On the potentials of Tensor-based Quantum Machine Learning for SAR land-cover classification

Sreejit Dutta[a], Sigurd Huber[a], Gerhard Krieger[a]
[a] Microwaves and Radar Institute (HR), German Aerospace Center (DLR), 82234 Wessling, Germany.

Abstract

Synthetic Aperture Radar (SAR) data, characterised by its high-dimensionality and complex spatial correlations, poses significant challenges in terms of efficient processing and meaningful interpretation. Classical algorithms, while effective, often struggle with the sheer volume and intricacy of the data. This paper introduces a novel approach employing tensor quantum machine learning (QML) to tackle the intricacies of SAR data. By harnessing the computational advantages of quantum mechanics and the representational efficiency of tensor networks, we try to achieve enhanced feature extraction and pattern recognition. We look at various Tensor decomposition schemes to reduce data dimensionality as well as Tensor based quantum circuits to perform land-cover classification. Preliminary results, based on simulations, demonstrate the potential of our tensor QML framework. For the scope of this research so far, we worked with simulated amplitude and phase data, but we will be applying the same for real world data in the future. This interdisciplinary study not only opens avenues for improved SAR data analysis but also enriches the burgeoning field of quantum machine learning by highlighting its applicability in remote sensing domains.

1 Introduction

Synthetic Aperture Radar (SAR) has revolutionised the domain of remote sensing, producing high-resolution imagery that remains unaffected by lighting or atmospheric conditions. Its applications span a wide spectrum: from monitoring urban sprawl, assessing flood damages, observing agricultural practices, to tracking deforestation, mapping geological formations, and even conducting planetary exploration[1]. While conventional techniques have made significant strides in interpreting SAR data, the escalating complexity and sheer volume of such data present formidable challenges.

Machine learning, a discipline teeming with algorithms and statistical models that empower systems to autonomously accomplish specific tasks[2], has been instrumental in surmounting several of these challenges. The integration of machine learning into geoscience, climate change research, and environmental monitoring has heralded transformative advances in SAR research[3].

However, as the SAR data deluge shows no signs of abating, classical algorithms reach their performance ceilings. Enter quantum computing—a paradigm that synergizes classical information theory, computer science, and quantum physics[4]. Particularly promising is the emergence of Noisy Intermediate-Scale Quantum (NISQ) technology[5], signalling a significant progression towards more formidable quantum technologies.

This paper embarks on an exploration of tensor quantum machine learning (QML) for SAR data processing. Tensor methods, foundational in modern numerical methods used for simulating many-body physics, have found their niche in machine learning[6]. Quantum machine learning models utilising tensor networks, deemed apt for imminent quantum hardware, have sprouted at the intersection of quantum computing and machine learning[7]. These models, as demonstrated in tensor network quantum machine learning studies[8], can leverage tensor network algorithms to compress classical data representing quantum states.

Harnessing the computational power of quantum mechanics and the data representation proficiencies of tensor networks, this interdisciplinary foray aspires to not only refine SAR data analysis but also enrich the vibrant tapestry of quantum machine learning research, accentuating its ramifications in remote sensing.

2 Methodology

In this study, we embarked on the task of land cover classification using Synthetic Aperture Radar (SAR) data. Thus far, our work has focused on synthetically generated data; however, our future analyses will target real SAR data. Given SAR's complex-valued nature, we meticulously simulated both amplitude and phase components to represent diverse land cover classes such as forests, urban areas, agricultural fields, and water bodies. We then processed the data through various tensor decomposition methods, namely CP Decomposition, Tucker decomposition and Tensor Train/ Matrix Product State decomposition. This was done to tackle the high dimensionality of SAR data. Tensor decompositions allow us to reduce the size as well as work in a lower Hilbert space[9], while at the same time preserving the inherent underlying structures of the data, multi-dimensional interactions, as well as dominant patterns.[6] Convolutional Neural Networks (CNNs) were employed to establish a baseline for classification. We then established a Quantum baseline using a Quantum CNN.

