Quantum Computing for Pattern Classification

Maria Schuld[1], Ilya Sinayskiy[1,2], and Francesco Petruccione[1,2]

[1] University of KwaZulu-Natal, Quantum Research Group,
Durban 4000, South Africa
schuld@ukzn.ac.za,
http://quantum.ukzn.ac.za
[2] National Institute for Theoretical Physics, KwaZulu-Natal,
Durban 4000, South Africa

Abstract. It is well known that for certain tasks, quantum computing outperforms classical computing. A growing number of contributions try to use this advantage in order to improve or extend classical machine learning algorithms by methods of quantum information theory. This paper gives a brief introduction into quantum machine learning using the example of pattern classification. We introduce a quantum pattern classification algorithm that draws on Trugenberger's proposal for measuring the Hamming distance on a quantum computer [CA Trugenberger, *Phys Rev Let* 87, 2001] and discuss its advantages using handwritten digit recognition as from the MNIST database.

Keywords: Machine learning, quantum computing, quantum artificial intelligence.

1 Quantum Methods for Machine Learning

With the rapid growth in the volume of data that is transferred, stored and processed on a daily basis, innovative methods of machine learning become more and more important. Supervised machine learning algorithms infer an input-output relation from large sets of training data that consist of 'correct' examples of mappings. In other words, the computer *learns* from experience how to treat new inputs. A prominent example where a mapping needs to be learned is the pattern classification problem, in which a new data vector has to be assigned to one of a number of classes, given a set of correctly classified data vectors. A data vector thereby contains information on the features of the entity that is to be classified (for example the clicking behaviour of an online user, the structure of a molecule or the pixel of an image), and is also called *feature vector*. Pattern classification is the abstract formulation of the problem of interpreting information, and it finds application in areas as diverse as information technology, the food industry or the financial sector. These tasks come to humans much more natural than to machines (e.g., when we recognise other humans as a response to the large amount of photons that enter our eyes in every second), and as a subdiscipline of artificial intelligence, machine learning is indeed inspired by the way of how our brain deals with data.

D.-N. Pham and S.-B. Park (Eds.): PRICAI 2014, LNAI 8862, pp. 208–220, 2014.

Quantum computing is a relatively new branch combining computer science and physics, in which the properties of small particles formulated in quantum theory are exploited to process information. Controlling quantum objects that encode information (so called qubits or qudits) is a highly nontrivial task, and the realisation of a mature quantum computer is still far from being accomplished. However, there is no lack of theoretical studies on the scope and power of quantum information. As part of these efforts, quantum information scientists recently realised that quantum computing could improve classical machine learning algorithms in three basic ways. First, subroutines that are costly on a classical computer when subjected to big data - such as the evaluation of an inner product or searching for a minimum distance - could be executed on a quantum computer with a linear or even exponential speedup in complexity due to quantum parallelism [18,23,35]. Second, from the perspective of quantum computing, quantum machine learning (from here on QML) opens up new possibilities for quantum information processing, such as quantum state classification [30,12]. Third, especially in the area of intelligent agents and reinforcement learning, quantum physics offers unique types of logics that is often compared to fuzzy logics [24,6].

The advantage of computing with quantum objects is that data can theoretically be represented exponentially more compact in a so called quantum superposition of both the 0 and 1 state. On the downside, information retrieval is limited by the laws of measurement of a quantum system, which is a destructive process that changes it substantially preventing us from assessing all information at once [27]. (However, as we will see later the probabilistic nature of the outputs to quantum algorithms can be valuable for pattern classification). The last decade of quantum information research provided a 'toolbox' of algorithms that can be implemented on a potential quantum computer, which are the building blocks used to tackle the more sophisticated problems of QML. Here, we will add to these methods and propose a quantum pattern classification algorithm for binary feature vectors, which follows the principle of a distance weighted k-nearest neighbour method [9]. Our idea uses a variation of Trugenberger's [32] subroutine to determine the Hamming distance between two binary patterns on a quantum computer.

This paper is organised as follows. We will first give a brief introduction to quantum computing which can be skipped by readers familiar with quantum information theory. We then outline the problem of pattern recognition (Section 3). In Section 4 we briefly introduce into distance-based methods of pattern classification such as *k-nearest neighbours*, translate the problem into the language of quantum physics and give an example of a quantum classification algorithm. We discuss its merit using handwritten digit recognition and give a general outlook in the context of quantum machine learning.

