DEIM Forum 2020 F3-4

Support Vector Machine for Multiclass Classification using Quantum Annealers

BA DEMA[†], Junya ARAI[†], and Keitarou HORIKAWA[†]

† NTT Software Innovation Center, Nippon Telegraph and Telephone Corporation
3-9-11 Midori-cho, Musashino-shi, Tokyo, 180–8585, Japan
E-mail: †{tokuma.tomoe.py,junya.arai.at,keitarou.horikawa.sd}@hco.ntt.co.jp

Abstract Quantum annealing is easier to realize compared with universal gate quantum computing, and there are already commercialized quantum annealers, such as D-Wave 2000. Quantum annealing can efficiently solve combinatorial optimization problems. However, to expand the applications of quantum annealing, some studies have begun to focus on how to use quantum annealing for other problems such as machine learning algorithm. We have developed a support vector machine algorithm that solves multiclass classification problems on a quantum annealer. We evaluated our method on three-class synthetic data and a benchmark dataset (IRIS). The results show that our method can classify multiclass data at a precision comparable to that of classical implementations.

Key words Ising model, quantum annealing, support vector machine (SVM), machine learning

1 Introduction

With the end of Moore's Law, it has become difficult for traditional silicon-based computers to significantly improve their computing performance. However, a new generation of computers such as quantum computers is beginning to prosper. Both universal gate model quantum computing [1] and quantum annealing [2] may be able to improve computing performance in the future. Quantum annealing is specialized for combinatorial optimization problems [3], and it is easier to realize compared with universal gate quantum computing. There are already some commercialized quantum annealers, such as D-Wave 2000 [4], and quantum-inspired annealers, such as the Fujitsu Digital Annealer (DA) [5].

To expand the applications of quantum annealing, some studies have begun to focus on how to use quantum annealing for other problems such as binary classification [6] and other machine learning problems [8–11]. Some studies [12,13] proposed binary quantum support vector machines SVMs, i.e., quantum versions of a famous machine learning method, and there are more multiclass classification problems than binary ones. In this paper, we propose a multiclass classification SVM algorithm that can be used on a quantum annealer. Our main idea is to compare the energy values obtained by quantum annealing between multiple binary classifiers to find the largest margin between each class. We evaluated our method using three-class synthetic data and a benchmark dataset (IRIS [14]) on a quantum annealer and using simulated annealing [15]. The results show that our method can classify multiclass data at a precision comparable to classical implementations.

This paper is organized as follows. Section 2 introduces the background of quantum annealers and binary classification algorithms for classical and quantum SVM. In Section 3, we introduce our multiclass classification algorithm. Section 4 describes the results of using the proposed method in experiments on synthetic data and a benchmark dataset. We conclude this paper in Section 5.

2 Preliminaries

Here, we start by briefly introducing quantum annealing and the classical SVM. Then we introduce the quantum SVM algorithm for binary classification.

2.1 Quantum Annealing

Quantum annealing (QA) is a metaheuristic for finding the global minimum of a given objective function over a given set of candidate solutions, by a process using quantum fluctuations. QA-based computers are called quantum annealers. QA has also led to the development of quantum-inspired annealers that use classical algorithms to imitate quantum fluctuations. Both types of annealer use the same problem formulation, either an Ising model [16,17] or quadratic unconstrained binary optimization (QUBO) [18]. Thus, an algorithm running on a quantum-inspired annealing machine can be directly applied to a quantum annealer. In this study, we used QUBO to express the optimization problem. A QUBO problem can be represented as one of minimizing an energy function,

$$E = \sum_{i \leqq j} a_i Q_{ij} a_j \tag{1}$$

where $a_i \in \{0, 1\}$ are the binary variables of the optimization problem and Q_{ij} is the QUBO weight matrix, which is an upper-triangular matrix of real numbers. We use quantum annealers or quantum-inspired annealing machines to solve the QUBO and obtain a_i, which is the solution to the optimization problem.

2 2 The classical SVM

Support vector machines [19] are the most popular classification algorithm in machine learning. The basic principle of SVM is to find a hyperplane which separates the dataset into two classes. The aim of an SVM algorithm is to maximize the margin between data points and the hyperplane. When the dataset is not linearly separable, kernel tricks [20,21] are used. A kernel trick projects the data points into a feature space that is linearly separable by hyperplanes.

