Quantum Machine Intelligence (2025) 7:22
https://doi.org/10.1007/s42484-025-00250-y

RESEARCH ARTICLE

A randomized transformation for data processing on quantum computers

Johann Maximilian Zollner[1] · **Martin Werner**[1]

Received: 27 March 2024 / Accepted: 31 January 2025

Abstract
Quantum state preparation circuits for loading classical data into quantum computers significantly influence the performance and complexity of variational quantum algorithms within hybrid quantum-classical systems. Since real-world data is often high-dimensional and mapping it to a quantum state is costly and non-trivial, it is transformed on classical computers to overcome limitations regarding quantum register size and circuit complexity. We present a data transformation method using Bloom filters to represent classical data for quantum state preparation circuits. Further, the paper illustrates that tiny fragments of this classical data encoding can be used for quantum machine learning to make an ensemble of the resulting quantum models surprisingly powerful. Due to the representation of the transformed data as a fragmented bit array, the quantum state preparation circuit only relies on a single rotational gate per qubit and small quantum registers. We demonstrate our approach and its representation power in a series of simulations. The simulations indicate that the randomized transformation provides diversity for ensemble models and that even small bit arrays with high error rates in data representation are sufficient for binary classification tasks.

Keywords Data transformation · Data structure · Data encoding · Bloom filter · Quantum state preparation · Quantum machine learning · Ensemble learning

1 Introduction

Recent work shows that current and near-term quantum computers will likely not show a practical advantage over classical computers for big data problems due to input and output bandwidth limitations (Hoefler et al. 2023). There, it is concluded that instead, computational problems of large complexity formulated using small input data are more likely to gain from this technology. The reason is that, especially in the "noisy intermediate-scale quantum" era, quantum computers are limited in the number of quantum bits and the depth of a series of operations. Hardware noise and faults during computation demand small registers of qubits and prevent circuits with high depth from being used. The impact of quantum noise further depends on the circuit complexity and architecture and strongly affects the quality of results of quantum programs (Pan et al. 2023).

Nevertheless, the so-called hybrid quantum-classical algorithms provide a promising class of algorithms for current and near-term hardware (McClean et al. 2016; Benedetti et al. 2019). For example, variational or parameterized quantum circuits can be inserted into otherwise classical programs. In this setting, a classical program delegates part of the computation to a quantum kernel, which takes classical data as input and provides classical data back to the classical program through measurement. The quantum kernel is often called a variadic quantum circuit, as it is also typically parametrized with classical data. In quantum machine learning, for example, a classical optimization algorithm aims to find good parameters for a variadic circuit to solve a machine learning problem. In this context, particular parameterized quantum circuits for machine learning are a well-studied field, and especially, quantum classifiers have received substantial attention in recent years (Lu and Braunstein 2014; Farhi and Neven 2018; Havlíček et al. 2019; Adhikary et al. 2020; Li et al. 2022b). Specifically, an efficient design of

✉ Johann Maximilian Zollner
maximilian.zollner@tum.de

Martin Werner
martin.werner@tum.de

[1] Department of Aerospace and Geodesy, Technical University of Munich, Lise-Meitner-Straße 9, Ottobrunn 85521, Bavaria, Germany

Published online: 17 February 2025

the circuit architecture in terms of structure and layout of quantum gates is crucial to maximize the use of quantum hardware (Sim et al. 2019). Consequently, an efficient and compact circuit design with low depth and parameter count is crucial when utilizing current quantum hardware for real-world tasks.

A unique challenge in quantum computing is finding efficient methods to bring classical data into a quantum state on which quantum circuits can operate. Therefore, implementing a state preparation circuit is required, which is a non-trivial task. An inherent problem in this context is that quite a few short-state preparation circuits are known, but typically, limited in the amount of classical information, they can encode. For example, angle encoding can encode N classical bits on N qubits where $N \in \mathbb{N}^+$ is a positive integer. Quantum state preparation significantly influences the performance of quantum algorithms, and considerable efforts were made to improve quantum state preparation (Gil Vidal and Theis 2020; Pérez-Salinas et al. 2020; Caro et al. 2021; Schuld 2021). More specifically, in the case of quantum classifiers, the chosen quantum encoding method determines the robustness against noise and the learnable decision boundaries of the classifier (LaRose and Coyle 2020). Also, with increasing width and depth of the state preparation circuit, the capabilities of quantum classifiers get limited with respect to the distinguishability of encoded states (Li et al. 2022a). However, the most direct impact on the performance of a variational circuit is that the depth of the state preparation and the circuit add up, severely limiting the use of quantum algorithms for real-world data.

In this context, we introduce an approach to represent classical data in a few bits such that the range of classical problems on which these circuits can be applied is increased. One of the key ideas is that beyond a low-dimensional representation of data, a lossy representation can be applied in which a quantum classifier is provided only with a limited amount of information. We show how multiple classifiers with non-overlapping information can be combined into a powerful overall classifier. This allows us to solve classification problems by distributing the required input information across multiple tiny quantum circuits. In this paper, we mainly consider low-cardinality data of high dimensionality. This data type can be quantized in all dimensions into comparably few classes but can have exceptionally high dimensionality. For example, binary images can be understood by humans and machine learning with as few as 1 bit per pixel. Still, many pixels are typically required as the actual information is distributed spatially across the image.

The scientific contribution of this paper is threefold. We introduce a data representation framework to encode classical, high-dimensional, low-cardinality data, including, but not limited to, binary images into bit sequences. Furthermore, we present a technique to split the bit sequence into multiple fragments that cover the information content of the longer sequence. Finally, we validate the previous concepts with classical and quantum machine learning in various numerical simulations and evaluate the performance of the approach. Note that, for completeness, we also present and investigate an extension that allows floating point data to be represented. However, this comes at a comparably high representation cost in the proposed framework and will, in most cases, not outperform direct floating point encoding mechanisms such as angle encoding.

The remainder of this work is structured as follows. In Sect. 2, we discuss the fundamental concepts connected to this work, which are the probabilistic data structures Bloom filter and GloBiMaps, the classical and quantum parts of hybrid quantum-classical systems, and ensemble learning. Next, we present our pseudo-randomized data encoding for quantum state preparation and a qualitative visualization and discussion of the impact of randomization and lossy compression and its connection to ensemble learning in Sect. 3. Following, we show the first results and the surprising representation power of the randomized encoding with supervised machine learning in Sect. 4. We present and discuss the numerical simulations regarding setup, findings, and results and answer the questions: "Can we sufficiently represent real-world data with fragments of randomized few-bit representations for classification?", "How does the false positive rate of a Bloom filter impact the model's performance?", and "Does the fragmentation of a filter provide sufficient diversity for an ensemble model?". Finally, we summarize and conclude the work in Sect. 5.

2 Fundamental concepts and related work

This section discusses the fundamental concepts and related work connected to this paper. First, we will present the probabilistic data structures Bloom filter (Bloom 1970) and Global Binary Maps (Werner 2019a), which are core concepts of this work. Then, we will look into hybrid quantum-classical systems for machine learning and define parameterized quantum circuits and related concepts. Finally, we will define and show how ensembles of weak base classifiers can be used to enhance quantum machine learning.

2.1 Bloom filter

The Bloom filter is a space-efficient probabilistic data structure proposed by Burton Howard Bloom in 1970 (Bloom 1970). The compact structure represents a set as a bit array and enables efficient set membership queries. While a Bloom filter guarantees no false negatives, it may contain false positives, which means a tested element is either not in the set or maybe in the set with a certain probability. By accepting that

a portion of bits of the filter are falsely identified as part of the set, the structure makes it possible to represent data with a significantly shorter bit array. Thus, the data structure provides the freedom to trade the accuracy of correctly identifying the membership of an element with the filter size. This option for lossy compression is an advantage of the Bloom filter over similar data structures since we can use as little memory as required while maintaining short computing time.

Bloom filters have been a constant subject of research over the last decades, with many variants presented in literature (Kirsch and Mitzenmacher 2006; Christensen et al. 2010; Debnath et al. 2011; Chikhi and Rizk 2013; Luo et al. 2018). While Bloom originally developed the data structure for spell-checking and word separation, today, there are many applications (Abdennebi and Kaya 2021). Nevertheless, the data structure is most popular in database management. For example, Apache Cassandra and PostgreSQL use Bloom filters to reduce disk lookups for non-existent rows and columns. However, the data structure is useful for a wide variety of tasks and applied in network routing, traffic measurement, and processing (Kumar et al. 2003; Broder and Mitzenmacher 2004; Song et al. 2005; Nayak et al. 2021) and network security (Geravand and Ahmadi 2013; Patgiri et al. 2018) and bioinformatics (Melsted and Pritchard 2011; Jackman et al. 2017).

We will now define the elementary mathematical concepts of Bloom filters. Let B denote a Bloom filter, a bit array of size m, to represent a set $x \in \mathcal{X}$ with n elements. An empty Bloom filter represents an empty set with all elements set to zero. Now, we chose k pairwise independent hash functions h_i, which map each element of the set x to k bits b_j in B where $0 < k \ll m$. These hash functions are non-cryptographic and more efficient regarding computational time and resources than cryptographic functions. Also, it must be considered that while the mapping is theoretically random, it is practically deterministic since a dependence exists due to the nature of randomization on computational hardware, which is always bound to some seed. If an element of x is inserted into B, the corresponding k bits b_j are set to one. The mapping is visualized in Fig. 1. Consequently, the fraction of zeros (foz) describes the percentage of zeros of a filter:

$$\text{foz} = \left(1 - \frac{1}{m}\right)^{kn} \approx \exp\frac{-kn}{m} \tag{1}$$

and depends on the filter size m, the number of hash functions k, and the number of inserted elements n. Here, foz $= 0.5$ is the desirable value for filters created from $\mathcal{X}$ since it provides the highest entropy for the bit string. Note that Eq. 1 is not statistically correct but gives a good intuition and holds for reasonable large m. The effect of small bit arrays on the fraction of zeros has been discussed in detail in previous literature, such as in Mitzenmacher (2001).