Building upon this, we integrated tensor-based Quantum Machine Learning techniques, aiming to harness the computational potential of quantum mechanics and the representational prowess of tensor networks. We combined manual data generation with sophisticated algorithms to enhance classification results. Our approach contrasts classical methods with quantum ones. In **Fig. 1** we can see a detailed description of the methodology used.

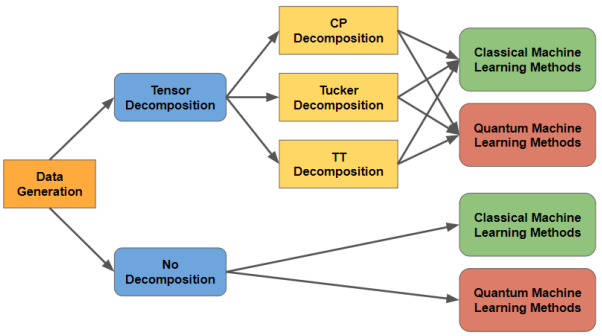

Figure 1 A description of the methodology where we start with generating data, and run different quantum and classical machine learning algorithms on it.

2.1 Data Generation and Characteristics

Synthetic Aperture Radar (SAR) imagery is renowned for its capacity to capture detailed and high-resolution images of the Earth's surface[10]. This research harnesses simulated SAR data, intricately crafted to mirror the characteristics of real-world SAR data. SAR data is inherently complex-valued, composed of both amplitude and phase components. The amplitude, which indicates the strength of the radar response, is particularly sensitive to the texture and structure of the target surface. Conversely, the phase component, dictated by the distance between the sensor and the target, provides a wealth of information about the geometric properties of the surface. To simulate real-world SAR phase and amplitude data, we generated four land cover classes with the following characteristics:

Land Cover Class Simulation:

Forest: Mimicking dense forests, the simulated data for this category manifests high amplitude coupled with speckle noise. The random phase is attributed to multiple scattering events occurring due to the myriad of elements present, such as trees, shrubs, and underlying vegetation.

Urban Areas: Representing built environments like cities or industrial regions, urban areas in SAR data are characterised by very high amplitude. This is due to the strong reflections from buildings and infrastructural elements. The phase varies, capturing the diverse and intricate layout of urban settings.

Agricultural Fields: Simulating vast stretches of farmland or plantations, the agricultural field data showcases medium amplitude. The phase exhibits some variability, influenced by the structure, height, and density of crops.

Water: Bodies of water, such as lakes, rivers, or oceans, are distinguished by low amplitude in SAR data. This is a consequence of the specular reflection where most of the radar signal is reflected away from the sensor. The phase is near-zero, capturing the relatively smooth and uniform nature of water surfaces.

The simulation process started with the creation of empty amplitude and phase canvases of the desired size. These canvases are filled with patterns that mimic real-world land-cover characteristics, such as water, forest, urban, and agricultural regions. Each pattern is mathematically modelled using sinusoidal and cosine functions with specific frequencies and phase offsets. To add a touch of realism and capture the intrinsic speckle noise observed in SAR imagery, we infused random speckle noise into the amplitude data. This noise is generated using Gaussian random variables scaled by a speckle strength factor, which is randomised for each patch to simulate local variability.

Further randomness is introduced by varying frequency multipliers, phase offsets, and amplitude scalings. These introduce variability in the patterns' frequencies and change the phase of the patterns. They also scale the amplitude, ensuring that the simulated data offers a diversified representation akin to real-world observations. The final step involves converting the amplitude and phase information into a complex SAR data representation through the multiplication of amplitude with the exponential of the phase.