Training an SVM is equivalent to solving a quadratic programming (QP) problem [20]

$$\begin{aligned}
\text{minimize} \quad & E = \frac{1}{2}\sum_n^N \sum_m^N \alpha_n \alpha_m t_n t_m k(\mathrm{x}_n, \mathrm{x}_m) \\
& \quad - \sum_n \alpha_n & (2) \\
\text{subject to} \quad & 0 \leqq \alpha_n \leqq C & (3) \\
\text{and} \quad & \sum_n \alpha_n t_n = 0 & (4)
\end{aligned}$$

where N is the number of points in the dataset, $\mathrm{x}_n \in \mathbb{R}^d$ is a point in a d-dimensional space dataset, $t_n = \pm 1$ is the target label assigned to x_n, $\alpha_n \in \mathbb{R}$ denotes the members of a set of N coefficients, C is a regularization parameter, and $k(\cdot, \cdot)$ is the kernel function.

The coefficients α_n are determined using SVM; they define a $(d-1)$-dimensional hyperplane that can separates $\mathbb{R}^d$ into two regions corresponding to the predicted class label. The hyperplane is determined by the points x_n corresponding to $\alpha_n \neq 0$, which is called the support vector of the SVM.

There are several kernels that are often used in the kernel trick. We use a Gaussian kernel (also known as a radial basis function kernel or rbf kernel). It is defined by

$$\mathrm{rbf}(\mathrm{x}_n, \mathrm{x}_m) = e^{-\gamma \|\mathrm{x}_n - \mathrm{x}_m\|^2} \tag{5}$$

where the hyperparameter $\gamma > 0$ is usually determined in a calibration procedure prior to the training phase. In the following sections, we use cSVM(C, γ) to denote the classical SVM defined by Eqs. (2)-(4).

2 3 Binary classification quantum SVM

As a quantum annealer or quantum-inspired annealing machine can only produce discrete binary solutions in the form of a QUBO (Eq. (1)), we need to encode real numbers to binary values. The previous study [12] encodes α_n as follows:

$$\alpha_n = \sum_{k=0}^{K-1} B^k a_{Kn+k} \tag{6}$$

where $a_{Kn+k} \in \{0, 1\}$ are binary variables whose value is obtained by a quantum annealer, K is the number of binary variables used to encode α_n, and B is the encoding base. They obtained good results for $B = 2$ or $B = 10$ and a small K.

By using Eq. (6), the QP problem in Eqs. (2)-(4) can be encoded to

$$\begin{aligned}
E = & \frac{1}{2} \sum_{n,m,k,j} a_{Kn+k} a_{Km+j} B^{k+j} t_n t_m k(\mathrm{x}_n, \mathrm{x}_m) \\
& - \sum_{n,k} B^k a_{Kn+k} + \xi \Big(\sum_{n,k} B^k a_{Kn+k} t_n\Big)^2 & (7) \\
= & \sum_{n,m=0}^{N-1} \sum_{k,j=0}^{K-1} a_{Kn+k} \tilde{Q}_{Kn+j, Km+j} a_{Km+j} & (8)
\end{aligned}$$

where the multiplier ξ is a squared penalty term, and $\tilde{Q}$ is a $KN \times KN$ matrix:

$$\begin{aligned}
\tilde{Q}_{Kn+j, Km+j} = & \frac{1}{2} B^{k+j} t_n t_m (k(\mathrm{x}_n, \mathrm{x}_m) + \xi) \\
& - \delta_{nm} \delta_{kj} B^k & (9)
\end{aligned}$$

Since the QUBO formulation requires an upper-triangular matrix. $Q_{ij} = \tilde{Q}_{ij} + \tilde{Q}_{ji}$ for $i < j$ and $Q_{ii} = \tilde{Q}_{ii}$. The two constraints in Eqs. (3) and (4) are also included in the encoded energy function (Eqs. (7)-(8)). Since the maximum for α_n is given by

$$C = \sum_{k=0}^{K-1} B^k \tag{10}$$

the constraint $\alpha_n \geqq 0$ in Eq. (3) is automatically included in Eqs. (7)-(8). The constraint $\sum_n a_n t_n = 0$ in Eq. (4) is also included in constraint term $\xi(\sum_{nk} B^k a_{Kn+k} t_n)^2$ in Eq. (7). The coefficient ξ guarantees this term to be greater than or equal to zero. When $\sum_{nk} B^k a_{Kn+k} t_n \neq 0$, the energy of QUBO will increase. However, because QA is an algorithm that finds the lowest energy of all the candidate solutions, this term will be restricted to zero through QA.

