End2End Quantum Learning

Mario Bifulco[1,*], Luca Roversi[1]

[1]*Università degli studi di Torino, Dipartimento di Informatica, Corso Svizzera 185 - 10149 Torino*

Abstract

This work presents a fully quantum approach to support vector machine (SVM) learning by integrating gate-based quantum kernel methods with quantum annealing-based optimization. We explore the construction of quantum kernels using various feature maps and qubit configurations, evaluating their suitability through Kernel-Target Alignment (KTA). The SVM dual problem is reformulated as a Quadratic Unconstrained Binary Optimization (QUBO) problem, enabling its solution via quantum annealers. Our experiments demonstrate that a high degree of alignment in the kernel and an appropriate regularization parameter lead to competitive performance, with the best model achieving an F1-score of 90%. These results highlight the feasibility of an end-to-end quantum learning pipeline and the potential of hybrid quantum architectures in quantum high-performance computing (QHPC) contexts.

Keywords

Quantum Machine Learning, Quantum Support Vector Machine, Quantum High Performance Computing, Quantum Gate, Quantum Annealing

1. Introduction

Quantum computing ultimately aims to expand the boundaries of what is currently considered efficiently computable. Among the various fields where such advancements appear promising is machine learning.

When discussing quantum computing, it is essential to specify the computational paradigm being considered. *Gate-based quantum computing* refers to a universal quantum computing model, typically implemented using superconducting circuits, where quantum circuits are programmed analogously to how logic gates define classical circuits. *Annealing-based quantum computing*, on the other hand, refers to a non-universal subset of adiabatic quantum computing, primarily designed for solving optimization problems.

In the context of quantum machine learning, support vector machines (SVMs) have been studied under both computational paradigms[1, 2]. Within SVMs, we can distinguish two principal components that contribute to model training: 1. The *kernel method*, which enables the mapping of examples into a higher-dimensional feature space, thereby increasing the likelihood of linear separability, as per Cover's theorem[3]; 2. The *optimization problem*, which identifies the model parameters used for inference on new data instances.

In the literature, the term *Quantum Support Vector Machine (QSVM)* is used with two distinct meanings. In the quantum annealing framework, QSVM refers to the use of classical kernels combined with a reformulation of the optimization problem to be solved via quantum annealing. In the gate-based approach, by contrast, quantum computing is employed to compute the kernels, while the optimization is handled by classical processors.

This work aims to unify the aforementioned approaches to construct a fully quantum learning pipeline for support vector machines using a kernel based on quantum gates and a model optimisation process based on quantum annealing. Moreover, the use of different types of QPUs enables experimentation within the domain of Quantum High Performance Computing (QHPC). In this domain, traditional CPU

and GPU systems can collaborate with various QPUs—regardless of their underlying architecture—with the goal of addressing computationally intensive problems.

2. A short recap about gate-based generated Quantum Kernels

This section briefly recalls the main steps to construct a gate-based quantum circuit that implements a quantum kernel.

Let V be a set of input vectors.

1. Each $\vec{x} \in V$ is interpreted as a quantum state $|\phi(\vec{x})\rangle$ by means of a quantum circuit $\phi(\vec{x})$;

2. For each pair $(\vec{x}_i, \vec{x}_j)$, the quantum circuit $\phi(\vec{x}_i); (\phi(\vec{x}_j))^{-1}$ is constructed. It pipelines $\phi(\vec{x}_i)$, building the state $|\phi(\vec{x}_i)\rangle$, with the inverse $(\phi(\vec{x}_j))^{-1}$ of $\phi(\vec{x}_j)$, building the state $|\phi(\vec{x}_j)\rangle$;

3. The overlap between the states $|\phi(\vec{x}_i)\rangle$ and $|\phi(\vec{x}_j)\rangle$ is estimated by measuring the probability of the resulting state collapsing to the all-zero state, i.e., computing $|\langle\phi(\vec{x}_i)|\phi(\vec{x}_j)\rangle|^2$.