2 Computing with Quantum Objects

Quantum computing analyses the manipulation of quantum objects in order to solve computational problems[1]. A 'quantum object' thereby refers to any particle or system of particles for which Newton's mechanics proves to be an insufficient description while quantum theory explains our observations. This is becomes important for the description of microscopic particles such as atoms, electrons or photons, and allows for entirely new ways of information processing on a microscopic scale.

The equivalents to bits on a classical computer are quantum objects with two distinct configurations or *states*, called qubits[2], which can have various physical implementations such as the energy of atoms or the polarisation of photons. But if bits are either carrying a signal encoding a 0 or 1, qubits use the superposition principle of quantum objects to be in both states 'at the same time'. In the notation for quantum states, this looks like

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle\,, \qquad |\alpha|^2 + |\beta|^2 = 1,$$

where α, β are complex numbers called *amplitudes* and $|\cdot\rangle$ represents a state vector describing a quantum object. Later on the *phase* ϕ of a qubit becomes important, which is a part of the amplitude $\alpha = \tilde{\alpha}e^{i\phi}$. Quantum theory is built around the observation that the squared amplitudes $|\alpha|^2, |\beta|^2$ denote the probability to measure the qubit either in state $|0\rangle$ or $|1\rangle$. A qubit state is thus not characterised by whether it is in the '0' or '1' state, but by *how likely it is to measure it in either of them*. Computations can work on both states at the same time, a fact that is often referred to as quantum parallelism.

The power of quantum information processing becomes apparent if we consider a system of n qubits each with the two available states $\{|0\rangle\,, |1\rangle\}$. The quantum system can be put into a superposition of all 2^n combinations $\{|00...00\rangle\,, |00...01\rangle\,, ..., |11...11\rangle\}$ and an algorithm can work on all these configurations in parallel. However, quantum computing is always constrained by the probabilistic nature of the results, as well as the destruction caused by measurement. After a qubit has been measured to be either $|0\rangle$ or $|1\rangle$, the state 'collapses' into the measurement result and will subsequently only produce the same output. We can therefore only access a limited amount of information from the result, and the output is of probabilistic nature (i.e. evolving and measuring the same system several times produces a distribution of results, of which the most likely result can be regarded as the output of the computation). This is why it is rather difficult to come up with powerful algorithms for a quantum computer [22].

It is important for the following to introduce some formal basics of quantum information theory, but the interested reader shall be referred to [22]. The discrete states of a quantum object (such as the above mentioned polarisation or energy level) are mathematically modelled as vectors in a d-dimensional Hilbert

[1] For a comprehensive introduction to quantum computing, see [22].
[2] Note that *qudits* would be the generalisation to d-dimensional states.

space $\mathcal{H}_d$. For qubits, d equals 2, and a system of n qubits that encodes a binary string of the same length can be described by vectors in $\mathcal{H}_2 \otimes ... \otimes \mathcal{H}_2$ (remember that the d-dimensional generalisation of a qubit is then called a 'qudit'). Transformations from one vector to another that obey the general laws of quantum theory are represented by unitary operators U with the property $U^\dagger U = 1$ where $U^\dagger$ is the hermitian conjugate. These unitary transformations define the dynamics of the quantum system and quantum algorithms can be represented by a sequence of such operations on an input quantum state.

In quantum computation, these unitary transformations are called 'quantum gates', since they correspond to classical gates that manipulate bits. Some standard 1-qubit gates are the X-gate that flips the state of a qubit, the Z-gate that changes the sign of its amplitude, or the Hadamard or H-gate that creates a superposition $\frac{1}{\sqrt{2}}(|0\rangle \pm |1\rangle)$ from $|0\rangle$ $(+)$ or $|1\rangle$ $(-)$ respectively. A central 2-qubit gate is the controlled-NOT operation cNOT which only flips the state of a second qubit if the first one is in state $|1\rangle$. The cNOT together with standard single qubit gates form in fact a universal set for quantum computation [22]. A more general formulation for a quantum gate that is derived from fundamental quantum theory based on the Schrödinger equation can be described by a unitary transformation $U = e^{-i\mathrm{H}t}$ where H denotes a hermitian operator called Hamiltonian.

