SUPPLEMENTARY INFORMATION

doi:10.1038/nature24047

ABSTRACT

We provide additional details in support of "On a Higgs optimization problem with quantum annealing"

1 The quantum annealer approach to the Higgs optimization problem

Our problem, toward which we apply quantum annealing for machine learning (QAML), is that of constructing a binary classifier that can detect the "signal" of the decay of a Higgs boson into two photons in a "background" of noise from other Standard Model processes. The classifiers are trained on a set of simulated collision events (synthetic data sets) where the signal sample contains events with a Higgs boson and the background sample contains a cocktail of background physics processes that mimic Higgs events. The classification is achieved by exploiting deep correlations in various physical properties of the signal and background events. Classifiers such as boosted decision trees [e.g., XGBoost (XGB)], or deep neural networks (DNN) have seen great success in many other contexts, from speech and image recognition,[1] to marketing, finance and manufacturing.[2] In the high energy physics context there are challenges and limitations of these techniques often related to the level of agreement between the synthetic and observed data. Supervised learning requires an accurately labeled training data set, and the simulation procedure requires calculations of the matrix elements of the physics processes of interest,[3] modeling of the hadronization of colored particles[?] and simulation of the interaction of the final state particles with the detector.[4] The complexity of the end-to-end simulation operation is encapsulated in the uncertainties associated with the level of agreement with the observations.

The binary classifier proposed and studied in this work, is trained with a "quantum annealing for machine learning" (QAML) algorithm[5,6] and takes the form of a linear neural network (LNN) that relies on explicitly linearized correlations. This reduces sensitivity to errors in the model of the detector, and due to the binary weights it guards against overtraining. In this model it is simple to control and correct the correlations between the kinematical observables in the Monte Carlo simulations. Additionally the model provides a straightforward interpretation of the criteria used to classify the events. This comes at the price of a provably NP-hard training problem, with a training time that grows exponentially with the number of variables. This is a price sometimes worth paying in return for robustness in the presence of label noise, a fact that has become increasingly recognized in the machine learning community.[7–9]

Heuristic optimization techniques such as classical simulated annealing (SA)[10,11] and quantum annealing (QA)[12,13] may reduce the training time sufficiently to solve problems of practical interest with this linear model. QA and the closely related quantum adiabatic algorithm,[14] hold the potential for significant improvements in performance over classical techniques, though the delineation of the improvement boundary remains an active area of research.[15] Here we use both QA and SA to train a classifier and examine its performance compared to traditional methods.

To implement QA we use a programmable quantum annealer[16] built by D-Wave Systems Inc.,[17,18] the D-Wave Two X (DW) model housed at the University of Southern California's Information Sciences Institute, comprising 1098 superconducting flux qubits. Such QA devices have been employed to study, e.g., graph isomorphism,[19] Bayesian network structure,[20] operational planning,[21] DNNs,[22] quantum Boltzmann machines,[23–25] and tree cover detection in aerial imagery.[26] Both the quantumness[27–31] and speedup[32–35] in these devices are intensely scrutinized topics of ongoing research.

2 DNN and XGB optimization procedure

We benchmark the performance of QAML against DNN and XGB.

We train a DNN using Keras[36] with the Theano backend,[37] a standard tool in deep learning and increasingly popular in high energy physics. Our network has two fully connected hidden layers with 1000 nodes each. The model is optimized using the Adam algorithm[38] with a learning rate of 0.001 and a mini-batch size of 10. We find that network performance is not affected by small changes in the number of nodes or the initial guesses for the weights. The model hyperparameters, regularization terms, and optimization parameters for our deep neural net are selected using the *Spearmint* Bayesian optimization software.[39,40] Early stopping is used (with patience parameter 10) to avoid overtraining and have sufficient generalization.

We also train an ensemble of boosted decision trees using XGB[41] with a maximum depth of 10, a learning rate of 0.3, and L2-regularization parameter $\lambda = 2000$.