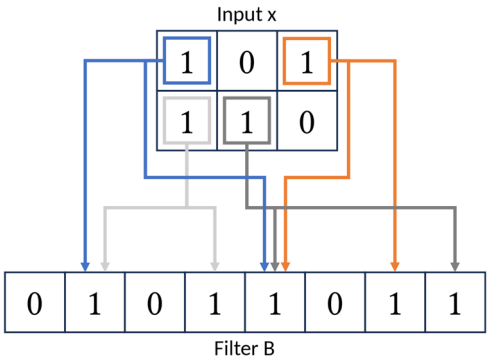


Fig. 1 An example of the creation of a Bloom filter B with $m = 8$ and $k = 2$, representing raster data x with 2×3 binary values: $h_i(x) = b_j$ where $i = 1, ..., k$ and $j = 1, ..., m$

As previously mentioned, the data structure provides a trade-off between the size m and the error probability. This error probability only refers to one type of error: the false positive (FP). The FP rate p of a Bloom filter configuration is given by the following:

$$p = \left(1 - \exp\frac{-kn}{m}\right)^{k} \tag{2}$$

and describes that an FP happens if k random tests of an element yield a one. Although the actual FP rate may be higher in real applications (Gremillion 1982; Mullin 1983), it has been theoretically proven that Eq. 2 gives a lower bound on the FP rate (Bose et al. 2008).

Further, we can find the minimum number of hash functions k for an expected FP rate p by differentiating Expression 2 and finding the roots. Given a memory size m and the number of elements to be inserted n, we can obtain the global minimum of the FP rate with the following:

$$k_* = \frac{m}{n} \ln 2. \tag{3}$$

In this paper, k is constrained to be an integer number, so we must round k_*.

Furthermore, the optimal number of bits b_j, which is the memory budget m_*, can be calculated for a target FP rate p in a straight forward way:

$$m_* = \frac{-n \ln p}{(\ln 2)^2}. \tag{4}$$

Since the filter length m strongly affects the FP rate p, this leads to the aforementioned trade-off between size and accuracy.

2.2 Global binary maps

Based on the Bloom filter, we now discuss Global Binary Maps (GloBiMaps), which was first presented in Werner (2019a) and extended in Werner (2021). GloBiMaps efficiently stores a binary raster into a global bit map and offers possibilities for augmentations. The probabilistic data structure was initially made to avoid costly disk lookups when working with sparse, high-resolution, low-cardinality raster data. A Bloom filter is the core of the data structure, where pixels are inserted by hashing with the Murmur3 hash function (Appleby 2008, 2016). Murmur3 is a general-purpose, non-cryptographic hash function known for quality distribution while maintaining high performance. Further, GloBiMaps utilizes an advanced hashing method based on Kirsch and Mitzenmacher (2006) and linear congruential generators that allow the creation of multiple hash functions with a single computation of Murmur3. Therefore, the 128 bit hash returned by the Murmur3 function is split into h_{low} and h_{high} with 64 bit each. Then, k independent and identically distributed hash functions h_i for a Bloom filter are created with a single computation of Murmur3 by the following:

$$h_i(x) = h_{low}(x) + i h_{high}(x) \mod 2^m \tag{5}$$

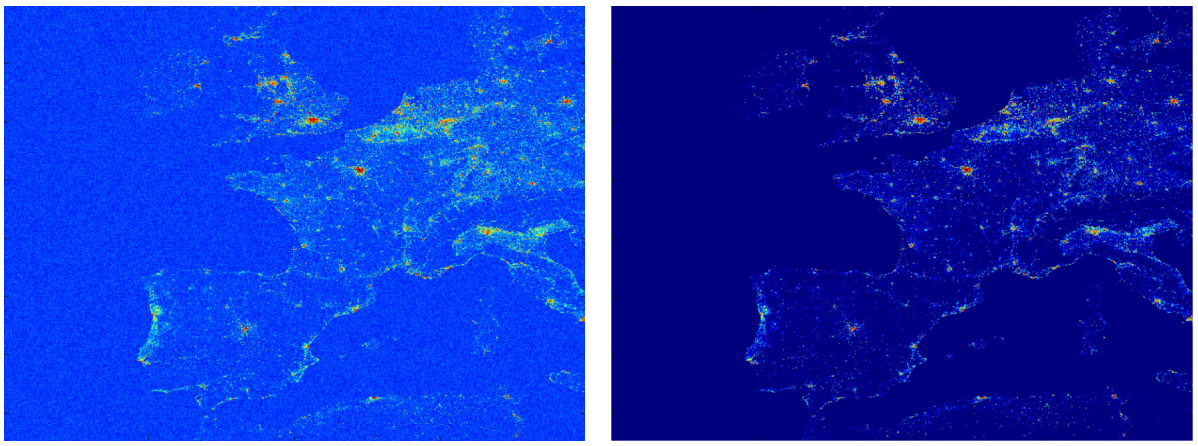
where $i \in \{1...k\}$. However, the process is only pseudo-random, which means the generated numbers are interdependent.

Although the modulo operation enables a computational advantage, this also limits the data structure in this work to the power of two sizes for m for Bloom filters. While this may seem restrictive at first, the power of two filter sizes allows the application of the following compression scheme. By halving the filters and combining them with a binary OR operation, the size of B is reduced to $m/2$, creating a pyramid-like filter structure. The resulting bit array is equivalent to the one that would have been created with the same set of hash functions. However, we use a value of $m/2$ instead since the modulo operation maps the integer hash values into the index range of the filter. Another advantage is that the computation of $\mod 2^m$ can be implemented without an actual division since it is equivalent to the computation of a binary AND with $2^m - 1$. Thus, this implementation can significantly increase the computational performance of the hashing procedure and ensure uniform bit rates for the buckets of information.

Since only small-scale filters are used in this paper, an example of the scalability of GloBiMaps and Bloom filters is given here. In previous work Werner (2021), a low-resolution version of the Global Urban Footprint (Esch et al. 2017) with a size of $192,857 \times 462,859$ for a total of about $89,256$ gigapixels is used to demonstrate how GloBiMaps compresses image information. Figure 2 shows two renderings of Europe created by averaging 20×20 input pixels into a single pixel value. It shows that a modulo-2 compression with 8 megabyte, although noisy due to FPs, clearly represents the structure of European urban regions. For compressions with 64 megabytes, errors are already rare and impossible for a human to notice. GloBiMaps may also be augmented with error correction tables modeling FPs, which could be considered for future work. However, this is not further discussed in this paper, and instead, we refer to Werner (2021) for more detailed information.

2.3 Hybrid quantum-classical systems for machine learning

Within hybrid quantum-classical systems, quantum computers can be used for small-scale machine learning tasks with real-world data. By running subroutines on classical



(a) $m = 8$ megabytes (b) $m = 64$ megabytes

Fig. 2 Example of GloBiMaps application from Werner (2019b): Urban regions over Europe by color-coding 20×20 pixel patches. **a** The 8 megabyte representation already has clear structures but is noisy due to the FPs. **b** With a 64 megabyte representation, FPs are already rare

hardware, modern quantum computers utilize parameterized quantum circuits as a central part of machine learning models (Benedetti et al. 2019). Real-world data is transformed on classical computers, which usually is a compression before being processed by the parameterized quantum circuit. Finally, the retrieved output is classically post-processed. Figure 3 shows a generalized hybrid system for supervised classification and outlines the interactions between classical and quantum parts. Note that the output is a single value when simulating quantum circuits on classical hardware, while it can be an expectation value on real quantum hardware.

In this paper, a parameterized quantum circuit $\hat{U}_\theta(x)$ is built out of an encoding circuit $\hat{E}_x$, which is parameterized by the input x, and a variational circuit $\hat{P}_\theta$, which is parameterized by a set of parameters θ, in form of

$$\hat{U}_\theta(x) = \hat{P}_\theta \hat{E}_x. \tag{6}$$

A circuit is built from a set of unitary transformations $\{U_t(\theta_t)\}$, which may also be fixed instead of parameterized, in the form of $\hat{U}_\theta = \hat{U}\left(\{U_t(\theta_t)\}\right) = U_T(\theta_T)U_{T-1}(\theta_{T-1})...U_1(\theta_1)$. The number of sequential transformations determines the depth of the circuit. Now, the parameterized quantum circuit acts on the initial quantum state $\hat{U}_\theta(x)|0\rangle = |\psi(x,\theta)\rangle$ to produce a new state that can be measured to obtain an output. The initial quantum state depends on the quantum computer but is, in this work, assumed to be $|0\rangle$. An arbitrary quantum circuit has a given width N, equivalent to the number of involved qubits, which indicates the size of the qubit register.

Before mapping classical information to a quantum state with a state preparation circuit, it is processed on classical hardware and transformed to fit in the limited input domain of the parameterized quantum circuit. Such transformation is the main subject of this paper and is denoted in this remaining section as ϕ. The concept of a randomized transformation will be discussed in detail in Sect. 3.

The state preparation circuit is parameterized by the transformed information in the form of $\hat{E}_{\phi(x)}$, which is the quantum encoding. There is a large variety of encoding methods that have been studied extensively in previous work (LaRose and Coyle 2020; Weigold et al. 2020, 2021a, b; Schuld 2021; Ashhab 2022). However, an efficient quantum encoding in terms of depth is, for example, basis encoding, which only relies on state preparation circuits with depth $\mathcal{O}(1)$. Assuming the initial quantum state is $|0\rangle$, for basis encoding, a Pauli-X gate acts on the quantum state when the input bit is 1, resulting in the new quantum state $|1\rangle$, in the form of

$$X|0\rangle = |1\rangle. \tag{7}$$

If the input bit is 0, no Pauli-X gate is applied, and the initial quantum state remains $|0\rangle$.

