Quantum Bootstrap Aggregation

David Windridge and Rajagopal Nagarajan

Department of Computer Science, School of Science and Technology,
Middlesex University, London NW4 4BT, UK
email: {d.windridge@mdx.ac.uk, r.nagarajan@mdx.ac.uk}

Abstract. We set out a strategy for quantizing attribute bootstrap aggregation to enable variance-resilient quantum machine learning. To do so, we utilise the linear decomposability of decision boundary parameters in the Rebentrost *et al.* Support Vector Machine to guarantee that stochastic measurement of the output quantum state will give rise to an ensemble decision without destroying the superposition over projective feature subsets induced within the chosen SVM implementation. We achieve a linear performance advantage, $O(d)$, in addition to the existing $O(log(n))$ advantages of quantization as applied to Support Vector Machines. The approach extends to any form of quantum learning giving rise to linear decision boundaries.

1 Introduction

Quantum Machine Learning is a recent area of research initiated by the demonstration of a quantum Support Vector Machine (SVM) by Rebentrost, Mohseni & Lloyd [1] and the k-means algorithm by Aïmeur, Brassard & Gambs [2] (cf also [3–8]). The development of the quantum SVM can be regarded as particularly significant in that the classical SVM constitutes perhaps the exemplar instance of a *supervised binary classifier*, i.e. an entity capable of learning an optimal discriminative decision hyperplane from labeled vectors $\{(\boldsymbol{x}, y) \mid \boldsymbol{x} \in \tilde{X},\ y \in \{-1, +1\}\}$ existing within a feature space.

Bootstrap Aggregation ('Bagging') is a well established method within stochastic machine learning for removing variance from classifiers via the production of bootstrap ensembles to refine the final decision accuracy. It shall be the argument of this paper that this decision ensemble can be equivalently represented via a quantum superposition, such that the final decision can be straightforwardly obtained via quantum measurement. Moreover, we shall demonstrate that this is necessarily more economic in both execution time and the total number of logic gates required in comparison to classical (and even parallelized quantum) SVM implementations.

2 Methodological Background

The Classical SVM The standard SVM [9] seeks to maximize the margin (*i.e.*, the distance of the decision hyperplane to the nearest data point), subject to a

constraint on the classification accuracy of the labelling induced by the hyperplane's delineation of a general decision boundary. In its primal form, the soft margin SVM optimization takes the form of a Lagrange optimization problem:

$$\arg\min_{(\boldsymbol{w},b)}\left\{\frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{M}\xi_i\right\} \quad \text{subject to:} \quad \forall_i\, y_i(\boldsymbol{w}\cdot\boldsymbol{x_i} - b) \geq 1-\xi_i,\ \ \xi_i \geq 0$$

where $(\boldsymbol{x_i}, y_i)$ $i = 1 \ldots M$ are the training vectors/labels, $y_i \in \{-1, +1\}$, $\boldsymbol{w}$ is the weight orientation vector of the decision hyperplane, and b is its bias offset. (The margin is inversely proportional to $\|\boldsymbol{w}\|$). The ξ_i are slack variables that give rise to the soft margin with sensitivity controlled by hyper-parameter C.

In the dual form [9], the slack parameters disappear such that the problem is solved in terms of the Karush–Kuhn–Tucker (KKT) multipliers α_i:

$$\arg\max_{(\alpha_i)}\sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j(\boldsymbol{x}_i^T\boldsymbol{x}_j) \quad \text{subject to:} \sum\alpha_i y_i = 0 :\ \forall_i\, 0 \leq \alpha_i \leq C$$

The problem is one of quadratic programming. As the optimization proceeds, only a sparse set of the α_s's retain non-zero values. These denote the support vectors defining the decision hyperplane. This sparsity (i.e. the low parametric complexity of the decision boundary with respect to the training data) gives the SVM substantial resilience to over-fitting (and thus reduces classifier variance).