When α_n are obtained by using QA, we get the decision boundary through

$$f(\mathrm{x}) = \sum_n^N \alpha_n t_n k(\mathrm{x}_n, \mathrm{x}) + b, \tag{11}$$

Here, $\sum_n^N a_n t_n = 0$ mathematically corresponds to an optimal bias b in the decision function Eq. (11). A reasonable choice of b is given by [20]

$$b = \frac{\sum_n^N \alpha_n (C - \alpha_n)[t_n - \sum_m^N \alpha_m t_m k(\mathrm{x}_m, \mathrm{x}_n)]}{\sum_n^N \alpha_n (C - \alpha_n)}. \tag{12}$$

The previous study [12] showed that $\xi = 0$ suffices to get a reasonable result, and that $\xi > 0$ produces equally good results.

In the following sections, we refer to the quantum SVM defined by Eq. (8) as qSVM.

3 qSVM for Multi-class classification

SVM is fundamentally a binary classifier. However, there are many multiclass problems in real applications. Two approaches are commonly used to make binary SVM recognize multiclass data: *one-versus-the-rest* [24] and *one-versus-one*.

In this section, we first introduce the main idea of our method; we then define the problems and show how to solve them on quantum annealers.

3 1 Main idea

Our main idea is to compare the energy values (Eq. (8)) obtained by quantum annealing between multiple binary classifiers to find the largest margin between each class. This enables our multiclass classification qSVM method, which follows the *one-versus-the-rest* or *one-versus-one* approach, to separate data into multiple classes when the result is uncertain. In the next two sections, we describe how to construct a multiclass qSVM classifier from a binary qSVM and then how to realize one on a quantum annealer or quantum-inspired annealing machine.

3 2 Multiclass classification problems

The *one-versus-the-rest* method is usually implemented using a "winner-takes-all" strategy. Assuming we have N_c classes, this strategy constructs N_c separate binary classifiers in which the n_c^{th} classifier $y_{n_c}(x)$ is trained using the data points of class C_{n_c} as positive examples and the data points of the remaining $N_c - 1$ classes as negative examples. Here, N_c is the number of classes. According to [22], this strategy makes predictions in accordance with

$$y(x) = \max_{n_c} y_{n_c}(\mathrm{x}). \tag{13}$$

However, this may lead to inconsistent results wherein one input datum is assigned to multiple classes.

The other approach, *one-versus-one*, is usually implemented using a "max-wins" voting (MWV) strategy. It constructs $N_c(N_c-1)/2$ binary classifiers for all possible pairs of distinct classes; then, the classes which have the most votes are the prediction result. The classifier C_{ij} uses examples from class C_i as positive examples and examples from class C_j as negative ones. However, this approach also faces a problem that there may be classes with the same number of votes.

3 3 Multiclass Classification qSVM

Using the two multiclass approaches mentioned above, we can use multiple binary qSVMs to construct a multiclass qSVM. However, both the *one-versus-the-rest* approach and the *one-versus-one* approach face certain problems. In particular, when the obtained results are inconsistent or the number of votes is the same, it is difficult to judge which class the data belongs to. Therefore, our multiclass classification qSVM uses the *one-versus-the-rest* and *one-versus-one* strategies to build a multiclass qSVM and then uses the minimum energy obtained from a quantum annealer or quantum-inspired annealing machine to find the separating hyperplane between classes. When the results are inconsistent or the number of votes is the same, we compare the energies of each binary classifier and assign the uncertain data to a certain class. This approach aims to find the largest margin between each class, and this strategy is also the basic idea of SVM. Take account of this method, we can achieve a multiclass classification qSVM on a quantum annealer or quantum-inspired annealing machine.