An efficient quantum encoding regarding the number of qubits is, for example, amplitude encoding that requires at least $log_2 N$ qubits to encode a N-dimensional data point, while basis and angle encoding need N qubits for the same data point. However, the cost for amplitude encoding shifts from the width to the depth of the state preparation circuit. While most known encoding algorithms generally require quantum circuits with a depth of $\mathcal{O}(N)$ to load N-dimensional vector data, recent work also showed that it is possible to encode a N-dimensional vector using a circuit with polylogarithmic depth and entangled information in ancillary qubits (Araujo et al. 2021). In the case of imagery, that is high-dimensional data, a well-known encoding method similar to amplitude encoding is the so-called flexible representation of quantum images where N pixels are encoded using $log_2 N + 1$ qubits (Le et al. 2011). The

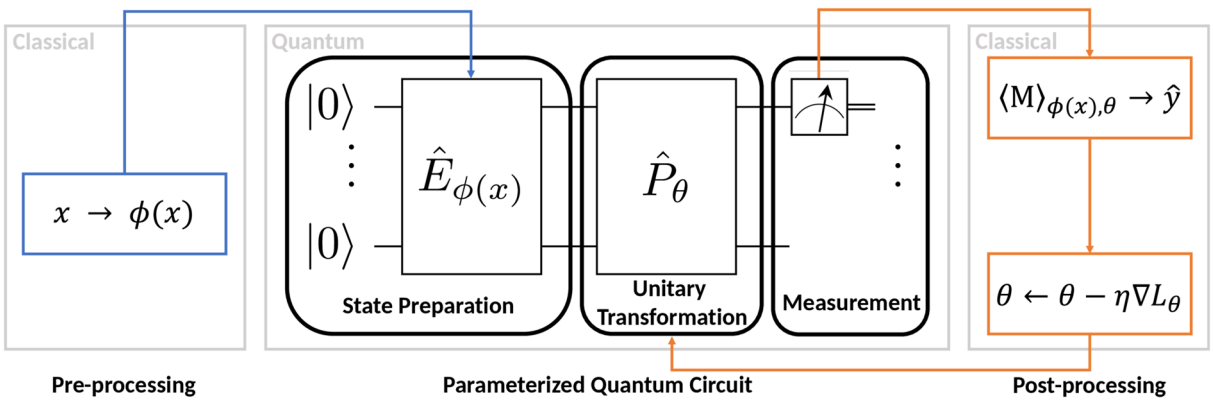


Fig. 3 A hybrid quantum-classical system for supervised classification. Data is transformed on classical hardware before being processed by a parameterized quantum circuit. The parameterized quantum circuit consists of a state preparation circuit, a unitary transformation that implements the classification algorithm, and a measurement. Finally, the output is mapped to the prediction, and parameters are updated on classical hardware

flexible representation of quantum images requires a state preparation circuit depth polynomial in N, thus also shifting the hardware requirements from the width to the depth of the quantum circuit. Note that previous work addressed the depth requirement of the flexible representation of quantum images by using an approximate compressed representation based on matrix product states and reduced the depth to $\mathcal{O}(poly(\chi)log_2 N)$, where χ denotes the bond-dimension of the matrix product states (Dilip et al. 2022).

Nevertheless, encoding methods like basis and angle encoding are still the most efficient regarding the number of operations and time with a depth of $\mathcal{O}(1)$ for the state preparation circuit since they only require single-qubit rotations. However, the disadvantage of such direct encoding methods is that they require N qubits for N input vector components. Consequently, regarding the choice for state preparation circuits, one must trade qubit register size with circuit depth while maintaining a low enough complexity to be robust against quantum noise.

Following the state preparation circuit, the unitary transformation $\hat{P}_\theta$ is the variational circuit that implements the algorithm on the quantum computer and processes the quantum encoded data. Variational circuits $\hat{P}_\theta$ have shown immense potential and are, besides quantum state preparation, one of the most studied fields when it comes to quantum machine learning (Lu and Braunstein 2014; Farhi and Neven 2018; Havlíček et al. 2019; Adhikary et al. 2020; Li et al. 2022b). The measured quantum state is denoted by $\langle M_{x,\theta}\rangle$ and depends on the input x and some set of parameters θ. Note that, in this work, the measurement will always refer to a measurement in the computational basis. Following, with $\hat{U}_\theta(x) = \hat{P}_\theta \hat{E}_x$ acting on the initial quantum state $|0\rangle$, we get the following:

$$\langle 0|\hat{U}_\theta^\dagger(x) M \hat{U}_\theta(x)|0\rangle = \langle M_{x,\theta}\rangle \tag{8}$$

where the model's output $\langle M_{x,\theta}\rangle$ is the measurement of one or multiple qubits of the quantum circuit, which is mapped to the model's prediction $\hat{y}$. By denoting the unitary transformation of the quantum state as $\hat{U}_\theta(\phi(x))|0\rangle = |\psi(\phi(x),\theta)\rangle$, we get the following:

$$f_\theta\big(\phi(x)\big) = \langle \psi\big(\phi(x),\theta\big)|M|\psi\big(\phi(x),\theta\big)\rangle \tag{9}$$

which is the quantum machine learning model. Finally, the model parameters θ are updated. The parameter update is done by computing a loss function L_θ and an optimizer of choice. Gradients with respect to the circuit parameters are estimated following the parameter-shift rule (Mitarai et al. 2018; Schuld et al. 2019).

2.4 Ensemble learning

Ensemble learning is a powerful way to enhance machine learning models, combining predictions from multiple base models. With a collective decision according to a suitable combination rule, the ensemble can perform significantly better than the base models. As each base model is trained on a subset of the data, they acquire distinct characteristics, making the ensemble more diverse and less error-prone. Diversity plays a crucial role in improving the generalization performance of an ensemble model. Each base model needs to produce unique predictions, as models with differing predictions are more likely to make dissimilar errors. Consequently, the ensemble model's overall error is reduced through the inclusion of diverse base models.

Ensemble learning is a well-studied field, and previous publications showed how it improves generalization and helps to avoid overfitting and getting stuck in local optima (Dietterich 2000; Zhang and Ma 2012; Sagi and Rokach 2018). Collective decision-making with ensembles has also proven to be a powerful tool for improving quantum machine learning approaches, and recently, extensive efforts have been made to enhance quantum ensemble methods (Schuld and Petruccione 2018; Macaluso et al. 2020; Niu and Ma 2023). Furthermore, quantum ensemble learning may enable the usage of fewer qubits, which is particularly interesting for the small-scale and near-term intermediate-scale quantum era (Incudini et al. 2023). Using less deep and complex models reduces the influence of noise in quantum hardware so that ensembles of weak models achieve superior performance compared to larger and more complex models. Furthermore, ensemble learning allows for more extensive dimensionality reduction and lossy compression since the base classifiers can have defects for certain instances and comparable low performance. Thus, smaller qubit register sizes can be realized, reducing the influence of quantum noise. Also, using fewer qubits mitigates barren plateau effects, one of the major challenges for quantum machine learning (McClean et al. 2018; Cerezo et al. 2021). Note that although ensemble learning applies to various tasks, in the following work, we will specifically refer to classification for simplicity.

Bootstrap aggregating, or bagging, is an ensemble learning method particularly well suited for low-depth quantum circuits (Incudini et al. 2023). A bagged model which takes the mean over base models f_c with $c = 1, ..., C$ can be denoted as follows:

$$f_{bag}(x) = \frac{1}{C}\sum_{1}^{C} f_c(x). \tag{10}$$

Although averaging is the straightforward method to combine the model outputs, previous work indicates that a

majority vote is superior for classical classifiers (Murphy 2022). For the majority vote, the class with the absolute majority in the set of predicted classes, which means the subset of predictions for that class is more than half the total number of predictions, determines the final prediction. Nevertheless, the final prediction $\hat{y}$ for the true label y is a collective decision of the base classifiers' individual predictions $\hat{y}_c$. Generally, bagging reduces variance and, thus, contradictory to the related ensemble learning method boosting, helps to avoid overfitting.

While an ensemble will have an identical bias to the base classifier, it typically improves performance. The effect of the collective decision on the performance of a classifier can be easily shown by calculating the probability for an ensemble to pick a class A in a binary classification task. Let $y_c = \{A, B\}$ be the prediction of a base model f_c with a known accuracy a. The probability that the ensemble model picks class A is as follows:

$$p = \sum_{c}^{|\frac{C}{2}|} \frac{C!}{c!(C-c)!} a^c (1-a)^{C-c} \tag{11}$$

which is the cumulative distribution function of the binomial distribution with parameters C, p_c evaluated at $C/2$. For instance, with $a = 0.55$, this results in $p = 0.74$ for $C = 10$, $p = 0.86$ for $C = 100$, and $p = 0.99$ for $C = 1000$, which illustrates the power of an ensemble of weak base classifiers. In the context of Eq. 11, we assume that individual classifiers, labeled as f_c, are diverse and generate independent errors, even though practical scenarios may involve correlations.

3 Randomized data structure for quantum state preparation

Following, we will introduce the concept of a randomized data structure for quantum state preparation in detail. We present how the preliminary discussed probabilistic data structure and hybrid quantum-classical systems connect. Next, we present a qualitative visualization of the impact of the FP rate on the representation of imagery. Finally, we discuss how the proposed data encoding naturally fits ensemble learning and how this improves a hybrid system's computational efficiency and classification performance. Besides introducing the concept, we motivate and formulate research questions, which we will answer with several sets of numerical simulations in Sect. 4.

The core of the randomized data structure is the Bloom filter utilized by GloBiMaps, and the hashing of the values incorporates the method shown in Eq. 5. Thus, the input data $x \in \mathcal{X}$ must be binarized and rasterized, which we generally denote as $g(x)$. Then, the binary raster created by $g(x)$ can be mapped to a Bloom filter B, which is a pseudo-random bit string of a power of two sizes m, by k hash functions h_i in the form of

$$F\big(h_i\big(g(x)\big)\big) := 1 \tag{12}$$

where $i = 1, ..., k$ and $x \in \mathcal{X}$. Here, equivalent to Bloom filters, all elements b_j are initially set to 0, and then k elements are set to 1 when information is inserted.