For the purpose of variability, we randomly generate multiple images with *size*=64 which defines the size of the image as 64x64. The image is further divided into a 8x8 grid where each element of the grid represents a land-class region.. To add more variability and randomness, each region is further divided into 8x8 pixels with varying phase and amplitude values. A detailed image of the simulated data can be seen in **Fig. 2**. To introduce more data into the models, we generate many such 8x8 grid images, each of the size 64x64.

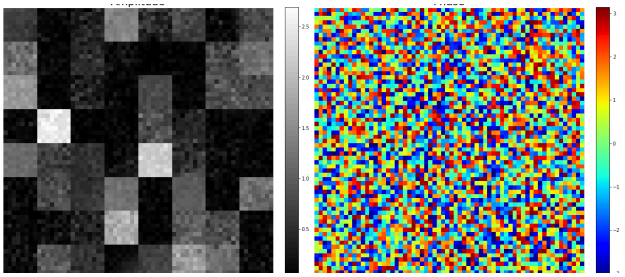

Figure 2 On the left we can see simulated amplitude data and on the right we can see the simulated phase data for the four land-cover classes. The image of size 64x64 pixels is divided into 8x8 smaller land cover regions.

2.2 Classical CNN Baseline

For our baseline model, classical machine learning techniques were employed to process and interpret the simulated SAR data. This began with preprocessing steps tailored to prepare the SAR data for optimal input into a CNN. CNNs were chosen because of their ability to understand spatial relationships in data, as well as their adaptability to complex data[11]. Techniques such as data normalisation, augmentation, and appropriate splitting into training and testing sets ensured the quality and reliability of our dataset. With the data prepared, Convolutional Neural Networks (CNNs) were chosen for their proven proficiency in handling image-based datasets like SAR. These networks were trained on the amplitude and phase components of the data, aiming to classify distinct land cover classes. This classical approach served as our foundation, setting the benchmark against which the quantum models would be assessed.

The architecture was designed to ingest patches of size 8x8. The network begins with a convolutional layer consisting of 8 filters, each of size 3x3, coupled with ReLU activation. To counteract potential overfitting and to introduce regularisation, an L2 regularizer was applied to this layer. Following the convolution, a batch normalisation step was introduced, optimising the activations' scale and accelerating convergence. A max-pooling layer was subsequently employed, effectively halving the spatial dimensions to 4x4, while retaining the most salient features. A dropout layer with a rate of 0.25 was then added to further prevent overfitting.The second stage of the network incorporates a convolutional layer with 32 3×3 filters, again followed by batch normalisation, max-pooling, and dropout. The third convolutional stage features 64 3×3 filters, aiming to capture more intricate spatial patterns, succeeded by batch normalisation, max-pooling, and dropout. Post these convolutional stages, the data is flattened, transitioning the 2D spatial data into a 1D tensor, which is then processed by a dense layer with 32 neurons. Another dropout layer with a rate of 0.5 ensures robustness before the final dense layer, which consists of 4 neurons with softmax activation, corresponding to the four land cover classes. The entire architecture was compiled with the Adam optimizer and categorical cross-entropy as the loss function, ensuring the model's capability to discern and classify the intricate patterns of SAR data effectively.

2.3 Quantum Baseline

For the Quantum baseline, we work with a Quantum Convolutional Network (QCNN) as described in [12] which represents a fusion of quantum computing principles with the foundational architecture of classical convolutional neural networks (CNNs). We also take a heavy influence from the Qiskit QCNN tutorial [14] to design the circuits and the model. At its core, the QCNN leverages quantum circuits to process information in a superposed state, offering potential computational advantages especially for specific problems that are hard for classical computers [12]. The methodology begins with encoding classical data into quantum states, a step achieved via quantum feature maps. These maps transform classical data inputs into corresponding quantum states by applying a series of parameterized quantum gates. Following this encoding, the QCNN introduces quantum convolutional layers, which are constructed using parameterized quantum circuits that mimic the function of classical convolutional filters. Each quantum convolutional filter processes the quantum data, leading to interference and entanglement patterns that capture the data's inherent features. Analogous to pooling layers in classical CNNs, the QCNN can also employ quantum pooling layers to reduce dimensionality and capture the most salient features from preceding layers. The final quantum state is then measured, converting quantum information back into classical form, which can be further processed or directly interpreted as the network's output. Training a QCNN involves tuning the parameters in the quantum circuits to minimise a loss function, much like its classical counterpart. The hybrid nature of QCNNs, combining quantum processing with classical layers or post-processing, offers a promising avenue for harnessing the power of quantum computation in the realm of machine learning, especially as quantum hardware continues to mature.