3 The *k*-Nearest Neighbour Algorithm for Pattern Classification

In a pattern classification problem we want to assign one out of a number of classes to a pattern, according to a rule learned from a set of example classifications. It is thus a problem of *supervised* learning, or learning from training data. This abstract formulation contains an impressive range of important decision problems in real life. For example, a doctor diagnosing a disease given a number of symptoms and his experience from other cases, an email being automatically marked as 'important' or 'spam' on the grounds of previous emails, or a handwritten digit on a postal envelope being recognised by a scanning device. Even more, our human thinking process can be described through decision problems, for example when we 'recognise' (= classify) people, things and smells around us, or when we classify a situation as dangerous or not depending on sensual stimuli. Some authors replace the term pattern classification by *pattern recognition*, which is a more generalised expression as it also looks at the problem of seeing patterns without classifying it, as well as *template matching* or *associative memory*, in which a close example from learnt data is retrieved upon an input.

Describing the pattern classification problem more precisely, we are given a set of n-dimensional data vectors $\boldsymbol{v}^k$ and their class assignments c^p, $\mathcal{T} = \{(\boldsymbol{v}^p, c^p)\}_{p=1,...,N}$ that makes up the 'training set' to our problem. Each of the vectors encodes n features v_i^p. These features may represent the grey shade of a certain pixel, information on whether a patient has had cancer in his or her family, or the number of times a certain word occurs in a text sample.

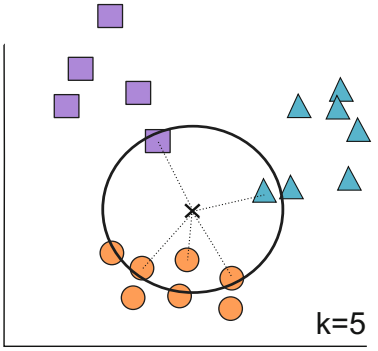

Fig. 1. (Colour online) Illustration of the kNN method of pattern classification. The new vector (black cross) gets assigned to the class that the majority of its k closest neighbours have (in this case it would be the orange circle shape). In this example, k is set to 5.

The features are given as by binary, integer or real-valued numbers, while the class c^p of a feature vector is often encoded by a finite number d of positive integers $c \in \{1, ..., d\}$. Also given is an unclassified input vector $\boldsymbol{x}$ from the same vector space as the training vectors, encoding n features. The task of pattern classification is to match the new vector $\boldsymbol{x}$ to a class, using information from the training data. This is usually done by defining some distance measure and assigning the new input vector to the class whose members are the most 'similar' in terms of this distance. A common distance measure is the Euclidean metric or in case of binary features, the Hamming distance [3] (the number of differing bits on two binary strings [13]).

The discipline of machine learning developed a number of algorithms to solve the problem of pattern classification. One the most famous is the k-nearest neighbour (kNN) method [14,25]. Given a training set $\mathcal{T}$ stored in a memory, the k training vectors closest to the input vector are selected. The class to which the majority of these neighbours belong consequently gets assigned to the input vector (see Fig. 1). There are many variations to this simple method. For example, in the distance-weighted kNN, the neighbours get weighted by their distance to the input vector, so that closer neighbours make a bigger contribution to which class gets selected [9]. Another variation includes to preprocess the training data and calculate the centroid $\bar{\boldsymbol{v}}_c = |\boldsymbol{x} - \frac{1}{L}\sum_{l \in c} \boldsymbol{v}^l|$ of each class $c \in \{1, ..., d\}$ ($l = 1, ..., L_c$ is the index for the members of class c). The input vector then becomes part of the class with the closest centroid vector.

The advantage of the kNN method and its variations is not only their simplicity. They are *nonparametric* examples of supervised learning, since they do not require initial information on the distribution of vectors [8]. The only assumption on the data used is that *similar inputs have similar outputs* [3] (in our case, similar input vectors should be in general members of the same class). An important task is to choose an optimal parameter k, and in the original kNN method, a balance between noise reduction and maintaining the locality information has

to be found (for example, for $k \to$ all, then the class assignment would always result in the class with the most members). The distance-weighted kNN version has the advantage of being independent of the choice of k, as the "number of nearest neighbors is implicitly hidden in the weights" [15]. The quantum algorithm introduced in the following is based on the same principle as kNN, namely assuming that 'close' feature vectors carry the class that is to be assigned to the new vector.