To show how to find the largest margin, we will start with a simple two-class classification linear model. The model is

$$y(x) = \mathrm{w}^T \phi(\mathrm{x}) + b \tag{14}$$

where $\phi(\mathrm{x})$ denotes a fixed feature-space transformation, and we make the bias parameter b explicit. The task of finding the largest margin in this model can be simplified to maximizing $\parallel \mathrm{w} \parallel^{-1}$; that is, we can solve the optimization problem by

$$arg \min_{\mathrm{w},b} \frac{1}{2} \parallel \mathrm{w} \parallel^2 \tag{15}$$

which is derived in [22]. To solve this constrained optimization problem, we use Lagrange multipliers $\alpha_n \geqq 0$ and construct the Lagrangian function [22],

$$L(\mathrm{w}, b, \mathrm{a}) = \frac{1}{2} \parallel \mathrm{w} \parallel^2 - \sum_{n=1}^{N} \alpha_n \{t_n(\mathrm{w}^T \phi(\mathrm{x}_n) + b) - 1\} \tag{16}$$

Then using the *Karush-Kuhn-Tucker* (KKT) conditions to solve for the Lagrangian function, we get

$$\max_{\alpha} \qquad W(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} y_i y_j \alpha_i \alpha_j \langle \mathrm{x}_i, \mathrm{x}_j \rangle \tag{17}$$

$$\text{subject to} \qquad 0 \leqq \alpha_i \leqq C, i = 1, ..., m \tag{18}$$

$$\text{and} \qquad \sum_{i=1}^{m} \alpha_i y_i = 0 \tag{19}$$

where we write $W(\alpha)$ as a function of α, and $\langle \mathrm{x}_i, \mathrm{x}_j \rangle$ is a kernel function when using the kernel trick. In this paper, we use the rbf kernel (Eq. (5)).

According to the derivation above, to find the largest margin between classes, we only need to solve Eq. (17)-(19).

From Eqs. (7)-(8), we know that the margin will be largest when the energy in Eq. (7) is at a minimum. Thus, the minimum energy obtained by the quantum annealer can assign uncertain results to a certain class. We use this strategy to decide the prediction label in the prediction time of our method. When using the *one-versus-the-rest* approach, we choose the label of positive examples of the classifier with the lowest energy as the prediction result for the inconsistent result.

$$t_{predict} = t_{\arg\min E_{C_{n_c}}}, n_c \in \{1, ..., N_c\} \qquad (20)$$

where $E_{C_{n_c}}$ is the energy of the classifier C_{n_c}. Moreover, when using *one-versus-one* approach, we choose the label with the smallest average energy of the classifier as the result when the number of votes are the same.

$$t_{predict} = t_{\arg\min avgE_{C_{n_c}}}, n_c \in \{1, ..., N_c(N_c-1)/2\} \quad (21)$$

4 Experiments

To verify the binary qSVM concept presented in the previous study and the characteristics of the hyperparameters, we first conducted a two-class experiment using synthetic data. Then, we evaluated our method on three-class synthetic data and a well-known benchmark dataset (IRIS). The existing quantum annealers and quantum-inspired annealing machines will offer different results owing to their having different random initial values due to quantum randomness. Therefore, we performed several experiments under the same conditions on each dataset. In addition, all the experiments on qSVM were conducted on the quantum-inspired annealing machine (DA) described in Section 2.

4 1 Two-class synthetic data

First, to prove the concept of the binary qSVM, we evaluated the binary qSVM on a small set of two-dimensional synthetic data, which was generated using scikit-learn.make.moon (sklearn) [26]. Fig. 1 is the recognition result optimized by DA, and Fig. 2 is the result using cSVM.

In the figures, color is used to distinguish classes, and the edges between the colored areas indicate the decision boundaries. The results in Fig. 1 show that the synthetic data cannot be separated when $\gamma = 0.1$ and $\gamma = 1$. On the other hand, qSVM recognized the synthetic data with 100% accuracy when $\gamma = 10$. γ is a hyperparameter of the rbf kernel, and it defines how far the influence of a single training example reaches, with low values meaning " far " and high values meaning " close ". When γ is small, the curvature of the decision boundary is very low, and thus, the decision region is very broad and the data may be underfitted. When γ is high, the curvature of the decision boundary is high, and the data are easily overfitted. Therefore, we can see that γ works