2.3.1 Data Encoding

Data encoding in quantum computing is a critical and challenging step that involves translating classical data into quantum states that can be processed by quantum algorithms [15]. The process is foundational to performing any form of quantum computation or quantum machine learning, including tasks like classification, clustering, or optimization[16]. One of the primary challenges in data encoding is the efficient utilisation of qubits, the basic units of quantum information. Given the limited number of qubits available on current Noisy Intermediate-Scale Quantum (NISQ) devices, encoding strategies must be carefully designed to maximise information density per qubit while maintaining the fidelity of the encoded data [15]. Techniques such as amplitude encoding offer efficient use of qubits by representing data in the amplitudes of quantum states but require complex state preparation and normalisation of data. Furthermore, the inherently probabilistic nature of quantum measurements introduces challenges in accurately retrieving encoded information. Another significant hurdle is the development of encoding schemes that are resilient to quantum noise and decoherence, which can rapidly degrade the information stored in quantum states.

For our circuit and data, we go with amplitude encoding as our preferred method. Our data set is of the size 64x8x8, where we separate out the real and imaginary parts to make it easier to encode for a quantum computer. This doubles our data to 128 values for each land-cover region. We then flatten the 128 values of each region to a 128-dimensional vector. In amplitude encoding, we encode N values, to $\log_2(N)$ qubits. In amplitude encoding, the data samples are always normalised as the

sum of the squares of all amplitudes must be equal to 1 since they represent probabilities in a quantum state. For measuring our classes we introduce 2 additional ancilla qubits. The combined state of the 2 qubits 00, 01, 10 and 11 represent the 4 different classes of our problem. We add additional coherence of the ancilla qubits to the circuit, and only the ancilla qubits are measured at the end.

2.3.1 Circuit Design

For the efficient running of our quantum model, we design a circuit based on the classical version of a CNN. We use unitary gates to define the primary functions of CNN, namely the convolutional layer and the pooling layer. We then stack multiple instances of these operations to effectively represent a Quantum Convolutional Neural Network (QCNN). As we see in [14], to effectively represent different layers of a convolutional network, we use multiple types of fundamental quantum gates to represent various operations. For the convolutional layers we use primarily Cy and Cz gates in conjunction with CNOT gates to induce coherence in the circuit. Similarly for the pooling layer we use Ry and Rz gates as well along with CNOTs for coherence. We arrange the circuit ansatz along our 7 qubits with amplitude encoding, and add CNOT gates with the ancillary measurement qubits to infer the result of our classification.

The circuit design relies heavily on [14] for inspiration, however, changes have been made to accommodate for our specific problem. Much like their classical counterparts, QCNNs require experiments and testing to come up with the correct ansatz for our requirement. The primary problem in our case being the limited ability to compute and train QCNNs with the restrictions that come along with it. We tried to optimise our ansatz for a balance between computation needed, and the requirements of the task at hand to effectively classify SAR data. Since complex data in general is a challenge for classical, and especially quantum machine learning, we for the moment only work with the real parts of our data, represented efficiently for our networks.