4 Quantum Pattern Classification

Translating the pattern classification problem into the language of quantum physics reads as follows. Our feature data set is represented by quantum states $\{|v_1^p ... v_n^p, c^p\rangle\} \in \mathcal{H}_2^{\otimes n} \otimes \mathcal{H}_d$ where $p = 1, ..., N$ runs over the training states and the class of feature vector $\boldsymbol{v}^p$ is stored in the qudit $|c^p\rangle$. The product state of n 2-dimensional Hilbert spaces thereby represents the feature space, while the additional qudit in $\mathcal{H}_d$ encodes the d possible classifications. The input vector is a quantum state $|x_1, ..., x_n\rangle$ from the feature Hilbert space $\mathcal{H}_2^{\otimes n}$. As we are now dealing with quantum information, classical data either has to be translated into quantum states, or -as suggested in [11,18]- taken from some form of a quantum random access memory (especially if the machine learning algorithm is a subroutine to a larger computation on a future quantum computer).

4.1 Related Work

Many of the textbook machine learning methods already faced attempts to be translated into quantum physics (for a detailed review, see [28]). Amongst them are support vector machines [23], decision trees [20], principal component analysis [19], learning from membership queries [31], neural networks [34,36] and clustering [2,16,1]. Most contributions are dedicated to pattern recognition or classification tasks [18,23,10,21,27,30,35,29,33]. Some of these proposals are based on the idea of taking a computationally expensive subroutine from an original machine learning algorithm and executing it more efficiently on a quantum computer [18,35,29,23]. In [27,30] we find an attempt to use the insights of Bayesian decision theory for the classification of unknown quantum states. Some use adiabatic quantum computing to solve a learning optimisation problem [21,18]. A number of contributions also try to execute classical distance measures through quantum computation [18,23,35,33]. Finally, some authors emphasize the observation that the theory of open quantum systems is close to machine learning methods based on Markov models [7,4]. Despite this growing number of contributions, quantum machine learning is still a premature discipline, which derives its relevance from its potential to extend machine learning by a new paradigm, rather than from a given theoretical foundation. Although touched upon in several articles [26,17], there is yet no fundamental theory of how quantum information can in general be exploited for intelligent forms of computing. The expression 'quantum learning' [5,26,12] is so far used interchangeably with

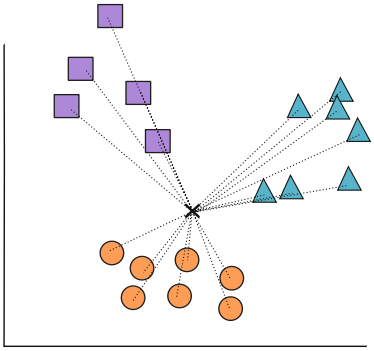

Fig. 2. (Colour online) Illustration of the principle on which the quantum pattern classification is based. The new vector (black cross) gets assigned to the class with the closest members. As in Figure 1, this would be the class of orange circles.

the term 'quantum machine learning' and simply refers to the various ideas brought forward in order to integrate quantum information into methods of machine learning or vice versa.

4.2 A Quantum Pattern Classification Algorithm

The quantum pattern classification (QPC) algorithm we present here uses the same distance-based classification principle as kNN, only that instead of chosing nearest neighbours, the distance of the entire training vector set is considered (see Figure 2). It draws on an algorithm presented in the context of quantum associative memory in [32]. The idea is to create a superposition of the training data set and 'write' the Hamming distance to the input state into the amplitude of each vector in the superposition. Measuring the class-qudit then retrieves the desired class with the highest probability. Even more, if repeated enough times to achieve statistical significance, the algorithm leads to a probability distribution containing information on how close each class members are to the input vector. Note that the following requires an understanding of the circuit model of quantum computing that was touched upon in Section 2, and readers not sufficiently familiar with quantum information theory might prefer to only consider the result in Eq (2) and the discussion thereafter.