By controlling the parameters m and k, the data structure allows trading the representation size with the FP rate p (Eq. 2). Thus, p can be viewed as an indicator of the quality of the representation or, the other way around, as the amount of purposively induced noise. To get a more intuitive view of the impact of p on the input, we present a qualitative visualization of the decoded data structure. Although it is theoretically impossible to invert the hash function h_i and reverse the randomization, it is practically possible to inverse the mapping. There are two straightforward ways. First, with a known input raster size and hash function, we can query set membership for each pixel in the form of $h_i\big(I_{ones}(a, b)\big)$ where I_{ones} is a raster of ones. Second, by saving the indices of binarized pixels $g(I(a, b))$ and the indices of b_j while mapping $g\big(I(a, b)\big) \mapsto b_j$, the hash functions h_i may be pseudo-inverted to retain the binary input value. Such an inversion can be generally described as follows:

$$h_i^{-1}(b_j) = g\big(I(a, b)\big) \in \{0, 1\} \tag{13}$$

and results in an archetype for the input. While this is possible for every kind of input data, it is best illustrative and convenient to understand in the case of images, as shown in Fig. 4. For this example, we use imagery of handwritten digits from Alpaydin and Kaynak (1998), initially introduced in Fisher (1936) (for a description of the dataset, see Sect. 4.1). It shows that the image can be error-free reconstructed from the filter with a low FP rate as $p = 0.02$ (Fig. 4c). On the other hand, for a high FP rate as $p = 0.64$ (Fig. 4f), the image is highly disturbed by false positives. However, even with a high error rate, since there can be no other error than an FP, the original structure of the encoded information will always be inside the image. Also, note that the FP pixels are not the same for all examples due to the randomization during hashing (Fig. 4 e, f). While this visualization gives an intuitive view of the influence of the FP rate, it can not reveal the impact of the FP rate on the performance of an algorithm. This induces our first research question: "How does the false positive rate of a Bloom filter impact a model's classification performance?"

Since all encoded samples are presented with binary precision, we can now map each $b_j \in [0, 1]$ directly to the quantum state with a state preparation circuit with a depth $\mathcal{O}(1)$, consisting of single-qubit rotational gates. One of N

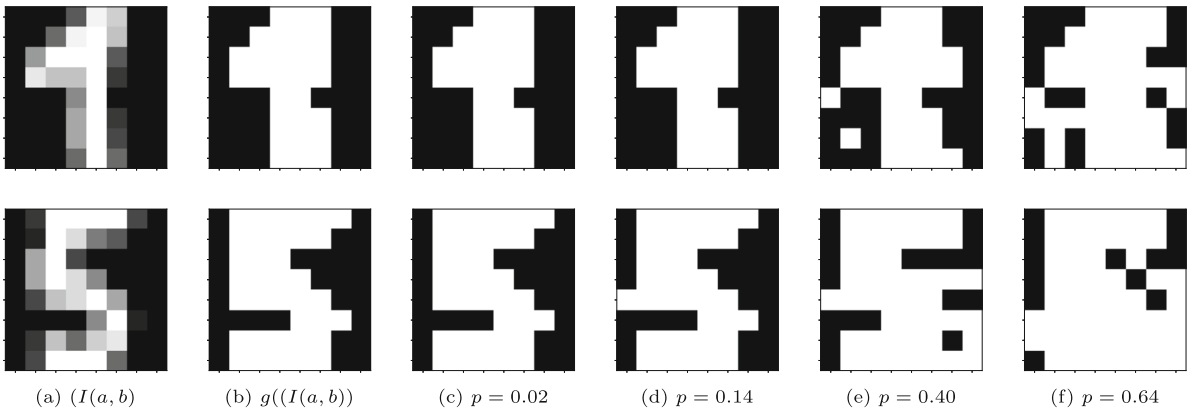


Fig. 4 Qualitative visualization of the impact of the FP rate p of a Bloom filter on an image $I(a, b)$. **a** Original image. **b** Binarized image. **c–f** Archetypes from pseudo-inverted hash functions $h_i^{-1}(b_j)$ with varying filter configuration

qubits then represents one of m bits of classical information, where $N = m$. Basis encoding with Pauli-X gates or angle encoding with a single rotational gate of the set $\{R_x, R_y, R_z\}$ or any other suitable single-qubit operation may be used to encode the input. Here, we chose basis encoding since it creates the best distinguishable quantum states. Following, the mapping of the bits to quantum state can be described as follows:

$$b_j \mapsto |b_j\rangle \tag{14}$$

which results in an orthogonal computational basis state. The quantum encoded data can then be processed by some arbitrary parameterized quantum circuit $\hat{U}_\theta(B)$ with a qubit register size N.

Regarding the number of inserted elements n and the filter size m, it must be clarified that filters with $m < n$ lead to unreasonable high FP rates. Thus, filters easily reach sizes where basis encoding is not feasible anymore due to the limitations of current quantum hardware since $N = m$ when reading the whole filter. One solution is to use only a fragment of the array as input for the model, which involves a loss of information. After generating the pseudo-random bit arrays by hashing the input, a random choice of N bits b_j from B may represent the encoded input x. Note that any selection of bits b_j from B is practically a random selection. Now, let $\hat{B}$, which has $N \ll m$ bits, denote a fragment, which is a random subset of the encoded binary input data B. While too much fragmentation would lead to insufficient features, a low fragmentation could contribute to the generalization performance of the model. Here, our second research question arises: "Can we sufficiently represent real-world data with fragments of randomized few-bit representations for classification?"

However, we can use even smaller fragments and further reduce the width of the parameterized quantum circuit by incorporating ensemble learning, where the pseudo-random data structure manifests as a natural fit. In ensemble learning, the training dataset $\mathcal{X}$ is commonly instanced such that every base model f_c learns from a random subset, which improves the diversity of the classifiers, enhances generalization performance, and helps to avoid overfitting. The proposed data structure opens up the possibility that every model f_c learns from a fragment with $N \ll m$ random bits from every encoded sample in $\mathcal{X}$. Hence, now, we consider base models $f_c(\hat{B}_c)$ with N qubits and trained with fragments $\hat{B}_c$. The fragments are stratified, each consisting of a sequence of N unique bits starting from the first element. Thus, we can guarantee that every bit of the filter B was seen by at least one model with fragments of size m/N while having a practically random selection of bits from the transformed input. Nonetheless, a higher number of classifiers improves the ensemble's performance, which can be achieved by overlapping the stratified fragments of the filter. Figure 5 shows exemplarily the proposed fragmentation of the transformed input for an ensemble of $2m/N - 1$ quantum classifiers. We train weak classifiers f_c with fragments of the encoded samples from the whole train set and build an ensemble model $f_{bag}(B)$. The final prediction of the quantum ensemble is then the collective decision $f_{bag}(B) = y_{bag}$ (Sect. 2.4). This leads to our third research question: "Does the fragmentation of a filter provide sufficient diversity for an ensemble model?"

As described in the previous section (Sect. 2.4), quantum ensembles enable less complex circuits and fewer qubits while improving classification performance by collective decision-making. While large ensembles are typically connected to high compute times, in the case of the simulation

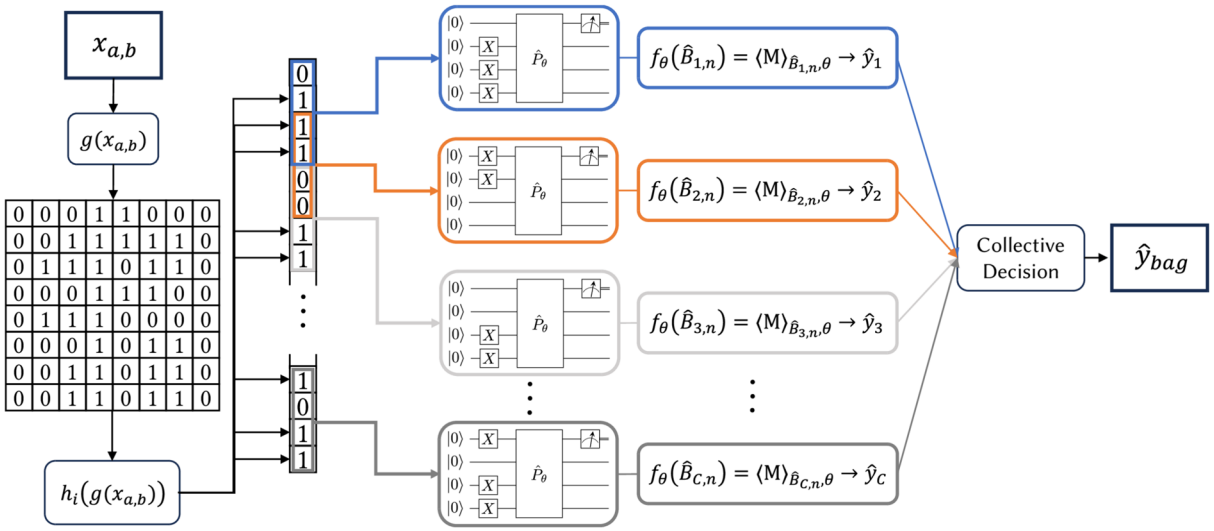


Fig. 5 Transformation and fragmentation of an input x for an ensemble of $2m/N-1$ quantum classifiers with $N=4$ qubits and basis encoding. The binarized raster $g(x)$ is mapped to a Bloom filter by hash functions h_i, and the stratified fragments $\hat{B}$ are classified by base classifiers f_θ. We obtain the final output $\hat{y}_{bag}$ according to a combination rule

of quantum circuits, our procedure improves the running time. Since the computational cost of simulating qubits grows exponentially with their number, an ensemble with a reduced number of qubits takes less compute time than fewer circuits with proportional more depth. Many less computationally intensive simulations are also advantageous because they can be easily parallelized on today's classical processing units.

Last, we want to mention a possible, less obvious application of the presented concept. Besides the advantages mentioned in the previous sections, the randomized data structure may be used for security and privacy in cloud computing. This can be accomplished by replacing the non-cryptographic Murmur3 hash function with a cryptographic one, for example, from the secure hash algorithm (SHA) family. A user could transform his data following the presented concept and only transmit the randomized bit arrays. Thus, no human-readable information would get to the operator of the quantum hardware while still being able to fully process the data. This can provide value if the available operators are not trustworthy.