Notably, the term $(\boldsymbol{x}_i^T\boldsymbol{x}_j)$ in the above (equating to the training vector Gram matrix) may be freely replaced by any kernel function $K(\boldsymbol{x}_i, \boldsymbol{x}_j)$ that satisfies Mercer's condition (i.e. positive semi-definiteness). This vastly extends the utility of the SVM by enabling the mapping of the input decision space into a large variety of alternative Hilbert spaces of potentially infinite dimensionality (thus guaranteeing linear separability). The decision boundary in the input space may thus undergo significant morphology variation while crucially retaining the low parametric support-vector characterization of the decision boundary within the Mercer embedding space (the space denoted $\boldsymbol{\phi}(\boldsymbol{x})$ for which $K(\boldsymbol{x}_i, \boldsymbol{x}_j) \equiv \boldsymbol{\phi}(\boldsymbol{x_i})^T(\boldsymbol{\phi}(\boldsymbol{x_j}))$. Critically, at no stage are we required to compute $\boldsymbol{\phi}(\boldsymbol{x_i})$). The KKT conditions guarantee the existence of $\boldsymbol{\phi}$, but the kernel itself may be calculated based on any similarity function that gives rise to a kernel matrix obeying Mercer's condition.

The Quantum SVM The quantum SVM implementation proposed by Rebentrost, Mohseni and Lloyd [1] uses a least square reimplementation of the classic kernelized SVM so as to implicate the efficient quantum matrix inversion of Harrow, Hassidim & Lloyd [10]. The problem to be solved now becomes:

$$F\begin{pmatrix} b \\ \boldsymbol{\alpha}\end{pmatrix} \doteq \begin{pmatrix} 0 & \mathbf{1}^T \\ \mathbf{1} & K + \gamma^{-1}I\end{pmatrix}\begin{pmatrix} b \\ \boldsymbol{\alpha}\end{pmatrix} = \begin{pmatrix} 0 \\ \boldsymbol{y}\end{pmatrix} \qquad \mathbf{1}^T \equiv (1, 1, 1 \ldots)^T \tag{1}$$

where K is the kernel matrix (i.e. a permissible generalization of the Gram matrix satisfying Mercer's condition), γ^{-1} is the trade-off parameter between the SVM

optimization and accuracy. Training object classifications are denoted by the vector $\boldsymbol{y} \in ([-1,1]^M)^T$ for the M training objects order-correlated with the kernel matrix K (training object vectors $\boldsymbol{x_k}$ are represented in their own basis). Finally, $\boldsymbol{\alpha}$ and b (the object of the optimization) are respectively the weight and bias offset parameters of the decision hyperplane within the Mercer embedding space induced by the kernel (though note that here the alpha represent distances from the margin).

Consequently, quantum matrix inversion of F solves for the SVM parameters $\boldsymbol{\alpha}$, b, producing the solution state:

$$|\,\boldsymbol{\alpha}, \beta\rangle = \frac{1}{b^2 + \sum_{k=1}^{M} \alpha_k^2} \left(b\,|\,0\rangle + \sum_{k=1}^{M} \alpha_k\,|\,k\rangle \right) \tag{2}$$

Utilization of these parameters for classification of novel data requires the implementation of a *query oracle* implicating all of the labeled data:

$$|\,\tilde{u}\rangle = \frac{1}{\left(b^2 + \sum_{k=1}^{M} \alpha_k^2 |\boldsymbol{x_k}|^2\right)^{\frac{1}{2}}} \left(b\,|\,0\rangle\,|\,0\rangle + \sum_{k=1}^{M} |\boldsymbol{x_k}|\,\alpha_k\;\;|\,k\rangle\;\;|\,\boldsymbol{x_k}\rangle \right) \tag{3}$$

and also the query state:

$$|\,\tilde{x}\rangle = \frac{1}{M|\boldsymbol{x}|^2 + 1} \left(|\,0\rangle\,|\,0\rangle + \sum_{k=1}^{M} |\boldsymbol{x_k}|\,|\,k\rangle\,|\,\boldsymbol{x_k}\rangle \right) \tag{4}$$

($|\,k\rangle$ is thus an index state over training vectors)

The classification is then carried out as the inner product of the two states, i.e. by performing a swap test and allocating class labels on the basis of the inner product probability being greater or less than $\frac{1}{2}$ (the swap test is performed via the use of an ancilla to construct the state $\frac{1}{\sqrt{2}}(|\,0\rangle\,|\,\tilde{u}\rangle + |\,1\rangle\,|\,\tilde{x}\rangle)$ which is then measured in the basis $\frac{1}{\sqrt{2}}(|\,0\rangle - |\,1\rangle)$).

