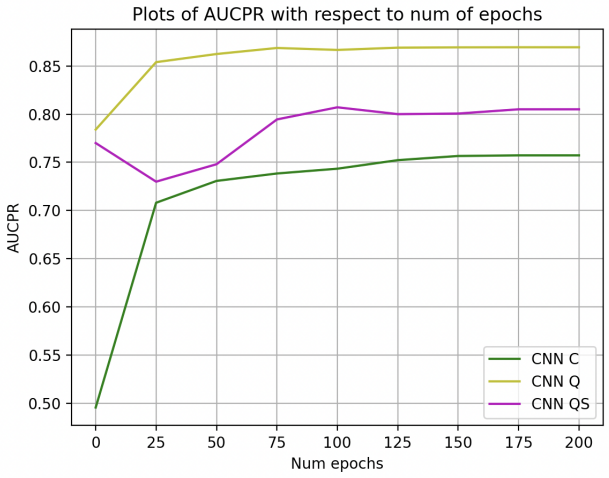

Fig. 9. This figure shows the hybrid architectures as well as the baseline architecture for Q and QS, CNN C

the hybrid architectures, CNN Q and CNN QS, we found that CNN C performed better when fed data that had been initially

passed through quantum kernels (sparse or not) than without. That is, CNN Q and CNN QS exhibited higher AUC-PR scores than their classical counterpart, CNN C. Furthermore, the hybrid architectures showed increased performance when full kernels were used as opposed to using a sparse kernel. CNN Q regularly performed better than CNN QS. In the figure above, CNN Q has an AUC-PR = 0.868, while CNN QS exhibited an AUC-PR = 0.804.

IV. Conclusions

A. Resource Limitations

While the NISQ era allows for greater acceleration of certain problems, classically simulating quantum machinery is time consuming and requires powerful hardware. Access to quantum computers is also limited. Thus, our research faced several time and resource limitations. This resulted in our research being limited to a subset of the full SVSS dataset, which we hope to further investigate in the future.

B. Further research

While using sparse kernels decreased model performance, one clear result is that using quantum kernels to create hybrid architectures increases performance in this use case. An interesting question to investigate in the future is the plausibility for quantum kernels to replace first convolutions in other CNN architectures. Further research should also look into the usage of additional quantum convolution layers or using more filters / kernels, as also proposed by Henderson et. al [1].

C. Concluding remarks

Our work investigated the generalization for 2D quantum kernels introduced by Henderson et. al to 3D CNN architectures [1]. We showed that for the binary classification of data cubes in a subset of the SVSS dataset, that the model performance improves when using quantum kernels. Furthermore, we introduce a type of quantum kernel (sparse quantum kernels) that are more computationally efficient to use than full quantum kernels in hybrid architectures, without performing worse than the baseline. However, we believe more research is necessary to truly establish quantum supremacy in 3D binary classification problems of synthetic aperture sonar data. We hope that this paper outlines the potential of quantum computing as a promising resource in computer vision tasks.

Acknowledgments

A thank you to my mentors, Greg Vetaw, Glen Uehara, M.S., Suren Jayasuriya, Ph.D., and Andreas Spanias, Ph.D., as well as Ms. Robina Sayed. This research is in part funded by the NSF.

References

[1] Henderson, Maxwell P. et al. “Quanvolutional Neural networks: powering image recognition with quantum circuits.” *Quantum Machine Intelligence 2, 2019*

[2] G. Vetaw, “Volumetric Hadamard Normalization for Sub-Bottom SAS ATR” *[Submitted to the Journal of Oceanic Engineering]*

[3] D. Williams and D. Brown, “New target detection algorithms for volumetric synthetic aperture sonar data,” *Proc. of Meetings on Acoustics, vol. 40, p. 070002, Sept. 2020.*

[4] Uehara, G. S., Spanias, A., Clark, W.. ”Quantum information processing algorithms with emphasis on machine learning.” *In IEEE IISA, July 2021.*

[5] Miller, L., Uehara, G., Spanias, A. (2024, March). ”Quantum Image Fusion Methods for Remote Sensing.” *In 2024 IEEE Aerospace Conference (pp. 1-9). IEEE.*

[6] D. Williams and D. Brown, ”Three-dimensional convolutional neural networks for target clas- sification with volumetric sonar data,” *Proc. of Meetings on Acoustics*, vol. 44, p. 070005, Aug. 2021.

[7] T. Hoang, K. S. Dalton, I. D. Gerg, T. E. Blanford, D. C. Brown, and V. Monga, “Resonant scattering-inspired deep networks for munition detection in 3d sonar imagery,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 61, pp. 1–17, 2023.

[8] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, “Scikit-learn: Machine learning in Python,” *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[9] J. Davis and M. Goadrich, “The relationship between precision-recall and roc curves,” in *Proceedings of the 23rd International Conference on Machine Learning*, ser. ICML ’06. New York, NY, USA: Association for Computing Machinery, 2006, p. 233–240. [Online]. Available: https://doi.org/10.1145/1143844.1143874