4 Numerical simulations

We present numerical simulations to serve as proof of concept and demonstrate the representation power of the proposed data structure for application in quantum machine learning. More specifically, we answer the questions: "Can we sufficiently represent real-world data with fragments of randomized few-bit representations for classification?", "How does the false positive rate of a Bloom filter impact a model's classification performance?", and "Does the fragmentation of a filter provide sufficient diversity for an ensemble model?". We will describe the setup for the numerical simulations in Sect. 4.1 and then discuss the results and answer the previously defined questions in Sect. 4.2.

4.1 Setup

We conduct numerical simulations with two well-established benchmark datasets: the handwritten digits dataset from Alpaydin and Kaynak (1998), which was introduced in Xu et al. (1992), and the Iris flower dataset from Fisher (1988), which was introduced in Fisher (1936). The digits dataset contains grayscale images of handwritten numerals with 8×8 pixels. The Iris dataset contains plant features in the form of four float values with one decimal place for each sample. It has three classes where one class is linearly separable from the other, and the others are not linearly separable. Regarding the digits data, we chose classes one and five with 182 samples each for binary classification. For the sake of simplicity, we binarize the imagery with a threshold where every pixel value >0 equals 1 to create binary representations for GloBiMaps. Thus, the images are represented with 8×8 binary values. Regarding the Iris dataset, the two non-linearly separable classes with 50 samples each were chosen for binary classification. One sample consists of four data points with one decimal place value. To obtain binary representations for the Iris dataset, we first multiply the decimal values of each sample by 10 and thus obtain integers since they only have a single decimal value. Then, we can map the integers to seven-bit representations since all integers $x \in [0, 128]$.

Finally, the four features, with 7 bits each, are concatenated, and we obtain a 4×7 binary representation for each sample of the Iris dataset. For both datasets, a train-test-split is performed where $1/3$ of the digits samples and $1/5$ of the Iris samples are used for testing. For binary classification, all labels are converted to hinge labels $y \in \{-1, 1\}$.

As outlined in Sect. 3, we use the Murmur3 hash function to encode the binary representations into a Bloom filter. Several filter configurations in terms of mask size m and number of hash functions k are evaluated. Note that filters with $m < n$, where n is the number of bits of the input data, lead to unreasonably high FP rates, so only filters with $m > n$ were considered in the numerical simulations.

We use basis encoding, outlined in Sect. 2, for the quantum state preparation circuit because of its simplicity and efficiency in terms of state preparation circuit architecture and the excellent distinguishability of the generated quantum states $|0\rangle$ and $|1\rangle$.

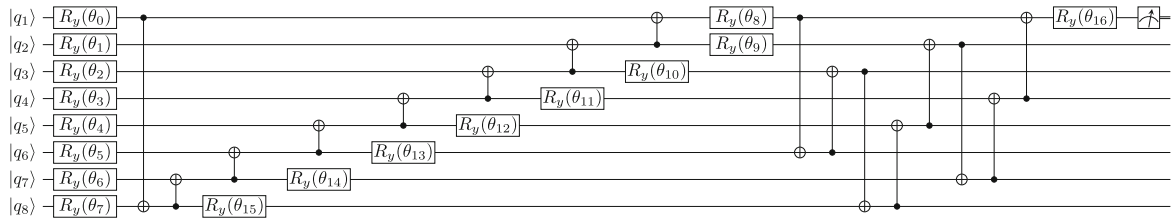
Two strongly entangled variational quantum circuit architectures and two classical approaches are considered for the supervised classification of the encoded data in quantum state. The quantum circuit architectures are based on previous work, namely the circuit-centric (CC) quantum classifier (Schuld et al. 2020) (Fig. 6b) and a multi-scale entanglement renormalization ansatz (MERA) (Vidal 2008) (Fig. 6a). These circuit architectures were chosen since they are widely accepted and used in literature and provide an appropriate reference. Both quantum circuits have low depth and only a few trainable parameters. In addition to the quantum models, we consider two classical approaches, support vector machine (SVM) and decision tree (DT), as a baseline comparison to validate the representation power of the data encoding.

As described in Sect. 2.3, a computational basis measurement is used to extract the label. Finally, we minimize the squared hinge loss function with adaptive moment estimation as the optimizer of choice. All quantum models train for 20 epochs with a learning rate of 0.01. The quantum circuits are simulated on a classical CPU using Tensorflow-quantum (Broughton et al. 2020). Due to the high number of low complexity circuits with small qubit registers, training multiple models in parallel is recommended, which we did with GNU Parallel (Tange 2011).

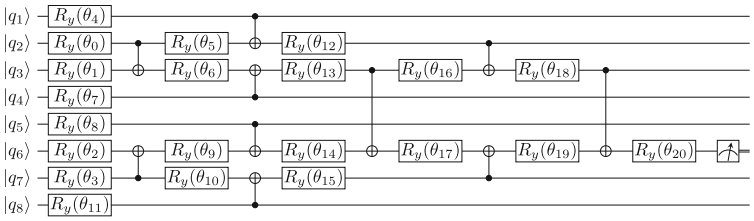
We consider two approaches to classify the filters: a classifier trained on fragments of the train data samples and an ensemble classifier where each classifier is trained on a distinct fragment of the train data samples. For the ensemble learning approach, we chose $2m/N - 1$ classifiers since this ensures overlap (as visualized in Fig. 5) and gives an uneven number of predictions $\hat{y}_c$. However, as demonstrated with Eq. 11, higher numbers of classifiers may further improve the accuracy. Independent of the classification approach, the size of the qubit register N equals the number of bits in $\hat{B}$. The ensemble learning approach follows classical bootstrap aggregating outlined in Sect. 2.4.

4.2 Results and discussion

All given results show the mean of five repetitions with their standard deviation. We conduct numerical simulations with varying qubit register sizes (Table 1), varying numbers of hash functions with a fixed filter size (Fig. 7), and varying numbers of hash functions and filter sizes to evaluate the impact of the FP rate (Figs. 8, 9).



(a) Circuit centric classifier (CC) based on ? with 8 qubits, a depth of 20, and 17 trainable parameters.



(b) Multi-scale entanglement renormalization ansatz classifier (MERA) based on ? with 8 qubits, a depth of 12, and 21 trainable parameters.

Fig. 6 Variational quantum circuits $\hat{P}_\theta$ as classifiers. Circuit architectures include single-qubit rotations $R_y(\theta) = exp(-i\frac{\theta}{2}Y)$, controlled-NOT operations (*CNOT*), and a computational basis measurement. The figures were generated with QPIC (Draper and Kutin 2017)

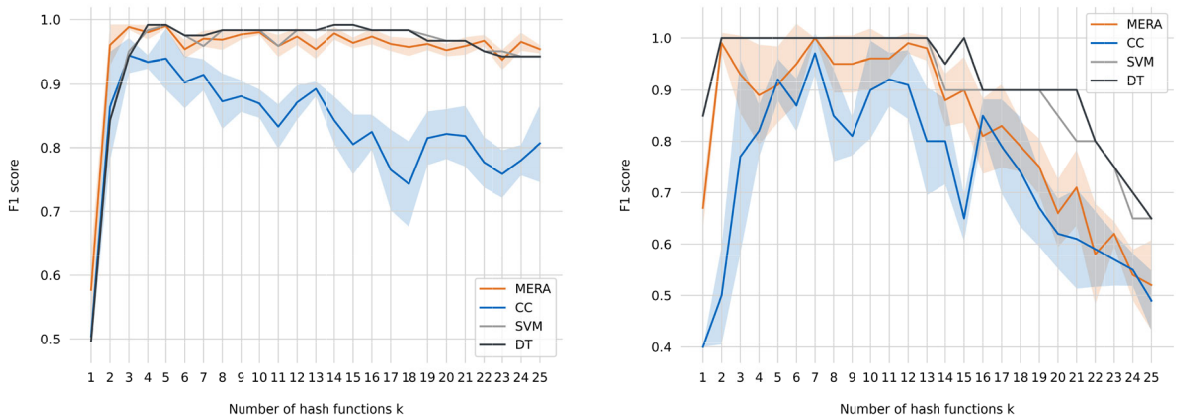
Table 1 Classification accuracy on the test dataset with varying qubit register size N for base and ensemble (Ens.) models

		Digits				Iris			
		Quantum		Classical		Quantum		Classical	
		CC	MERA	SVM	DT	CC	MERA	SVM	DT
Base	N=4	0.63	$0.62^{\pm 0.01}$	0.65	0.65	$0.70^{\pm 0.04}$	$0.71^{\pm 0.02}$	0.81	0.83
	N=8	$0.63^{\pm 0.01}$	$0.70^{\pm 0.01}$	0.73	0.73	$0.62^{\pm 0.05}$	$0.76^{\pm 0.06}$	0.91	0.92
	N=16	$0.65^{\pm 0.01}$	$0.73^{\pm 0.01}$	0.83	0.84	$0.55^{\pm 0.06}$	$0.77^{\pm 0.05}$	0.98	0.97
Ens	N=4	$0.9^{\pm 0.01}$	$0.9^{\pm 0.03}$	0.93	0.93	$0.93^{\pm 0.05}$	$0.94^{\pm 0.03}$	0.94	0.95
	N=8	$0.85^{\pm 0.01}$	$0.93^{\pm 0.01}$	0.93	0.93	$0.78^{\pm 0.11}$	$0.91^{\pm 0.04}$	1.0	0.95
	N=16	$0.72^{\pm 0.02}$	$0.88^{\pm 0.01}$	0.93	0.94	$0.55^{\pm 0.12}$	$0.9^{\pm 0.09}$	1.0	$0.97^{\pm 0.02}$

Filter configurations: Digits, $m = 64$ and $k = 1$; Iris, $m = 32$ and $k = 2$. Abbreviations: CC circuit-centric classifier, $MERA$ multi-scale entanglement renormalization ansatz, SVM support vector machine, DT decision tree

To begin with, we justify our choice for one of the two rules for collective decision-making described in Sect. 2.4: averaging and majority vote. We found that the majority vote gave better results since the magnitudes of the models' outputs $\hat{y}$ are often close to the decision boundary and do not indicate a specific certainty for the respective class. Thus, a single large output may outweigh multiple small outputs during averaging, which leads to a wrong result. Hence, the majority vote was used for the collective decision for all ensembles in the following numerical simulations.