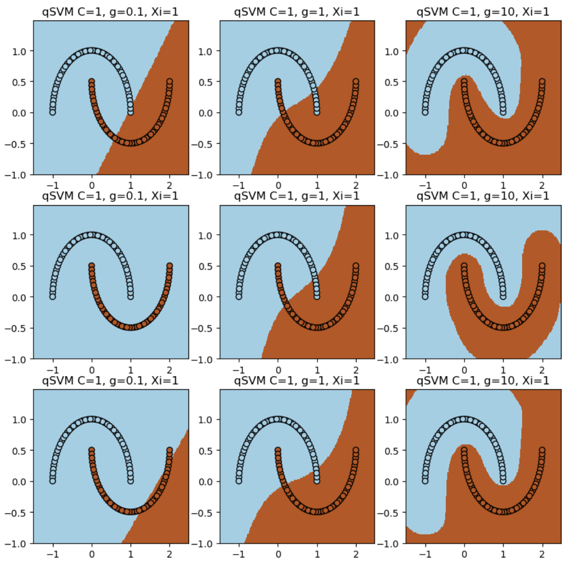


Figure 1 Recognition results of qSVM using DA. Results in the same column have the same hyperparameters, which represent three separate results optimized by quantum annealing for the same parameters. The results on the same line are for different γ.

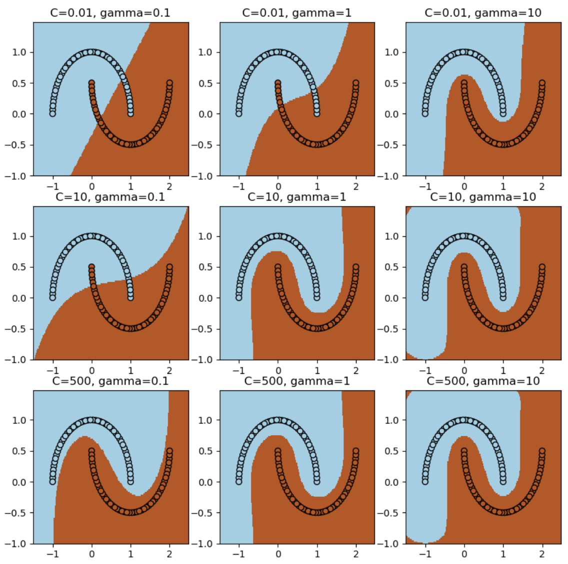
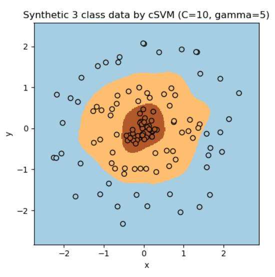


Figure 2 Recognition results of cSVM for different C and γ.

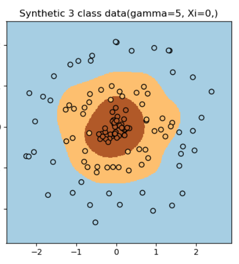
in qSVM as it does in cSVM.

4 2 Three-class synthetic data

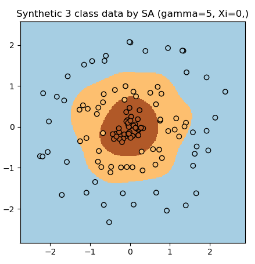
To prove that our algorithm is accurate, we tested it on linearly inseparable three-class synthetic data. The dataset consisted of $N = 120$ points, and each class had 40 points. The data were generated according to



(a) cSVM



(b) qSVM on DA



(c) qSVM on SA

Figure 3 Recognition results of three-class synthetic data for cSVM ($C = 10, \gamma = 5$), qSVM ($\gamma = 5, \xi = 0, B = 10, K = 3$) on DA and SA.

Table 1 Accuracy of different methods on three-class synthetic dataset

	cSVM	qSVM on DA	qSVM on SA
1	1.000	0.975	0.975

$$\mathrm{x}_n = r_n \begin{pmatrix} \cos\phi_n \\ \sin\phi_n \end{pmatrix} + \begin{pmatrix} s_n^x \\ s_n^y \end{pmatrix}, \qquad (22)$$

where $r_n = 2, 1, and\ 0.15$ when $t_n = 0, 1, and\ 2$, respectively. ϕ_n was linearly spaced on $[0, 2\pi)$ for each class, and s_n^x and s_n^y were drawn from a normal distribution with mean 0 and standard deviation 0.2.

We experimented with this dataset on cSVM and qSVM. The visualized results and accuracies are in Fig. 3 and Table 1. Each color represents one class, and the boundaries of the colored areas are the decision boundaries of SVM. Fig. 3(a) shows the recognition results of cSVM. It shows that cSVM separated the data into three classes and had high generalization ability. Fig. 3(b) and Fig. 3(c) are experimental results of qSVM on DA and SA, respectively. From these figures, we can see that DA and SA obtain similar solutions with low energy. (The parameters of DA were automatically tuned using a FUJITSU web API). qSVM also separates the data into three classes with high generalization performance, but with slightly lower accuracy. Both cSVM and qSVM used the same γ value. The results of the experiments described in the previous subsection also indicated that the γ of qSVM is more sensitive than that of cSVM; consequently, qSVM is more likely to underfit the data.