The initial step of the algorithm is to construct a 'training set superposition' containing the training data,

$$|T\rangle = \frac{1}{\sqrt{N}} \sum_p |v_1^p ... v_n^p, c^p\rangle \, .$$

While this 'training phase' is trivial in the classical case, the efficient preparation of a quantum system in an arbitrary initial state is still an open problem, and also questions of quantum memory devices have not been resolved yet. However, algorithms to construct the initial state from a ground state can be found in

[32,34] and have linear complexity, just as accessing each bit from a classical memory would have. From this we construct the initial state

$$|\psi_0\rangle = \frac{1}{\sqrt{N}} \sum_p |x_1 ... x_n; v_1^p ... v_n^p, c^p; 0\rangle \,.$$

It is made of three registers, the first containing the input state, the second containing the superposition $|T\rangle$ and the third containing an ancilla qubit set to zero. In the first step, the ancilla is put into a superposition through a Hadamard gate, leading to

$$|\psi_1\rangle = \frac{1}{\sqrt{N}} \sum_i |x_1 ... x_n; v_1^p ... v_n^p, c^p\rangle \otimes \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle).$$

For reasons of readibility we factor the ancilla state out for now. Following [32], in a second step the Hamming distance between each qubit of the first and second register,

$$d_k^i = \begin{cases} 1, & \text{if } |v_k^p\rangle = |x_k\rangle \,, \\ 0, & \text{else,} \end{cases}$$

replaces the qubits in the second register. This is done by applying an cNOT(a, b)-gate (see Section 2) which overwrites the second entry b with 0 if $a = b$ and else with 1. We use the X gate to reverse the states in the second register, since in the end we want a strong 'signal' for small Hamming distances. Note that the gates have no effect on the class and ancilla states. The second step consequently reads

$$\begin{aligned} |\psi_2\rangle &= \prod_k \mathrm{X}(x_k)\ \mathrm{cNOT}(x_k, v_k^p)\ \ |\psi_1\rangle \\ &= \frac{1}{\sqrt{N}} \sum_p |x_1 ... x_n; d_1^p ... d_n^p, c^p\rangle \otimes \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle). \end{aligned}$$

We then use the unitary operator

$$U = e^{-i\frac{\pi}{2n}H} \qquad \text{with} \qquad H = 1 \otimes \sum_k \left(\frac{\sigma_z + 1}{2}\right)_{d_k} \otimes 1 \otimes (\sigma_z)_c \,,$$

to sum up the reverse single-qubit Hamming distances d_k^p of each training vector $|v_1^p ... v_n^p\rangle$ in order to write the total reverse Hamming distance between input vector and the pth training vector, $\bar{d}_H(\boldsymbol{x}, \boldsymbol{v}^p)$, into the phase of the ith state of the superposition (together with a negative sign if the ancilla qubit is $|1\rangle$). The state after the third step is consequently given by

$$\begin{aligned} |\psi_3\rangle = U\,|\psi_2\rangle = \frac{1}{\sqrt{2N}} \sum_p & e^{i\frac{\pi}{2n}\bar{d}_H(\boldsymbol{x},\boldsymbol{v}^p)}\,|x_1 ... x_n; d_1^p ... d_n^p, c^p; 0\rangle \\ & + e^{-i\frac{\pi}{2n}\bar{d}_H(\boldsymbol{x},\boldsymbol{v}^p)}\,|x_1 ... x_n; d_1^p ... d_n^p, c^p; 1\rangle \,. \quad (1) \end{aligned}$$

Another Hadamard transformation on the ancilla state, $H = 1 \otimes 1 \otimes 1 \otimes H_a$, writes the phase information into the amplitudes,

$$|\psi_4\rangle = H\,|\psi_3\rangle = \frac{1}{\sqrt{N}} \sum_p \cos\left[\frac{\pi}{2n}\bar{d}_H(\boldsymbol{x}, \boldsymbol{v}^p)\right] |x_1...x_n; d_1^p...d_n^p, c^p; 0\rangle + \sin\left[\frac{\pi}{2n}\bar{d}_H(\boldsymbol{x}, \boldsymbol{v}^p)\right] |x_1...x_n; d_1^p...d_n^p, c^p; 1\rangle\,.$$