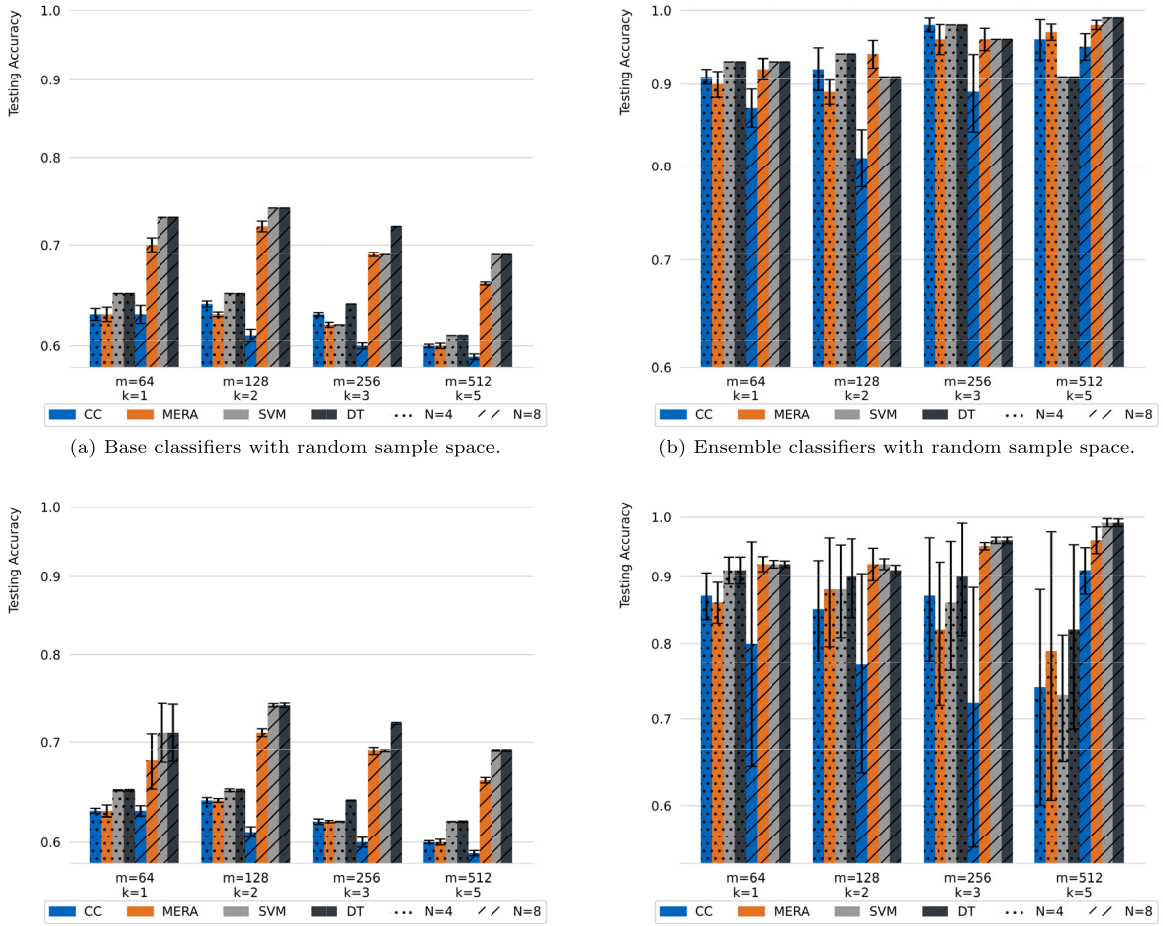
Table 1 gives first insights into the representation power of the randomized data transformation. The qubit register size is set to $N \in \{4, 8, 16\}$, as $N = 2$ offers too few possibilities and $N = 32$ is already too computationally intensive for simulation on classic hardware. Depending on N, we split the transformed data with $m = 64$ for the digits dataset and $m = 32$ for the Iris dataset in $2m/N - 1$ fragments for training and subsequent testing. We generally observe that, for base classifiers $f_c(\hat{B})$, the classification accuracy on the test set increases slightly with increasing N, both for quantum and classical approaches. However, even the base classifiers have a classification accuracy on the test set of around 65% for qubit registers of size $N \in \{4, 8, 16\}$. Thus, we can directly answer our first question "Can we sufficiently represent real-world data with fragments of randomized few-bit representations for classification?" positively. The majority of ensemble classifiers achieve accuracies of over 90% on the test data set. The results show that an increasing number of models has a stronger influence on the performance than an increasing number of qubits. Note that this is not the case for the classical ensembles built from SVMs or DTs, where the classification accuracy on the test set slightly increases the



(a) Transformed digits data with filter size $m = 512$. The theoretically computed optimal number of hash functions is $k^* = 5.55$.

(b) Transformed Iris data with filter size $m = 128$. The theoretically computed optimal number of hash functions is $k^* = 3.17$.

Fig. 7 The micro average of the F1 score and its standard deviation for different numbers of hash functions $k \in [1..25]$. All models learned from data in the form of $N = 8$ sized fragments of filters, and the visualized results are on the test set. Abbreviations: CC circuit-centric classifier, MERA multi-scale entanglement renormalization ansatz, SVM support vector machine, DT decision tree



(a) Base classifiers with random sample space.

(b) Ensemble classifiers with random sample space.

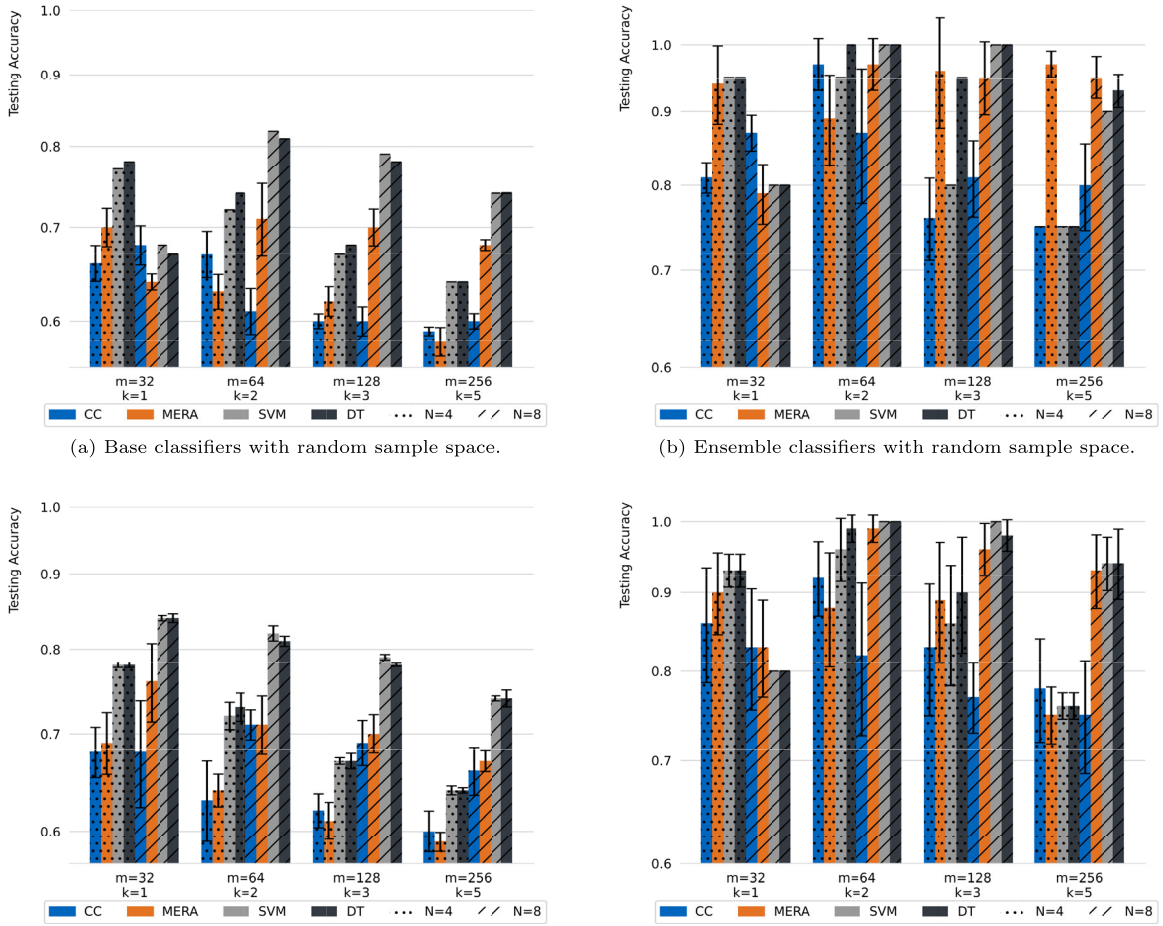
(c) Base classifiers with random sample space and random input space.

(d) Ensemble classifiers with random sample space and random input space.

Fig. 8 Classification accuracy on the test set for the digits dataset with varying filter configurations and qubit register size. FP rates p for the configurations from left to right: 0.64, 0.40, 0.14, 0.02. Abbreviations: CC circuit-centric classifier, MERA multi-scale entanglement renormalization ansatz, SVM support vector machine, DT decision tree

larger N gets. In addition, the quantum ensembles generally show a higher percentage improvement compared to base classifiers than the two classical models. While the SVMs and DTs generally show better performance than the quantum machine learning models in most simulations, a direct comparison of both approaches is not the aim of this paper. However, we can verify the transformed data's high expressiveness for both approaches and that the quantum models' performance in such a limited setting with few trainable parameters is promising.