4 3 Iris dataset

Next, we evaluated our method on a famous benchmark

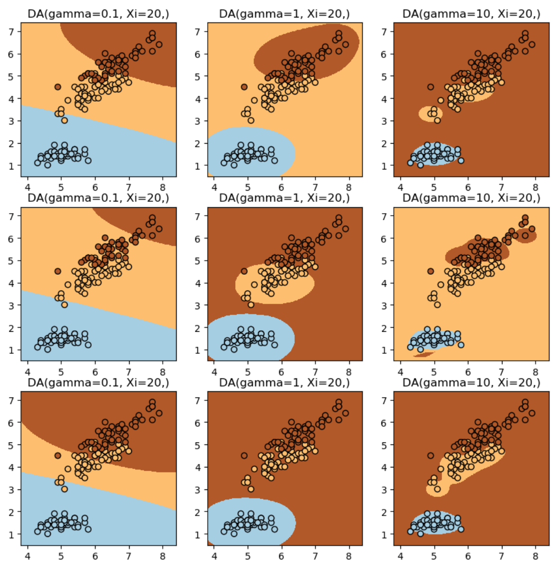


Figure 4 Recognition results of qSVM (B=10, K=3) using DA. Results in the same column have the same hyperparameters; they are three separate results optimized by quantum annealing for the same parameters. The results on the same line are for different values of γ.

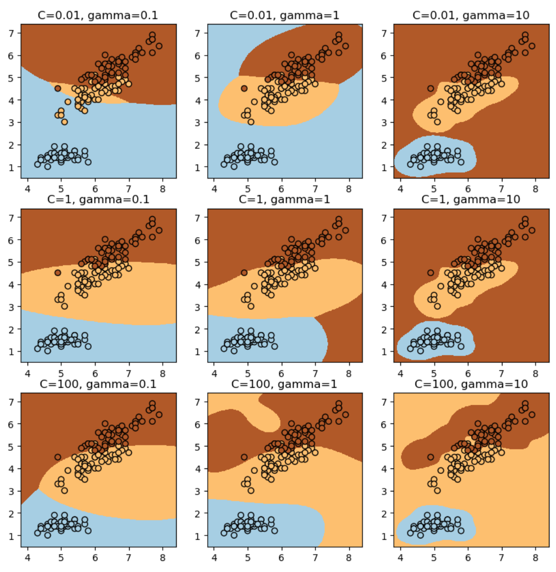


Figure 5 Recognition results using cSVM for different C and γ.

dataset, IRIS, which is a multivariate data set having three classes. We chose two variates as training data, as two variates are easy to visualize. Fig. 4 shows the recognition results optimized by DA, and Fig. 5 show those optimized using cSVM for comparison. Table 2 lists the accuracies obtained from DA for different parameter values.

Table 2 Accuracy of proposed method for different hyperparameters on IRIS dataset

No.	$\gamma = 0.1$	$\gamma = 1$	$\gamma = 10$
1	0.853	0.947	0.813
2	0.713	0.900	0.847
3	0.807	0.667	0.900

The colored areas in Fig. 4 and Fig. 5 distinguish the three classes, and their edges are the decision boundaries of SVM. In Fig. 4, when $\gamma = 0.1$, the decision boundaries are broad and underfit the data. When $\gamma = 10$, the decision boundaries fit the data better, but tend to overfit them. When $\gamma = 1$, qSVM separates the data well with high generalization performance.

The accuracy of our method is shown in Table 2. It can be seen that when the parameters are appropriate, our method has 94.7% accuracy ($\gamma = 1, \xi = 20$), which is means it can separate multiclass data successfully. Since the parameters were tuned simply, higher accuracy can be expected if more sophisticated parameter tuning is done.