To train and optimize XGB, we use 100 rounds of training and start with the default choices for the various parameters. We evaluate values of the learning rate $\eta \in \{0.001, 0.002, 0.003, 0.005, 0.008, 0.01, 0.02, 0.03, 0.05, 0.08, 0.1, 0.2, 0.3, 0.5, 0.8\}$ at tree depths of 5, 8, 10, 12, 15, and 20. Some of these parameters give small improvements in AUC over the defaults at value of the L2-regularization parameter $\lambda = 1$. Far larger improvements are found when λ is increased. Hence we hold the other parameters fixed and evaluate $\lambda \in \{5, 10, 20, 50, 100, 200, 500, 1000, 1500, 1800, 2000, 2200, 2500\}$, finding the approximate optimum AUC on the test set at 2000. Testing again, the tree depth and η are found to have minimal effect on the AUC (significantly smaller than the error), and $\eta = 0.3$ and tree depth 10 are chosen as the approximate optimum.

We note that the DNN and XGB settings are selected so as to prevent overtraining.

3 Mapping weak classifier selection to the Ising problem

In this section we closely follow Ref. [6], with slight changes of notation. Let V be the event space, consisting of vectors $\{\vec{x}\}$ that are either signal or background. We define a weak classifier $c_i(\vec{x}) : V \mapsto \mathbb{R}$, $i = 1, \ldots N$, as classifying event $\vec{x}$ as signal (background) if $c_i(\vec{x}) > 0$ ($c_i(\vec{x}) < 0$). We normalize each weak classifier so that $|c_i| \leq 1/N$. We introduce a binary weights vector $\vec{w} \in \{0,1\}^N$ and construct a strong classifier $R_{\vec{w}}(\vec{x}) = \sum_i w_i c_i(\vec{x}) \in [-\|\vec{w}\|/N, \|\vec{w}\|/N]$. The event $\vec{x}$ is correspondingly classified as signal (background) if $R_{\vec{w}}(\vec{x}) > 0$ ($R_{\vec{w}}(\vec{x}) < 0$). The weights $\vec{w}$ are to be determined; they are the target of the solution of the Ising problem.

Let $\mathscr{T} = \{\vec{x}_\tau, y_\tau\}$ denote a given set of training events, where $\vec{x}_\tau$ is an event vector collecting the values of each of the variables we use, and $y_\tau = \pm 1$ is a binary label for whether $\vec{x}_\tau$ is signal ($+1$) or background (-1). Let $Q_{\vec{w}}(\vec{x}) = \mathrm{sign}[R_{\vec{w}}(\vec{x})]$, so that $Q_{\vec{w}}(\vec{x}) = +1$ (-1) denotes signal (background) event classification. Thus $y_\tau Q_{\vec{w}}(\vec{x}_\tau) = +1$ if $\vec{x}_\tau$ is correctly classified as signal or background (y_τ and $Q_{\vec{w}}(\vec{x}_\tau)$ agree), and $y_\tau Q_{\vec{w}}(\vec{x}_\tau) = -1$ if $\vec{x}_\tau$ is incorrectly classified (y_τ and $Q_{\vec{w}}(\vec{x}_\tau)$ disagree). The cost function $L(\vec{w}) = \sum_\tau [1 - y_\tau Q_{\vec{w}}(\vec{x}_\tau)]/2$ thus counts the number of incorrectly classified training events, and minimizing it over all possible weight vectors returns the optimal set of weights, and hence the optimal strong classifier given the training set $\mathscr{T}$. To avoid overtraining and economize on the number of weak classifiers used, we can introduce a penalty term proportional to the number of weights, i.e., $\lambda\|\vec{w}\|$, where $\lambda > 0$ is the penalty strength. Thus the optimal set of weights for given λ is

$$\vec{w}_{\mathrm{opt}} = \mathrm{argmin}_{\{\vec{w}\}} \left[L(\vec{w}) + \lambda \|\vec{w}\| \right]. \tag{1}$$