Although the optimal number of hash functions k^* can be theoretically computed with Eq. 3, the choice of k for the optimal data representation for the classifiers is not obvious. Since k^* must be rounded to an integer value and the possible benefits of higher FP rates for the classifiers in terms of better generalization performance and training data variety, we present numerical simulations with $k \in [1..25]$. Figure 7 shows results from a total of 250 trained quantum ensemble models for each dataset, which amounts to 31750 base models for the digits dataset and 7750 base models for the Iris dataset. Furthermore, the same amounts of classical models were trained in the form of SVMs and DTs. A single hash function leads to high sparsity, which means almost empty filters, and results in low F1 scores. The classifiers yield similar performance with too-large k, which leads to an unreasonable high FP rate. However, the performance of the ensembles stays relatively stable with over 0.8 F1 score for $k \in [3..16]$ for digits in Fig. 7a and $k \in [4..13]$ for Iris data in Fig. 7b. We can observe that the rounded k^* does, for both datasets, not equal the actual optimum for k. Also, in Fig. 7b, a sharp drop can be seen at around $k = 14$, where the theoretically calculated error rate is $p > 0.5$. Hence, the optimum number



(a) Base classifiers with random sample space.

(b) Ensemble classifiers with random sample space.

(c) Base classifiers with random sample space and random input space.

(d) Ensemble classifiers with random sample space and random input space.

Fig. 9 Classification accuracy on the test set for the Iris dataset with varying filter configurations and qubit register size. FP rates p for the filter configurations from left to right: 0.58, 0.34, 0.11, 0.01. Abbreviations: CC circuit-centric classifier, MERA multi-scale entanglement renormalization ansatz, SVM support vector machine, DT decision tree

of hash functions should be treated as a hyperparameter and tuned depending on the data and classifier to achieve the best possible performance.

To answer the question "Does the fragmentation of a filter provide sufficient diversity for an ensemble model?", we present the following large sets of numerical simulations with fragmented bit arrays two times. First, with random sample space (Figs. 8 a, c, 9 a, c), which is the fragmentation of the filter, and second with random sample space and random input space, which is the additional selection of a random subset of the training data (Figs. 8 b, d, 9 b, d). Here, the parameter m was chosen to be as small as possible to emphasize the compression ability of the data transformation, and k was chosen close to the theoretically computed k^* while providing an expressive range of FP rates p. The comparison of random sample space and random sample space with random input space shows that the randomization induced by the data transformation is sufficient to provide diversity for the ensembles' base classifiers. Despite the additional randomization of the dataset, which intuitively could have led to enhanced generalizability, no improvement in classification accuracy on the test set can be seen. However, the additional randomization of the training subset increases the standard deviations of the five repetitions, which can be seen in Figs. 8 b, d and 9 b, d when compared to Figs. 8 a, c and 9 a, c.

Last, we want to answer "How does the false positive rate of a Bloom filter impact a model's classification performance?" As previously discussed, Fig. 7 shows a decrease in performance where the theoretically calculated FP rate exceeds 0.5. Nevertheless, even with $p = 0.64$ (Fig. 8) and $p = 0.58$ (Fig. 9), a classification accuracy of $> 90\%$ on the test dataset can be realized with ensemble models in some instances for $N = 4, 8$. As expected, a slight decrease in classification accuracy on the test set with increasing filter

size m can be seen for the base classifiers in Figs. 8a, c, 9a, c as the part of the input seen by the classifier decreases with increasing filter size m. However, even with only $4/512$ of the total bits as input, the base classifiers still have classification accuracy around 60% on the test data. Further, it shows that an error rate of $p = 0.64$ for digits and $p = 0.58$ for Iris data is still viable for ensemble models, and multiple ensembles result in $> 90\%$ classification accuracy on the test set. Also notable, the difference in accuracy compared to the difference in FP rates in Figs. 8 and 9 is surprisingly low. One possible reason is that, since the only error is an FP, the information, although noisy, is fully preserved. Overall, this indicates that the FP rate can be used to improve the generalization of models by intentionally inducing noise in the form of the FPs. The FP rate can, therefore, be set surprisingly high while still achieving an acceptable classification performance, enabling small filters and further emphasizing the data structure's compression capability.

5 Conclusion

We developed and presented a technique for representing classical data for quantum state preparation based on Bloom filters and input randomization. The key building block is a pseudo-random bit array created by encoding binarized raster data with a non-cryptographic hash function. By allowing some bits of the transformed data to be false positives, the encoding enables trading the representation size with the representation accuracy. This can be seen as intentionally induced noise and can be used to improve variety for small training datasets and enhance the generalization performance of models. The bit array can be quantum encoded with basis encoding, resulting in well-distinguishable quantum states.

The paper demonstrates how the fragmentation of the transformed data, combined with ensemble learning, enables the use of tiny quantum registers and state preparation circuits with a depth of $\mathcal{O}(1)$. With a series of simulations, we show that the transformation is powerful in representing data for quantum machine learning and naturally fits bootstrap aggregation. The small, practically random fragments of the transformed data are sufficient to train weak base classifiers, and the fragmentation of the randomized data structure provides the diversity needed to construct ensemble models, which often suffer from low variety in the training data subsets.

Expressive small-scale representations of high-dimensional data can be created that fit the input domain of current and near-term quantum hardware by tuning the filter parameters as additional model hyperparameters. The bit arrays are suitable for low-depth state preparation circuits since they rely only on a single quantum gate per bit to be encoded, and the fragmentation of the bit arrays further reduces the needed qubit register size. Thus, efficient parameterized quantum circuits with small qubit registers can be utilized. Note that quantum circuit simulations also profit from the data transformation since the computing cost of simulating qubits grows exponentially with their number, while the computing cost grows linearly with additional base classifiers with the same qubit register size.

This research contributes to the field of hybrid quantum-classical systems, presenting a data transformation that allows for low-depth and low-width parameterized quantum circuits. We emphasize how the data structure utilizes less complex circuits on quantum hardware, while quantum ensembles can further reduce the needed qubit register size while improving the classification performance. We see potential in randomized data encodings for quantum computing and propose further studies on lossy compression and intentionally induced noise for quantum machine learning. Novel data transformations for real-world data that allow efficient state preparation circuits to be used may be an important step towards a practical quantum advantage.

Acknowledgements We acknowledge our TUM Professorship Big Geospatial Data Management colleagues, who helped with proofreading and wording.

Author contribution M. W. proposed the initial idea and conceptualization for the research and contributed to the manuscript text and experiments. J. M. Z. developed the concept further, designed and carried out the experiments, analyzed and interpreted the results, wrote the main manuscript text, and prepared the figures. Both authors reviewed the contents of the work.

Funding Open Access funding enabled and organized by Projekt DEAL. Open Access funding provided by the Technical University of Munich.

Data availability References and information on the data used are included in the manuscript.

Declarations

Conflict of Interest The authors declare no competing interests.

References

Abdennebi A, Kaya K (2021) A bloom filter survey: variants for different domain applications. https://doi.org/10.48550/arXiv.2106.12189. arXiv:2106.12189

Adhikary S, Dangwal S, Bhowmik D (2020) Supervised learning with a quantum classifier using multi-level systems. Quantum Inf Process 19:1–12. https://doi.org/10.1007/s11128-020-2587-9

Alpaydin E, Kaynak C (1998) Optical recognition of handwritten digits. UCI machine learning repository. https://doi.org/10.24432/C50P49

Appleby A (2008) Murmurhash 2.0

Appleby A (2016) smhasher github repository. https://github.com/aappleby/smhasher/

Araujo IF, Park DK, Petruccione F et al (2021) A divide-and-conquer algorithm for quantum state preparation. Sci Rep 11(1):1–12. https://doi.org/10.1038/s41598-021-85474-1

Ashhab S (2022) Quantum state preparation protocol for encoding classical data into the amplitudes of a quantum information processing register's wave function. Phys Rev Res 4(1):013091

Benedetti M, Lloyd E, Sack S et al (2019) Parameterized quantum circuits as machine learning models. Quantum Sci Technol 4(4):043001. https://doi.org/10.1088/2058-9565/ab4eb5

Bloom BH (1970) Space/time trade-offs in hash coding with allowable errors. Commun ACM 13(7):422–426. https://doi.org/10.1145/362686.362692

Bose P, Guo H, Kranakis E et al (2008) On the false-positive rate of bloom filters. Inf Process Lett 108(4):210–213. https://doi.org/10.1016/j.ipl.2008.05.018

Broder A, Mitzenmacher M (2004) Network applications of bloom filters: a survey. Internet Math 1(4):485–509. https://doi.org/10.1080/15427951.2004.10129096

Broughton M, Verdon G, McCourt T et al (2020) Tensorflow quantum: a software framework for quantum machine learning. arXiv preprint. https://doi.org/10.48550/arXiv.2003.02989. arXiv:2003.02989

Caro MC, Gil-Fuster E, Meyer JJ et al (2021) Encoding-dependent generalization bounds for parametrized quantum circuits. Quantum 5:582. https://doi.org/10.22331/q-2021-11-17-582

Cerezo M, Sone A, Volkoff T et al (2021) Cost function dependent barren plateaus in shallow parametrized quantum circuits. Nat Commun 12(1):1791. https://doi.org/10.1038/s41467-021-21728-w

Chikhi R, Rizk G (2013) Space-efficient and exact de Bruijn graph representation based on a bloom filter. Algorithms Mol Biol 8:1–9. https://doi.org/10.1186/1748-7188-8-22

Christensen K, Roginsky A, Jimeno M (2010) A new analysis of the false positive rate of a bloom filter. Inf Process Lett 110(21):944–949. https://doi.org/10.1016/j.ipl.2010.07.024

Debnath B, Sengupta S, Li J et al (2011) Bloomflash: Bloom filter on flash-based storage. In: 2011 31st International conference on distributed computing systems. IEEE, pp 635–644. https://doi.org/10.1109/ICDCS.2011.44