Bootstrap Aggregation and Attribute Bootstrap Aggregation Standard bootstrap aggregation is an effective approach for stabilizing unstable classifiers such as decision trees and neural networks [11]. It consists in randomly sampling, with replacement, d groups from the total set of training samples M, training the resulting classifiers and combining the output either via decision fusion (such as majority voting) or averaging in the case of regression-like classifiers. Each set can be expected to have $M(1 - e^{-m/M})$ unique training vectors on average for draw size m.

It may be shown, via bias/variance analysis [12], that bagging can be considered primarily as a method for reducing variance with respect to training-set permutation/sampling. It is therefore be employed predominantly on low-bias classifiers, since classifiers must necessarily trade-off variance against bias in their design (bias is in this sense the expected discrepancy from Bayes optimality).

Bootstrap aggregation is therefore notably less effective for an idealized SVM (or in completely linearly separable problems) owing to the intrinsic variance-resilience implied in the definition of the maximum margin SVM classifier in terms of the support objects -i.e. those objects for which, in which the dual form of SVM optimization problem, the Lagrangian multipliers are non-zero [9]. Typically, these are highly sparse, and therefore training set sub-sample permutation has no effect unless explicitly excluding these objects.

There are, however, techniques for artificially adjusting bias/variance within SVMs. The parameter γ^{-1} above dictates the trade off between data fitting and maximization of the the margin. Favoring the former should therefore reduce the bias. An alternative strategy for bias reduction applicable to the quantum SVM applies when considering the polynomial kernel: $K(\boldsymbol{x}_j, \boldsymbol{x}_k) = (\boldsymbol{x}_j \cdot \boldsymbol{x}_k)^D \equiv \boldsymbol{\phi}(\boldsymbol{x_j}) \cdot \boldsymbol{\phi}(\boldsymbol{x_k})$. Here, D can control bias via the relationship between polynomial degree and functional localization.

The most generally effective strategy for bootstrap aggregation, however, utilizes feature subspaces (referred to as either 'attribute bagging' or the 'random subspace method (RSM)'). Here, it is the selection of the features for classification that constitutes the bootstrap set. Subspace remapping is, however, also a natural quantum operation (implemented by projectors). Thus, if the set of training vectors $\boldsymbol{x_k}$ are represented within an orthonormal Hilbert basis, we are implicitly concerned with the projectors $\{|\mathcal{P}_1\rangle\langle\mathcal{P}_1|, |\mathcal{P}_2\rangle\langle\mathcal{P}_2|, \ldots\}$ in carrying out subspace selection within a quantum context, where the $\mathcal{P}_i$ are Hilbert space vectors with spans corresponding to the feature subsets P_i. We will thus utilize this approach in the following to define distinct set of classifiers constituting the ensemble.

Critically, from the point of view of efficiently quantizing attribute bagging, we do not require individual classifier decisions to be identified as such within the final ensemble decision: in effect, the collective classifier acts as a single composite classifier. In quantum terms, this implies that classifiers are able to exist as a superposition without individual measurement prior to the final decision output.

3 Proposed Attribute Bootstrap Aggregation Method

The standard attribute bootstrap aggregation algorithm proceeds as follows: for M training objects with N features, we individually train S classifiers on the respective feature sets $d_s = \{d_s \subset N \mid |d_s| < N\}$, either with or without replacement. To classify test objects, we combine the S classifier outputs by e.g. majority vote or summation over posterior probabilities.

In the following quantum implementation of attribute bootstrap aggregation our objective will thus be to set up a quantum superposition of classifier decision hyperplanes associated with each attribute selection. This will give rise to an ensemble sum over decisions for which it may be demonstrated that a collective measurement is sufficient for ensemble classification (we are thus implicitly opting for the 'summation over posterior probability' form of attribute bootstrap aggregation.)

We initialize our approach by selecting a total of S random selections, p_s, from N features, either with or without replacement, such that $p_s \in \{0,1\}^N$. p_s is hence a characteristic function indexing basis states $|\, k\rangle$. For each projector indexed by s we can thus train an SVM : $F_s^{-1}\left(\boldsymbol{\phi}(P_s\boldsymbol{x_j})^T\boldsymbol{\phi}(P_s\boldsymbol{x_k}), \boldsymbol{y}\right)$ where P_s is the projection matrix corresponding to p_s (i.e P_s is the diagonal matrix having the binary values of p_s on its leading diagonal; $P_s\{i,j\} = 0$ if $i \neq j,\ P_s\{i,i\} = p_s(i)\ \ \forall i,j \in \{1,2,\ldots,N\}$).