5 Conclusions

We proposed a quantum SVM algorithm that solves multiclass classification problems and can be run on a quantum annealer. It based on the *one-versus-the-rest* and *one-versus-one* approaches. We use the minimum energy obtained from the quantum annealer to find the largest margin between classes. This enables us to assign uncertain answers to a certain class when using multiclass *one-versus-the-rest* and *one-versus-one* approaches. We evaluated our method on synthetic data and a well-known benchmark dataset, IRIS, using the Fujitsu Digital Annealer (DA), a quantum-inspired annealing machine. (As well, the algorithm can directly be used on a quantum annealer.) The results show that our method can classify multiclass data at a precision comparable to that of classical implementations. When the hyperparameter is set properly, our method can recognize data with high generalization performance.

In the future, we will add a constraint term to adjust the hyperparameter C. In a classical SVM, C is the penalty for misclassifying a data point. In qSVM, it is already included in the encoding formulation and is a fixed value. We can widen the range of applications of multiclass classification using qSVM by increasing the adjustment range of the hyperparameter.

References

[1] A.Steane, "Quantum computing." Reports on Progress in Physics 61.2 (1998): 117.

[2] M.W. Johnson et al. "Quantum annealing with manufactured spins." Nature 473.7346 (2011): 194-198.

[3] A. Lucas. "Ising formulations of many NP problems." Frontiers in Physics 2 (2014): 5.

[4] P.I. Bunyk et al. "Architectural considerations in the design of a superconducting quantum annealing processor." IEEE Transactions on Applied Superconductivity 24.4 (2014): 1-10.

[5] M. Sao et al. "Application of digital annealer for faster combinatorial optimization." Fujitsu Scientific & Technical Journal 55.2 (2019): 45-51.

[6] H. Neven et al. "Training a binary classifier with the quantum adiabatic algorithm." arXiv preprint arXiv:0811.0416 (2008).

[7] E. Aarts et al. "Simulated annealing and Boltzmann machines." (1988).

[8] H. Neven et al. "Training a large scale classifier with the quantum adiabatic algorithm." arXiv preprint arXiv:0912.0779 (2009).

[9] K.L. Pudenz and D.A. Lidar. "Quantum adiabatic machine learning." Quantum information processing 12.5 (2013): 2027-2070.

[10] R. Babbush et al. "Construction of non-convex polynomial loss functions for training a binary classifier with quantum annealing." arXiv preprint arXiv:1406.4203 (2014).

[11] S.H. Adachi and M.P. Henderson. "Application of quantum annealing to training of deep neural networks." arXiv preprint arXiv:1510.06356 (2015).

[12] D. Willsch et al. "Support vector machines on the D-Wave quantum annealer." Computer Physics Communications 248 (2020): 107006.

[13] P. Rebentrost et al. "Quantum support vector machine for big data classification." Physical Review Letters 113.13 (2014): 130503.

[14] C.L. Blake and C.J. Merz. UCI repository of machine learning databases, 1998. http://archive.ics.uci.edu/ml/datasets/Iris

[15] P.J.M. van Laarhoven and E.H.L. Aarts. "Simulated annealing." Simulated Annealing: Theory and Applications. Springer, Dordrecht, 1987. 7-15.

[16] E. Ising. "Beitrag zur theorie des ferromagnetismus." Zeitschrift fur Physik 31, 253 (1925).

[17] F. Barahona. "On the computational complexity of Ising spin glass models." J. Phys. A: Math. Gen. 15, 3241 (1982).

[18] F. Glover et al. "A Tutorial on formulating and using QUBO models." arXiv preprint arXiv:1811.11538 (2018).

[19] V. Vapnik. The nature of statistical learning theory. Springer Science & Business Media, 2013.

[20] W.H. Press et al. *Numerical recipes 3rd edition: The art of scientific computing.* Cambridge University Press, 2007.

[21] B. Schölkopf et al. *Learning with kernels: support vector machines, regularization, optimization, and beyond.* MIT Press, 2002.

[22] C.M. Bishop. *Pattern recognition and machine learning.* Springer, 2006.

[23] V. Vapnik. "Statistical learning theory." Wiley, New York (1998).

[24] Y. Lee et al. "Multicategory support vector machines, theory, and application to the classification of microarray data and satellite radiance data." Journal of Atmospheric and Oceanic Technology (2003).

[25] J.C. Platt et al. "Large margin DAGs for multiclass classification." Advances in Neural Information Processing Systems. 2000.

[26] F. Pedregosa et al. Scikit-learn: Machine Learning in Python, JMLR 12, pp. 2825-2830, 2011