2.4 Decomposition Methods

Tensor decompositions are fundamental techniques in the analysis of high-dimensional data, providing a framework to break down tensors, which are multi-dimensional arrays extending beyond matrices, into simpler, lower-dimensional components [17]. These methods, rooted in multilinear algebra, allow for the efficient representation and analysis of complex data structures. Among the key approaches are the Canonical Polyadic (CP) decomposition [18], which represents a tensor as a sum of rank-one tensors; the Tucker decomposition [19], a more general method that decomposes a tensor into a core tensor and a set of matrices corresponding to each mode; and the Tensor Train (TT) decomposition [20], designed for handling very high-dimensional tensors through a series of low-dimensional tensors in a sequential arrangement. By reducing the dimensions and complexity of the data, tensor decompositions play a crucial role in various applications, including signal processing, machine learning, and quantum physics, by enabling the extraction of meaningful patterns and reducing computational and storage requirements.

CP Decomposition: The Canonical Polyadic (CP) decomposition, also referred to as PARAFAC (Parallel Factors) decomposition, is a method for decomposing a tensor into a finite sum of rank-one tensors. This approach simplifies the representation of multi-dimensional arrays by expressing them as combinations of the most basic tensor units, facilitating both the analysis and interpretation of complex data. In the CP decomposition framework, each component tensor is defined by the outer product of vectors, corresponding to each dimension of the original tensor, thereby maintaining the data's multi-linear properties [18]. The utility of CP decomposition lies in its ability to reveal latent structures within the data, making it suitable for uncovering patterns across various dimensions.

Tucker Decomposition: The Tucker decomposition is a higher-order factorization method used to decompose a tensor into a core tensor and a set of orthogonal matrices. This method extends the concept of matrix singular value decomposition (SVD) to tensors, providing a more flexible and general approach to tensor decomposition. The core tensor in Tucker decomposition captures the interactions between different modes of the original tensor, while the matrices, associated with each mode, represent the principal components or factors along that dimension [19]. This decomposition not only reduces the dimensionality of the data but also retains its intrinsic structure, making it easier to analyse and interpret complex multi-dimensional datasets. The Tucker decomposition is particularly useful for data compression, noise reduction, and feature extraction.

Tensor Train Decomposition: The Tensor Train (TT) decomposition, also known as the Matrix Product State (MPS), is a decomposition technique that represents a high-dimensional tensor as a sequence of lower-dimensional tensors, specifically 3-dimensional tensors (or matrices for the first and last tensors in the sequence) [20]. This method provides an efficient way to handle and compute with tensors that have a large number of dimensions, which would otherwise be intractable due to the exponential growth of parameters with the tensor order. In TT decomposition, each element of the original tensor can be computed as a product of matrices, where each matrix is associated with a particular mode (dimension) of the tensor [20]. This sequential arrangement allows for a compact representation of the tensor, significantly reducing the amount of storage required and facilitating operations such as tensor addition, multiplication, and contraction.

3 Results

From our initial experiments focused on establishing a baseline, we see that classical CNNs are relatively good with our generated amplitude and phase data for land-cover classification. The accuracy results can be seen in **Fig. 3.** Even though our data had high variability, the architecture of our CNN adapts well for the problem for 20 epochs.