This procedure leads to define a kernel matrix K such that each entry K_{ij} corresponds to the similarity between $\vec{x}_i$ and $\vec{x}_j$ as inferred from their representation as quantum states $|\phi(\vec{x}_i)\rangle$ and $|\phi(\vec{x}_j)\rangle$. Intuitively, the advantage of constructing K as above lies in the ability of quantum kernels to exploit qubit entanglement and state superposition to compress the number of qubits required. Adding a single qubit to the feature map ϕ doubles the dimensionality of the space to which the input x_i is mapped.

3. Quantum Annealing Support Vector Machine

Annealing-based quantum computing naturally addresses quadratic optimization problems, particularly those formulated as QUBO (Quadratic Unconstrained Binary Optimization) instances. The dual formulation of the support vector machine (SVM) problem [4] leads to a multivariate quadratic polynomial, which serves as the objective function to be optimized. By incorporating constraints into the objective via penalty terms and converting continuous variables into their binary expansion, the problem can be reformulated as a QUBO model.

Constraint incorporation is typically achieved through Lagrangian relaxation [5]. However, the binarization process is less straightforward due to the continuous nature of the optimization variables, which lie in $\mathbb{R}$. On one hand, integer variables can be automatically encoded in binary format, provided that an upper bound is defined for each variable. This additional hyperparameter is crucial, as omitting it would result in each variable being represented with 64 bits by default, leading to problems that exceed the capacity of current quantum annealers. On the other hand, the binarization of real-valued variables incurs significantly more overhead.

Empirically, we found that treating all optimization variables as integers yields a degradation in model performance of less than one percentage point. This trade-off led us to favor an approximated yet more hardware-efficient formulation during model generation.

An additional hyperparameter of practical importance is a *regularization parameter C*. It controls the trade-off between maximizing the classification margin and minimizing classification errors. In this context, the margin refers to the distance between the separating hyperplane and the closest data points from each class.

Higher values of C emphasize correct classification of training examples, whereas lower values allow for a softer margin with more tolerance to misclassified points.

4. Results

Both the design of a kernel and the optimization of the SVM problem require the selection of hyperparameters prior to training. Section 4.1 presents our investigation into the parametrization of the
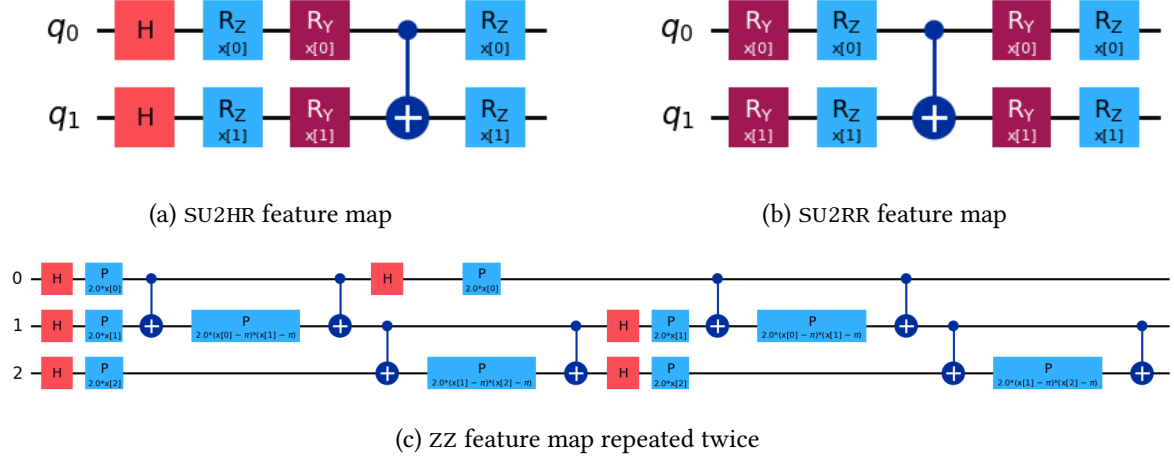
(a) SU2HR feature map   (b) SU2RR feature map

(c) ZZ feature map repeated twice

Figure 1: Examples of feature maps on two qubits.

contruction of a quantum kernel in the context of gate-based architectures, with respect to a chosen dataset. Section 4.2 reports on a set of experiments aimed at selecting the parameter C in the context of quantum-anneal based architectures, assuming no kernel is employed during training.