Dietterich TG (2000) Ensemble methods in machine learning. In: International workshop on multiple classifier systems. Springer, pp 1–15. https://doi.org/10.1007/3-540-45014-9_1

Dilip R, Liu YJ, Smith A et al (2022) Data compression for quantum machine learning. Phys Rev Res 4(4):043007. https://doi.org/10.1103/PhysRevResearch.4.043007

Draper T, Kutin S (2017) Qpic: quantum circuit diagrams in latex

Esch T, Heldens W, Hirner A et al (2017) Breaking new ground in mapping human settlements from space-the global urban footprint. ISPRS J Photogramm Remote Sens 134:30–42. https://doi.org/10.1016/j.isprsjprs.2017.10.012

Farhi E, Neven H (2018) Classification with quantum neural networks on near term processors. https://doi.org/10.48550/arXiv.1802.06002. arXiv:1802.06002

Fisher RA (1988) Iris. UCI machine learning repository. https://doi.org/10.24432/C56C76

Fisher RA (1936) The use of multiple measurements in taxonomic problems. Ann Eugenics 7(2):179–188. https://doi.org/10.1111/j.1469-1809.1936.tb02137.x

Geravand S, Ahmadi M (2013) Bloom filter applications in network security: a state-of-the-art survey. Comput Netw 57(18):4047–4064. https://doi.org/10.1016/j.comnet.2013.09.003

Gil Vidal FJ, Theis DO (2020) Input redundancy for parameterized quantum circuits. Front Phys 8:297. https://doi.org/10.3389/fphy.2020.00297

Gremillion LL (1982) Designing a Bloom filter for differential file access. Commun ACM 25(9):600–604. https://doi.org/10.1145/358628.358632

Havlíček V, Córcoles AD, Temme K et al (2019) Supervised learning with quantum-enhanced feature spaces. Nature 567(7747):209–212. https://doi.org/10.1038/s41586-019-0980-2

Hoefler T, Häner T, Troyer M (2023) Disentangling hype from practicality: on realistically achieving quantum advantage. Commun ACM 66(5):82–87. https://doi.org/10.1145/3571725

Incudini M, Grossi M, Ceschini A et al (2023) Resource saving via ensemble techniques for quantum neural networks. Quantum Mach Intell 5(2):39. https://doi.org/10.1007/s42484-023-00126-z

Jackman SD, Vandervalk BP, Mohamadi H et al (2017) Abyss 2.0: resource-efficient assembly of large genomes using a bloom filter. Genome Res 27(5):768–777. https://doi.org/10.1101/gr.214346.116

Kirsch A, Mitzenmacher M (2006) Less hashing, same performance: building a better bloom filter. In: Algorithms–ESA 2006: 14th Annual European Symposium, Zurich, Switzerland, September 11-13, 2006. Proceedings 14. Springer, pp 456–467. https://doi.org/10.1007/11841036_42

Kumar A, Xu J, Li L et al (2003) Space-code bloom filter for efficient traffic flow measurement. In: Proceedings of the 3rd ACM SIGCOMM conference on Internet measurement, pp 167–172. https://doi.org/10.1145/948205.948226

LaRose R, Coyle B (2020) Robust data encodings for quantum classifiers. Phys Rev A 102(3):032420. https://doi.org/10.1103/PhysRevA.102.032420

Le PQ, Dong F, Hirota K (2011) A flexible representation of quantum images for polynomial preparation, image compression, and processing operations. Quantum Inf Process 10:63–84. https://doi.org/10.1007/s11128-010-0177-y

Li J, Lin S, Yu K et al (2022b) Quantum k-nearest neighbor classification algorithm based on hamming distance. Quantum Inf Process 21(1):18. https://doi.org/10.1007/s11128-021-03361-0

Li G, Ye R, Zhao X et al (2022a) Concentration of data encoding in parameterized quantum circuits. Adv Neural Inf Process Syst 35:19456–19469. https://doi.org/10.48550/arXiv.2206.08273

Lu S, Braunstein SL (2014) Quantum decision tree classifier. Quantum Inf Process 13(3):757–770. https://doi.org/10.1007/s11128-013-0687-5

Luo L, Guo D, Ma RT et al (2018) Optimizing bloom filter: challenges, solutions, and comparisons. IEEE Commun Surv Tutorials 21(2):1912–1949. https://doi.org/10.1109/COMST.2018.2889329

Macaluso A, Clissa L, Lodi S et al (2020) Quantum ensemble for classification. https://doi.org/10.48550/arXiv.2007.01028. arXiv:2007.01028

McClean JR, Romero J, Babbush R et al (2016) The theory of variational hybrid quantum-classical algorithms. New J Phys 18(2):023023. https://doi.org/10.1088/1367-2630/18/2/023023

McClean JR, Boixo S, Smelyanskiy VN et al (2018) Barren plateaus in quantum neural network training landscapes. Nat Commun 9(1):4812. https://doi.org/10.1038/s41467-018-07090-4

Melsted P, Pritchard JK (2011) Efficient counting of k-mers in dna sequences using a bloom filter. BMC Bioinforma 12(1):1–7. https://doi.org/10.1186/1471-2105-12-333

Mitarai K, Negoro M, Kitagawa M et al (2018) Quantum circuit learning. Phys Rev A 98(3):032309. https://doi.org/10.1103/PhysRevA.98.032309

Mitzenmacher M (2001) Compressed bloom filters. In: Proceedings of the twentieth annual ACM symposium on Principles of distributed computing, pp 144–150. https://doi.org/10.1145/383962.384004

Mullin JK (1983) A second look at bloom filters. Commun ACM 26(8):570–571. https://doi.org/10.1145/358161.358167

Murphy KP (2022) Probabilistic machine learning: an introduction. MIT press

Nayak S, Patgiri R, Borah A (2021) A survey on the roles of bloom filter in implementation of the named data networking. Comput Netw 196:108232. https://doi.org/10.1016/j.comnet.2021.108232

Niu X, Ma W (2023) Selective quantum ensemble learning inspired by improved adaboost based on local sample information. Complex & Intelligent Systems, pp 1–11. https://doi.org/10.1007/s40747-023-00996-7

Pan Z, Feng Y, Li Z et al (2023) Understanding the impact of quantum noise on quantum programs. In: 2023 IEEE international conference on Software Analysis, Evolution and Reengineering (SANER). IEEE, pp 426–437. https://doi.org/10.1109/SANER56733.2023.00047

Patgiri R, Nayak S, Borgohain SK (2018) Preventing ddos using bloom filter: a survey. https://doi.org/10.4108/eai.19-6-2018.155865. arXiv:1810.06689

Pérez-Salinas A, Cervera-Lierta A, Gil-Fuster E et al (2020) Data re-uploading for a universal quantum classifier. Quantum 4:226. https://doi.org/10.22331/q-2020-02-06-226

Sagi O, Rokach L (2018) Ensemble learning: a survey. Wiley Interdiscip Rev Data Min Knowl Disc 8(4):e1249. https://doi.org/10.1002/widm.1249

Schuld M (2021) Supervised quantum machine learning models are kernel methods. https://doi.org/10.48550/arXiv.2101.11020. arXiv:2101.11020

Schuld M, Petruccione F (2018) Quantum ensembles of quantum classifiers. Sci Rep 8(1):2772. https://doi.org/10.1038/s41598-018-20403-3

Schuld M, Bergholm V, Gogolin C et al (2019) Evaluating analytic gradients on quantum hardware. Phys Rev A 99(3):032331. https://doi.org/10.1103/PhysRevA.99.032331

Schuld M, Bocharov A, Svore KM et al (2020) Circuit-centric quantum classifiers. Phys Rev A 101(3):032308. https://doi.org/10.1103/PhysRevA.101.032308

Sim S, Johnson PD, Aspuru-Guzik A (2019) Expressibility and entangling capability of parameterized quantum circuits for hybrid quantum-classical algorithms. Adv Quantum Technol 2(12):1900070. https://doi.org/10.1002/qute.201900070

Song H, Dharmapurikar S, Turner J et al (2005) Fast hash table lookup using extended bloom filter: an aid to network processing. ACM SIGCOMM Comput Commun Rev 35(4):181–192. https://doi.org/10.1145/1090191.1080114

Tange O (2011) Gnu parallel - the command-line power tool. USENIX Mag 36(1):42–47. http://www.gnu.org/s/parallel

Vidal G (2008) Class of quantum many-body states that can be efficiently simulated. Phys Rev Lett 101(11). https://doi.org/10.1103/PhysRevLett.101.110501

Weigold M, Barzen J, Leymann F et al (2021a) Encoding patterns for quantum algorithms. IET Quantum Commun 2(4):141–152. https://doi.org/10.1049/qtc2.12032

Weigold M, Barzen J, Leymann F et al (2020) Data encoding patterns for quantum computing. In: Proceedings of the 27th conference on pattern languages of programs, pp 1–11. https://doi.org/10.1145/3628797.3628946

Weigold M, Barzen J, Leymann F et al (2021b) Expanding data encoding patterns for quantum algorithms. In: 2021 IEEE 18th International Conference on Software Architecture Companion (ICSA-C). IEEE, pp 95–101. https://doi.org/10.1109/ICSA-C52384.2021.00025

Werner M (2019a) Globimaps-a probabilistic data structure for in-memory processing of global raster datasets. In: Proceedings of the 27th ACM SIGSPATIAL international conference on advances in geographic information systems, pp 3–12. https://doi.org/10.1145/3453184

Werner M (2019b) Globimaps github repository. https://github.com/mwernerds/globimap

Werner M (2021) Globimapsai: an ai-enhanced probabilistic data structure for global raster datasets. ACM Trans Spat Algorithms Syst 7(4):1–24. https://doi.org/10.1145/3453184

Xu L, Krzyzak A, Suen CY (1992) Methods of combining multiple classifiers and their applications to handwriting recognition. IEEE Trans Syst Man Cybern 22(3):418–435. https://doi.org/10.1109/21.155943

Zhang C, Ma Y (2012) Ensemble machine learning: methods and applications. Springer. https://doi.org/10.1007/978-1-4419-9326-7