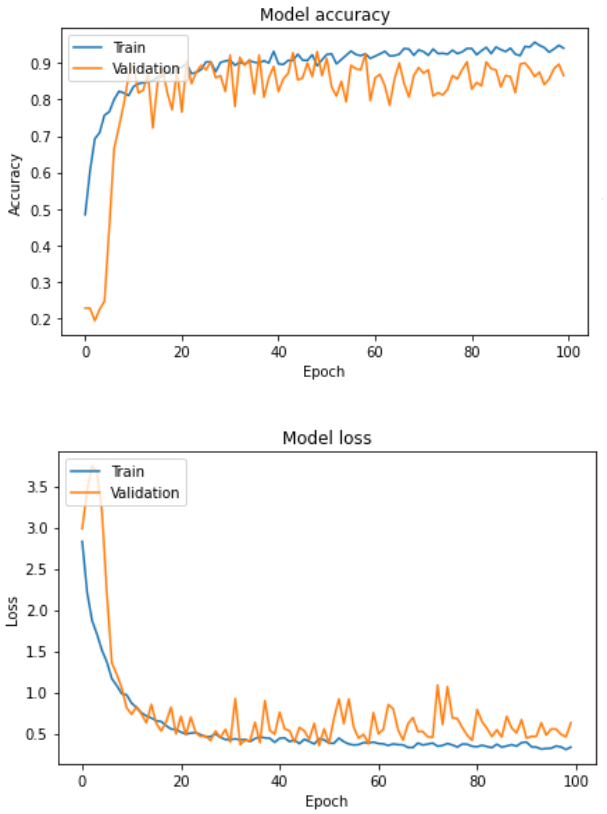

Figure 3 Although there is a variability in the validation accuracy and loss, in general it achieves a good result. The average accuracy is around 86% for 20 epochs.

For our Tensor-based decomposition, we used a Matrix Product State/Tensor-Train decomposition method on the same data which trained our CNN. We decomposed and reconstructed the data, and then fed it into the same CNN architecture. This resulted in similar accuracy and loss values while using less memory and working in lower dimensions. In **Fig 4.** we see how the decomposed and then reconstructed image tensors perform with the CNN.

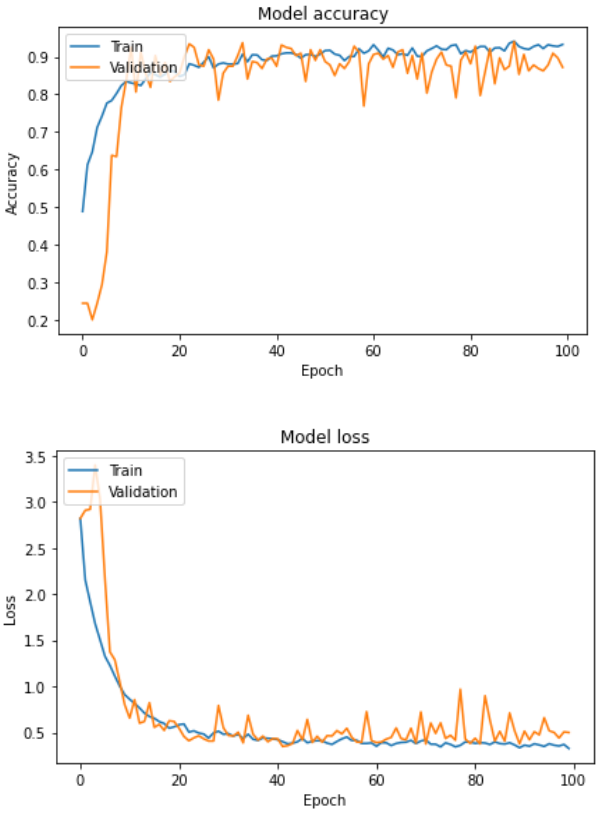

Figure 4 The Tensor Train decomposition for our data, does not alter the performance of the CNN so much. The average accuracy on validation data is 82% for 20 epochs.

We are currently waiting for our quantum circuit to run and simulate both decomposed and non-decomposed data and the paper will be updated as soon as we get the results. The process of simulation in a complex scenario such as ours takes a bit of time.

4 Future Work

The plan for the future involves working with multiple decomposition formats to begin with, such as CP and Tucker decompositions. We would then have a comparison of classical machine learning algorithms, such as CNNs, and look at their performance on non-decomposed and decomposed data. Then we would work on finalising a QCNN architecture for our problem. Currently we have a QCNN running on our non-decomposed data, after the evaluation of its performance, we would evaluate it on decomposed data. For the final step of establishing a framework, a Tensor-based quantum circuit architecture would be used to evaluate the two kinds of SAR data, as defined in [13]. After evaluation, we would modify the architecture to better suit SAR land-cover classification requirements.