The ancilla does not only allow for this trick, but also gives us a possibility to test if the Hamming distance between the input we aim to classify, $|x\rangle$, and the states $|v^p\rangle\,,\ p = 1, ..., P$ is on average large or small. If the new input is far away from most training patterns, we have a much higher probability to measure the ancilla in the state $|1\rangle$, if the input is close to many patterns we end up in state $|0\rangle$. Trugenberger in his quantum associative memory only accepts inputs that have a sufficiently high probability of an ancilla state in $|0\rangle$, arguing that only in this case an associative memory can be reliable. Although our QPC algorithm should not rely on the average distance between the input and the training vectors, for the following retrieval step we have to measure the ancilla until we get a $|0\rangle$ in order to retrieve the cosine part of the sum. Obviously, the closer the input is to the training set, the more likely that we succeed. Our simulations show that the probability for this measurement,

$$P(0_a) = \frac{1}{N} \sum_p \cos^2\left[\frac{\pi}{2n}\bar{d}_H(\boldsymbol{x}, \boldsymbol{v}^p)\right],$$

is higher than $\frac{2}{3}$ for standard examples like the MNIST[3] handwritten digit database.

There are two versions of how to proceed, one that corresponds to a "$k \to$ all" method assigning the class of vectors that are on average closer to the input, and another version that measures the pattern register and retrieves neighbours with a probability weighed by their distance, and chooses the class most represented by this pool.

Following the first version, the last step is a measurement of the class-qudit along the standard basis. This step varies from [32], in which step two gets reversed in order to measure along the basis of the training vectors and retrieve the most likely (i.e. close) candidate. However, for classification problems we are fortunately not interested in the actual features of the nearest neighbours of $|x_1...x_n\rangle$, but merely in their class assignment. In superposition $|\psi_4\rangle$, the different classes appear weighted by their member's distance to the input that is to classify. This is obvious if we rewrite state $|\psi_4\rangle$ as

[3] The *Mixed National Institute of Standards and Technology* database is a collection of handwritten digits and can be retrieved from `http://yann.lecun.com/exdb/mnist/` [last visit 19/9/2014].

$$|\psi_4\rangle = \frac{1}{\sqrt{N}} \sum_{c=1}^{d} |c\rangle \otimes \sum_{l \in c} \cos\left[\frac{\pi}{2n} \bar{d}_H(\boldsymbol{x}, \boldsymbol{v}^l)\right] |x_1 ... x_n; d_1^l ... d_n^l; 0\rangle + \sin\left[\frac{\pi}{2n} \bar{d}_H(\boldsymbol{x}, \boldsymbol{v}^l)\right] |x_1 ... x_n; d_1^l ... d_n^l; 1\rangle ,$$

where l runs over all training vectors classified with the label c. The probability to measure a certain class $c \in \{1, ..., d\}$ provided we previously measured the ancilla in 0 is given by

$$\mathrm{P}(c) = \frac{1}{NP(0)} \sum_{l \in c} \cos^2\left[\frac{\pi}{2n} \bar{d}_H(\boldsymbol{x}, \boldsymbol{v}^l)\right], \tag{2}$$

a value that scales with the average Hamming distance between the input and all training vectors in this class. If we measure the class qudit of a sufficient number of copies of superposition $|\psi_4\rangle$, we can consequently retrieve the optimal class label for $|x_1 ... x_n\rangle$. This can be further processed as classical information, or as a new training vector $\left|v_1^{P+1}, ..., v_n^{P+1}, c^{P+1}\right\rangle = |x_1 ... x_n, c_x\rangle$ if we discard the qubits $d_1 ... d_n$ in the second register.

The second version would go as in [32,33], only that we are not interested in the closest training vector, but in the class of a number of close vectors. As in kNN, we assign the class that is the most represented amongst the neighbours. The difference to the classical algorithm is thereby that we do not necessarily pick the k nearest neighbours, but any neighbours with a probability that is proportional to their proximity to the input vector.