In quantum terms, this means that we can construct a solution state superposition over training vector basis states in order to construct the query oracle (Note that the bootstrap sets occupy the full training vector Hilbert space, irrespective of the differential subspace dimensionalities, so that we are free to form a superposition over projected vectors; thus we do not consider explicit sums over projectors, $\sum_{s=1}^{S} |\, \mathcal{P}_s\rangle\langle\mathcal{P}_s \,|$, or density operators, such as would be implicit in a statistical ensemble approach).

Hence, (setting ϕ to the identity for convenience) we can construct the quantum state:

$$|\, \tilde{u_B}\rangle = \frac{1}{S^{\frac{1}{2}}}\sum_{s=1}^{S} |\, \boldsymbol{\alpha}_s, \beta_s\rangle \equiv \mathcal{N}\sum_{s=1}^{S}\left(b_s\, |\, 0\rangle\, |\, 0\rangle + \sum_{k=1}^{M} |P_s\boldsymbol{x_k}|\alpha_{(k,s)}\, |\, k\rangle\, |\, \mathcal{P}_s\rangle\langle\mathcal{P}_s|\, |\, \boldsymbol{x_k}\rangle\right)$$

$$\text{where}\quad \mathcal{N} = \frac{1}{(\sum_{s=1}^{S} b_s^2 + \sum_{k=1}^{M}\alpha_{(k,s)}^2 |P_s\boldsymbol{x_k}|^2)^{\frac{1}{2}}}$$

(implicitly obtaining the quantum speed up for each SVM implementation within the superposition).

A key point to note is that the set of $\alpha_{(k,s)}$'s for some arbitrary S acts over all of the M training vectors in order to define the decision hyperplane (in contrast to the Lagrange dual SVM formulation), each defining a distance from the optimal margin. As such, the set $\{\alpha_{(k,s)}\}$ defines a unique subspace of dimensionality $|p_s| - 1$ within the subspace subtended by the $\{P_s\boldsymbol{x_k}\}$ (i.e the decision hyperplane), where $|p_s|$ here indicates the Hamming weight. However, the *same* $\{\alpha_{(k,s)}\}$ also define a unique subspace of dimensionality $N-1$ within $\boldsymbol{X}$, namely the direct product of the decision hyperplane with the null space of P_s. We may therefore, (in constructing the solution state only, and absolutely not the SVM matrix inversion) treat $|\, \mathcal{P}_s\rangle\langle\mathcal{P}_s|\, |\, \boldsymbol{x_k}\rangle$ and $|\, \boldsymbol{x_k}\rangle$ equivalently with regard to the $\{\alpha_{(k,s)}\}$ (though not the $|\boldsymbol{x_k}|$).

If we thus set $\alpha'_k = \left(\sum_{s=1}^{S}\alpha_{(k,s)}\right)$, $b'_k = \left(\sum_{s=1}^{S} b_s\right)$ and $|\boldsymbol{x'_k}| = |P_s\boldsymbol{x_k}|$, then it may be seen (by moving the summation inside the bracket and gathering terms) that the following equivalences and equalities hold:

$$|\, \tilde{u_B}\rangle = \frac{1}{S^{\frac{1}{2}}}\sum_{s=1}^{S} |\, \boldsymbol{\alpha}_s, \beta_s\rangle = \mathcal{N}\left(\sum_{s=1}^{S} b_s\, |\, 0\rangle\, |\, 0\rangle + \sum_{k=1}^{M}\sum_{s=1}^{S} |P_s\boldsymbol{x_k}|\alpha_{(k,s)}\, |\, k\rangle\, |\, \mathcal{P}_s\rangle\langle\mathcal{P}_s|\, |\, \boldsymbol{x_k}\rangle\right)$$

$$\equiv \mathcal{N}\left(\sum_{s=1}^{S} b_s\, |\, 0\rangle\, |\, 0\rangle + \sum_{k=1}^{M}\sum_{s=1}^{S} |\boldsymbol{x'_k}|\alpha_{(k,s)}\, |\, k\rangle\, |\, \boldsymbol{x_k}\rangle\right)$$

$$\equiv \text{normalisation const.} \times \left(\mathrm{b}'\, |\, 0\rangle\, |\, 0\rangle + \sum_{\mathrm{k}=1}^{\mathrm{M}} |\boldsymbol{x'_k}|\, \alpha'_{\mathrm{k}}\, |\, \mathrm{k}\rangle\, |\, \boldsymbol{x_k}\rangle\right)$$