Once the optimal hyperparameters for the individual components of the problem have been indipendently identified, Section 4.3 presents the results of the end-to-end learning pipeline for fully quantum SVMs—namely, those employing a kernel generated via quantum gates and optimization performed through quantum annealing.

4.1. Finding a good Quantum Kernel

To identify the optimal hyperparameters to find a quantum kernel, we evaluated several values for the following parameters:

Number of qubits available: Determines the upper limit of encodable features. When the number of qubits was insufficient, we applied Principal Component Analysis (PCA) to project the data into an n-dimensional subspace capturing the most relevant variance.

Feature map: The quantum circuit responsible for encoding classical data into quantum states. These circuits determine the geometry of the induced Hilbert space and affect model expressivity. Figure 1a, Figure 1b and Figure 1c are three examples of possible feature maps.

Repetitions of the feature map: Defines how many times the feature map circuit is applied in sequence. The reason repetitions of the same feature map may be required is that, in certain cases, a single application is not sufficient to capture the desired non-linear transformation $\phi(\vec{x})$. This layered construction is analogous to deep architectures in classical machine learning, where increasing depth enhances representational capacity. Figure 1c shows a `ZZ_feature_map` repeated twice.

To evaluate the quality of the kernel derived from the various feature maps, we employed the *Kernel-Target Alignment* (KTA) strategy [6]. This metric allows for estimating the effectiveness of a given kernel on a reference dataset without the need to train a machine learning model.

Formally, given a kernel matrix $K \in \mathbb{R}^{n \times n}$ computed over a set of n examples, and a label vector $\mathbf{y} \in \{-1, 1\}^n$, the KTA is defined as:

$$\mathrm{KTA}(K, \mathbf{y}) = \frac{\mathbf{y}^\top K \mathbf{y}}{\|K\|_F \cdot n}$$

Table 1
Top three kernel configurations.

# Qubits	Feature Map	Repetitions	Alignment (KTA)
30	SU2HR	1	98.650%
8	ZMAP	2	93.558%
16	SU2RR	1	91.203%

where $\|K\|_F$ denotes the Frobenius norm of the kernel matrix. This normalized inner product quantifies the alignment between the kernel matrix and the ideal target kernel implied by the labels. A higher value of KTA indicates a stronger alignment and, consequently, a potentially more suitable kernel for classification tasks.

The top three configurations for the quantum kernel are reported in Table 1. All proposed configurations exhibit a high degree of alignment. In this study, no single hyperparameter emerged as universally optimal across configurations. This is evidenced by the presence of both circuits with a large number of qubits and those with fewer qubits, deep circuits as well as shallow ones, and the use of three distinct feature maps.

4.2. Selecting regularization parameter for Quantum Annealing SVM

In our experiments, we explored a range of values for C by selecting powers of two, from $2^2 - 1$ to $2^{12} - 1$. This choice is motivated by the way optimization variables are encoded in our quantum annealing formulation of SVMs. Specifically, each continuous optimization variable is discretized and represented using $\log_2(C)$ binary variables. By choosing C as $2^b - 1$, where b is the number of bits used, we ensure that the integer values these binary variables can represent range from 0 to C, thereby fully utilizing all possible bit combinations. This approach minimizes the number of unused configurations in the binary encoding and ensures a compact and efficient use of the available qubits, which is crucial given the hardware limitations of current quantum annealers.

We observed the best performance using regularization parameter values of $C = 7, 63, 255$. Nevertheless, the differences in performance across the various configurations are minimal, suggesting that—in the application context we investigated—the examples are likely to be linearly separable with not excessive rate.