5 Discussion

The quantum pattern classification algorithm sketched above runs in polynomial time $\mathcal{O}(TPn)$[4] where n is the size of the feature vectors, P is the number of training examples and T is a accuracy threshold. More precisely, we have $4n + 2$ operations for the retrieval algorithm (the unitary U can be decomposed into $2n$ elementary operations [33]), which we run T times to get a sufficiently precise picture from the measurement results. The construction of the superposition lies in $\mathcal{O}(Pn)$ [34,33]. As a rough comparison, the classical kNN also has to compute the distance to all P n-dimensional training examples, which leads to a similar complexity class. An interesting point is that if we find a more efficient way to construct the superposition $|T\rangle$ in $\mathcal{O}(n)$, or receive it from a quantum memory device, the quantum version of this pattern classification algorithm would be independent of the number of training vectors, something that seems impossible to achieve in a classical version. In addition to this, the distance weighting (assigning a weight to each neighbour) does not require an additional step, but is 'combined' with the measuring of the distances.

[4] The complexity of a quantum algorithm is measured through the number of elementary gates that have to be applied to simulate the quantum evolution.

Fig. 3. Example of an ambiguous input image for the task of handwritten digit recognition. The image is taken from the training set of the MNIST database and shows the original (left) and binarised (right) example of a handwritten '9' that can easily be recognised as a 0 or a 9.

To illustrate another advantage of the quantum pattern classification algorithm, we consider the problem of recognising (in other words, classifying) handwritten digits, for example from the above mentioned MNIST handwritten digit database. Of course, running the algorithm on the unpreprocessed data of the binarised grey-shade pixel is only as successful as any classical algorithm executing a majority decision based on the Hamming distance between input and training vectors. This is without question a rather imprecise approach and our simulations show that approximately 50% of the digits of a test set of 100 examples can be classified correctly by this method (using a training set of 400 examples), a value that can be slightly improved by a scaling parameter ϵ introduced through a global phase shift in Eq (1). Still, the MNIST example helps to demonstrate how quantum computing offers a general advantage in cases of ambiguous inputs. Consider an image of a handwritten 9 that is easily mistaken for a 0, especially when applying a rough classification method based on the Hamming distance (see Figure 3). The classical kNN algorithm would lead to a deterministic output of either 0 or 9. On the other hand, repeating the quantum algorithm several times would lead to a distribution of outputs governed by Eq (2), and we would expect the 0 and 9 to be almost equally frequent. As a consequence, the quantum algorithm produces additional information on the quality of the judgement in a classification task. In other words, the probabilistic output of quantum algorithms presents an asset for pattern classification, as can be shown through a method as simple as the one presented here. Future works would have to extend the quantum algorithm to allow for continuous inputs, and investigate ways to exploit its advantages using more complex distance measures. This is beyond the scope of this publication, in which we merely intend to demonstrate the potential of pattern classification through quantum information. In general, methods of quantum machine learning might become an important extension to the field of machine learning, and create an exciting opportunity for both quantum physicists and computer scientists.

Acknowledgements. This work is based upon research supported by the South African Research Chair Initiative of the Department of Science and Technology and National Research Foundation.