It may thus be seen that the modified basis is identical to the previous basis, and the ensemble training-data oracle has the same overall form as the original training oracle. Critically, this means that the training oracle is thus represented in the same basis as the query state $\tilde{x}$; i.e the sum over projectors $|\mathcal{P}_s\rangle\langle\mathcal{P}_s|$ has not altered the representation of the final decision state within the training vector basis, a result that comes about because of the linear separability of training weights in the least squares SVM implementation.

The solution is therefore read off as before (i.e by using an ancilla to construct the state $\frac{1}{\sqrt{2}}(|0\rangle\,|\tilde{u_B}\rangle + |1\rangle\,|\tilde{x}\rangle)$ measured in the basis $\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$).

Individual classification decisions are thus no longer resolvable in the swap measurement; only the ensemble decision is measurable. Importantly, no new logic gates or oracle basis is implicated in this construction. Consequently, bootstrap aggregation is "free" within the Rebentrost, Mohseni and Lloyd framework.

4 Conclusion

We have demonstrated that it is possible to implement quantum bootstrap aggregation, specifically quantum attribute bootstrap aggregation, without penalty in quantum machine learning scenarios, using as an exemplar the Rebentrost, Mohseni and Lloyd SVM model. Thus, we can harness the stabilizing characteristics of bagging without requiring either additional logical gates or computation time. To do so, we exploit quantum superposition in such a way as to guarantee that stochastic measurement of the output state will give rise to an aggregate (i.e. ensemble) decision without destroying the superposition over feature subsets induced within the SVM implementation. This is enabled by the linear decomposability of decision boundary parameters within the Kernel-induced Mercer embedding space.

Acknowledgment

The first author would like to acknowledge financial support from the Horizon 2020 European Research project DREAMS4CARS (no. 731593). The second author is partially supported by EU ICT COST Action IC1405 "Reversible Computation—Extending Horizons of Computing".

References

1. P. Rebentrost, M. Mohseni, S. Lloyd, Quantum support vector machine for big data classification, Physical Review Letters 113 (130501).
2. E. Aïmeur, G. Brassard, S. Gambs, Quantum speed-up for unsupervised learning, Machine Learning 90 (2) (2013) 261–287.
3. M. Altaisky, N. Zolnikova, N. Kaputkina, V. Krylov, Y. E. Lozovik, N. S. Dattani, Towards a feasible implementation of quantum neural networks using quantum dots, arXiv preprint arXiv:1503.05125.

4. S. Lloyd, M. Mohseni, P. Rebentrost, Quantum principal component analysis, Nature Physics 10 (9) (2014) 631–633.
5. J. Barry, D. T. Barry, S. Aaronson, Quantum partially observable markov decision processes, Physical Review A 90 (3) (2014) 032311.
6. S. Lu, S. L. Braunstein, Quantum decision tree classifier, Quantum information processing 13 (3) (2014) 757–770.
7. R. R. Tucci, Quantum circuit for discovering from data the structure of classical bayesian networks, arXiv preprint arXiv:1404.0055.
8. N. Wiebe, A. Kapoor, K. Svore, Quantum algorithms for nearest-neighbor methods for supervised and unsupervised learning, arXiv preprint arXiv:1401.2142.
9. C. Cortes, V. Vapnik, Support-vector networks, Machine Learning 20 (3) (1995) 273–297. doi:10.1007/BF00994018.
10. A. W. Harrow, A. Hassidim, S. Lloyd, Quantum Algorithm for Linear Systems of Equations, Physical Review Letters 103 (15) (2009) 150502. arXiv:0811.3171.
11. L. Breiman, Bagging predictors, Machine Learning 24 (2) (1996) 123–140.
12. G. Valentini, T. G. Dietterich, Low bias bagged support vector machines, in: Int. Conf. on Machine Learning, ICML-2003, Morgan Kaufmann, 2003, pp. 752–759.