4.3. Fully Quantum SVM

By combining quantum kernels with optimization via quantum annealing, we designed a QSVM pipeline consisting of the following steps: 1. Computing the quantum kernel matrix K using a gate-based quantum device, which encodes pairwise similarities between training examples in an implicit feature space; 2. Formulating an optimization problem where K is used to construct the objective function; 3. Solving this problem using quantum annealing to determine the optimal model parameters; 4. Using the obtained parameters to perform inference on new examples. These models are designed to solve supervised classification tasks by learning a hyperplane that separates samples belonging to the positive class from those of the negative class.

The best-performing model was based on the kernel with the highest alignment (see Table 1), specifically the kernel constructed using 30 qubits and, among the candidate values introduced in Section 4.2, a regularization parameter of $C = 255$. In this case, we achieved an `F1-score` of 90%. We also observed that the negative class exhibited higher `recall` but lower `precision`, whereas the opposite was true for the positive class. This suggests that the trained model suffers from classification issues that lead to the generation of false negatives. False negatives were found across all models, indicating that this is a systematic issue. As usual, a more effective preprocessing phase would likely result in a uniform performance improvement across all configurations.

The model with the smallest number of qubits yielded the poorest results. Using only 8 qubits to encode the input features and setting $C = 63$, the resulting `F1-score` was 52%, making this model comparable to random guessing. This behavior may be attributed to an insufficient number of features used to represent the data: reducing from 30 to 8 qubits likely caused a significant loss of information, leading to examples becoming overly similar and thus harder to distinguish.

5. Conclusion

In this study, we have explored the feasibility and effectiveness of constructing an end-to-end quantum learning pipeline for support vector machines. By integrating gate-based quantum kernel methods with quantum annealing-based optimization, we demonstrated that it is possible to train a fully quantum SVM model on real-world datasets.

Our results show that the choice of quantum kernel—particularly the number of qubits and the selected feature map—significantly influences classification performance. Furthermore, we observed that the approximation involved in representing real-valued optimization variables with discrete binary variables in the QUBO formulation does not lead to substantial performance degradation.

The best-performing configuration, employing 30 qubits and a regularization parameter $C = 255$, achieved an F1-score of 90%. However, the presence of systematic false negatives suggests that further improvements could be achieved with a more refined data preprocessing pipeline. While incorporating domain knowledge about the origin and nature of the training and test datasets could also be beneficial, we believe that the most effective improvements are likely to come from enhanced preprocessing strategies or the use of larger and more representative datasets. At present, however, increasing the dataset size is not feasible due to the computational cost associated with evaluating the quantum kernel, which scales unfavorably with the number of samples.

This work is a first step towards future investigations into scalable quantum machine learning pipelines and highlights the role of quantum high-performance computing in facilitating hybrid computational workflows across diverse quantum architectures.

References

[1] D. Willsch, M. Willsch, H. Raedt, K. Michielsen, Support vector machines on the d-wave quantum annealer, Computer Physics Communications 248 (2019) 107006. doi:10.1016/j.cpc.2019.107006.

[2] V. Havlíček, A. D. Córcoles, K. Temme, A. W. Harrow, A. Kandala, J. M. Chow, J. M. Gambetta, Supervised learning with quantum-enhanced feature spaces, Nature 567 (2019) 209–212. URL: http://dx.doi.org/10.1038/s41586-019-0980-2. doi:10.1038/s41586-019-0980-2.

[3] T. M. Cover, Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition, IEEE Transactions on Electronic Computers EC-14 (1965) 326–334. doi:10.1109/PGEC.1965.264137.

[4] C. Cortes, V. N. Vapnik, Support-vector networks, Machine Learning 20 (1995) 273–297. URL: https://api.semanticscholar.org/CorpusID:52874011.

[5] C. Lemaréchal, Lagrangian Relaxation, Springer Berlin Heidelberg, Berlin, Heidelberg, 2001, pp. 112–156. URL: https://doi.org/10.1007/3-540-45586-8_4. doi:10.1007/3-540-45586-8_4.

[6] N. Cristianini, J. Shawe-Taylor, A. Elisseeff, J. S. Kandola, On kernel-target alignment, in: Neural Information Processing Systems, 2001. URL: https://api.semanticscholar.org/CorpusID:17466014.