For the framework to be useful for real-world data, we would aim to test our methods on real SAR data. We hope to see a similar performance on real-world data of our framework due to the complexity and variability of our data generation process, however, the challenge would be

performing classifications of a quantum nature due to the size of the data itself. This is where we hope Tensor-based methods might come in handy, to help us tackle the complexity of our dataset.

Acknowledgement

This project was made possible by the DLR Quantum Computing Initiative and the German Federal Ministry for Economic Affairs and Climate Action, https://qci.dlr.de/qua-sar/.

5 Literature

[1] A. Moreira, P. Prats-Iraola, M. Younis, G. Krieger, I. Hajnsek and K. P. Papathanassiou: A tutorial on synthetic aperture radar, IEEE Geoscience and Remote Sensing Magazine, vol. 1, no. 1, pp. 6-43, March 2013

[2] Y. Lecun, L. Bottou, Y. Bengio and P. Haffner: Gradient-based learning applied to document recognition, Proceedings of the IEEE, vol. 86, no. 11, pp. 2278-2324, Nov. 1998

[3] Mauro M. Barbat, Christine Wesche, Adriano V. Werhli, Mauricio M. Mata: An adaptive machine learning approach to improve automatic iceberg detection from SAR images. ISPRS Journal of Photogrammetry and Remote Sensing, Volume 156, 2019.

[4] A. W. Harrow and A. Montanaro: Quantum computational supremacy, Nature 549, 203-209, 2017

[5] Preskill, John.: Quantum Computing in the NISQ era and beyond. Verein zur Forderung des Open Access Publizierens in den Quantenwissenschaften, 2018

[6] Huggins, Williams: Towards quantum machine learning with tensor networks. Quantum Science and Technology 2019.

[7] A. Montanaro: Quantum algorithms: an overview, Quantum Information, 15023, 2016

[8] Taylor L. Patti, Kossaifi J., Susanne F. Yelin and Anima Anandkumar: TensorLy-Quantum: Quantum Machine Learning with Tensor Methods. Arxiv, 2021

[9] Kolda, Tamara G., and Brett W.: Tensor decompositions and applications. SIAM review 51.3, 2009

[10] Lapini, Alessandro, et al.: Comparison of machine learning methods applied to SAR images for forest classification in mediterranean areas. Remote Sensing 12.3, 2020

[11] C. Szegedy et al: , Going deeper with convolutions. 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 2015

[12] Cong, Iris, Soonwon Choi, and Mikhail D. Lukin: Quantum convolutional neural networks. Nature Physics 15.12, 2019.

[13] Guala, D., Esther, C., Zhang, S., and Arrazola, J.: Tensor-network quantum circuits. Pennylane Demos, 2022.

[14] IBM: The Quantum Convolutional Neural Network, 2017

[15] Plesch, Martin and Brukner, Caslav: Quantum-state preparation with universal gate decompositions, Phys. Rev. A 83, 2011

[16] Manuela Weigold, Johanna Barzen, Frank Leymann, and Marie Salm: Data encoding patterns for quantum computing. In Proceedings of the 27th Conference on Pattern Languages of Programs (PLoP '20). The Hillside Group, USA, Article 2, 1–11, 2022.

[17] Stephan Rabanser and Oleksandr Shchur and Stephan Günnemann: Introduction to Tensor Decompositions and their Applications in Machine Learning, 2017.

[18] F. L. Hitchcock: The expression of a tensor or a polyadic as a sum of products. Journal of Mathematics and Physics. 6 (1–4): 164–189, 1927

[19] Ledyard R. Tucker: Some mathematical notes on three-mode factor analysis. Psychometrika. 31 (3): 279–311, 1966.

[20] Perez-Garcia, D.; Verstraete, F.; Wolf, M.M:. Matrix product state representations, Quantum Inf. Comput. 7: 401, 2008.