References

1. Aïmeur, E., Brassard, G., Gambs, S.: Machine learning in a quantum world. In: Lamontagne, L., Marchand, M. (eds.) Canadian AI 2006. LNCS (LNAI), vol. 4013, pp. 431–442. Springer, Heidelberg (2006)
2. Aïmeur, E., Brassard, G., Gambs, S.: Quantum clustering algorithms. In: Proceedings of the 24th International Conference on Machine Learning, pp. 1–8. ACM (2007)
3. Alpaydin, E.: Introduction to machine learning. MIT Press (2004)
4. Barry, J., Barry, D.T., Aaronson, S.: Quantum pomdps. arXiv preprint arXiv:1406.2858 (2014)
5. Bonner, R., Freivalds, R.: A survey of quantum learning. In: Quantum Computation and Learning, p. 106 (2003)
6. Busemeyer, J.R., Bruza, P.D.: Quantum models of cognition and decision. Cambridge University Press (2012)
7. Clark, L.A., Huang, W., Barlow, T.M., Beige, A.: Hidden quantum markov models and open quantum systems with instantaneous feedback. arXiv preprint arXiv:1406.5847 (2014)
8. Duda, R.O., Hart, P.E., Stork, D.G.: Pattern classification. John Wiley & Sons (2012)
9. Dudani, S.A.: The distance-weighted k-nearest-neighbor rule. IEEE Transactions on Systems, Man and Cybernetics (4), 325–327 (1976)
10. Gambs, S.: Quantum classification. arXiv preprint arXiv:0809.0444 (2008)
11. Giovannetti, V., Lloyd, S., Maccone, L.: Quantum random access memory. Physical Review Letters 100(16), 160501 (2008)
12. Guţă, M., Kotłowski, W.: Quantum learning: Asymptotically optimal classification of qubit states. New Journal of Physics 12(12), 123032 (2010)
13. Hamming, R.W.: Error detecting and error correcting codes. Bell System Technical Journal 29(2), 147–160 (1950)
14. Harrington, P.: Machine learning in action. Manning Publications Co. (2012)
15. Hechenbichler, K., Schliep, K.: Weighted k-nearest-neighbor techniques and ordinal classification (2004)
16. Horn, D., Gottlieb, A.: Algorithm for data clustering in pattern recognition problems based on quantum mechanics. Physical Review Letters 88(1), 018702 (2002)
17. Hunziker, M., Meyer, D.A., Park, J., Pommersheim, J., Rothstein, M.: The geometry of quantum learning. arXiv preprint quant-ph/0309059 (2003)
18. Lloyd, S., Mohseni, M., Rebentrost, P.: Quantum algorithms for supervised and unsupervised machine learning. arXiv preprint arXiv:1307.0411 (2013)
19. Lloyd, S., Mohseni, M., Rebentrost, P.: Quantum principal component analysis. arXiv preprint arXiv:1307.0401v2 (2013)
20. Lu, S., Braunstein, S.L.: Quantum decision tree classifier. Quantum Information Processing 13(3), 757–770 (2014)
21. Neven, H., Denchev, V.S., Rose, G., Macready, W.G.: Training a large scale classifier with the quantum adiabatic algorithm. arXiv preprint arXiv:0912.0779 (2009)
22. Nielsen, M.A., Chuang, I.L.: Quantum computation and quantum information. Cambridge university press (2010)
23. Rebentrost, P., Mohseni, M., Lloyd, S.: Quantum support vector machine for big feature and big data classification. arXiv preprint arXiv:1307.0471 (2013)
24. Rigatos, G.G., Tzafestas, S.G.: Neurodynamics and attractors in quantum associative memories. Integrated Computer-Aided Engineering 14(3), 225–242 (2007)

25. Rogers, S., Girolami, M.: A first course in machine learning. CRC Press (2012)
26. Sasaki, M., Carlini, A.: Quantum learning and universal quantum matching machine. Physical Review A 66(2), 022303 (2002)
27. Sasaki, M., Carlini, A., Jozsa, R.: Quantum template matching. Physical Review A 64(2), 022317 (2001)
28. Schuld, M., Sinayskiy, I., Petruccione, F.: Quantum machine learning, Contemporary Physics (2014), doi:10.1080/00107514.2014.964942
29. Schützhold, R.: Pattern recognition on a quantum computer. arXiv preprint quant-ph/0208063 (2002)
30. Sentís, G., Calsamiglia, J., Muñoz-Tapia, R., Bagan, E.: Quantum learning without quantum memory. Scientific Reports 2 (2012)
31. Servedio, R.A., Gortler, S.J.: Equivalences and separations between quantum and classical learnability. SIAM Journal on Computing 33(5), 1067–1092 (2004)
32. Trugenberger, C.A.: Probabilistic quantum memories. Physical Review Letters 87, 067901 (2001)
33. Trugenberger, C.A.: Quantum pattern recognition. Quantum Information Processing 1(6), 471–493 (2002)
34. Ventura, D., Martinez, T.: Quantum associative memory. Information Sciences 124(1), 273–296 (2000)
35. Wiebe, N., Kapoor, A., Svore, K.: Quantum nearest-neighbor algorithms for machine learning. arXiv preprint arXiv:1401.2142 (2014)
36. Zhou, R., Ding, Q.: Quantum pattern recognition with probability of 100%. International Journal of Theoretical Physics 47(5), 1278–1285 (2008)