This optimization problem cannot be directly mapped onto a quantum annealer, due to the appearance of the sign function. Instead we next introduce a relaxation to a quadratic form that is implementable on the current generation of D-Wave devices. Namely, using the training set we form the vector of strong classifier results $\vec{R}_{\vec{w}} = \{R_{\vec{w}}(\vec{x}_\tau)\}_{\tau=1}^{|\mathscr{T}|}$, the Euclidean distance measure $\delta(\vec{w}) = \|\vec{y} - \vec{R}_{\vec{w}}\|^2$ between the strong classifier and the set of training labels, and replace Eq. (1) by

$$\vec{w}_{\min} = \mathrm{argmin}_{\{\vec{w}\}} \delta(\vec{w}). \tag{2}$$

Finding $\vec{w}_{\mathrm{opt}}$ in this way is equivalent to solving a quadratic unconstrained binary optimization (QUBO) problem:

$$\vec{w}_{\min} = \mathrm{argmin}_{\{\vec{w}\}} \left[\sum_\tau R_{\vec{w}}(\vec{x}_\tau)^2 - 2 y_\tau R_{\vec{w}}(\vec{x}_\tau) + y_\tau^2 \right] = \mathrm{argmin}_{\{\vec{w}\}} \left[\sum_\tau \Big(\sum_{i,j} w_i w_j c_i(\vec{x}_\tau) c_j(\vec{x}_\tau) - 2 y_\tau \sum_i w_i c_i(\vec{x}_\tau) \Big) + |\mathscr{T}| \right]. \tag{3}$$

Regrouping the terms in the sum and dropping the constant we find:

$$\vec{w}_{\min} = \operatorname{argmin}_{\{\vec{w}\}} \left[\sum_{i,j} w_i w_j \left(\sum_{\tau} c_i(\vec{x}_\tau) c_j(\vec{x}_\tau) \right) - 2 \sum_i w_i \left(\sum_{\tau} c_i(\vec{x}_\tau) y_\tau \right) \right] = \operatorname{argmin}_{\{\vec{w}\}} \left[\sum_{i,j} C_{ij} w_i w_j - 2 \sum_i C_i w_i \right], \qquad (4)$$

where $C_{ij} = \sum_\tau c_i(\vec{x}_\tau) c_j(\vec{x}_\tau) = C_{ji}$ and $C_i = \sum_\tau c_i(\vec{x}_\tau) y_\tau$.

This has a tendency to overtrain. The reason is that $|R_{\vec{w}}(\vec{x}_\tau)| \le \|\vec{w}\|/N$, so that $|y_\tau - R_{\vec{w}}(\vec{x}_\tau)|^2 \ge (1 - \|\vec{w}\|/N)^2$, and hence $\delta(\vec{w}) = \sum_\tau |y_\tau - R_{\vec{w}}(\vec{x}_\tau)|^2 \ge |\mathscr{T}|(1 - \|\vec{w}\|/N)^2$. To minimize $\delta(\vec{w})$ the solution will be biased toward making $\|\vec{w}\|$ as large as possible, i.e., to include as many weak classifiers as possible. To counteract this overtraining tendency we add a penalty term that makes the distance larger in proportion to $\|\vec{w}\|$, i.e., $\lambda \|\vec{w}\|$ with $\lambda > 0$, just as in Eq. (1). Thus we replace Eq. (4) by

$$\vec{w}_{\min} = \operatorname{argmin}_{\{\vec{w}\}} \left[\sum_{i,j} C_{ij} w_i w_j + \sum_i (\lambda - 2C_i) w_i \right], \qquad (5)$$

The last step is to convert this QUBO into an Ising problem by changing the binary w_i into spin variables $s_i = \pm 1$, i.e., $w_i = (s_i + 1)/2$, resulting in:

$$\vec{s}_{\min} = \operatorname{argmin}_{\{\vec{s}\}} \left[\frac{1}{4} \sum_{i,j} C_{ij} s_i s_j + \frac{1}{2} \sum_{i,j} C_{ij} s_i + \frac{1}{2} \sum_i (\lambda - 2C_i) s_i \right], \qquad (6)$$

where we use the symmetry of C_{ij} to write the middle term in the second line, and we drop the constant terms $\frac{1}{4} \sum_{i,j} C_{ij}$ and $\frac{1}{2} \sum_i (\lambda - 2C_i)$. We now define the couplings $J_{ij} = \frac{1}{4} C_{ij}$ and the local fields $h_i = \frac{1}{2} \left(\lambda - 2C_i + \sum_j C_{ij} \right)$. The optimization problem is then equivalent to finding the ground state $\vec{s}_{\min} = \operatorname{argmin}_{\{\vec{s}\}} H$ of the Ising Hamiltonian

$$H_{\text{Ising}} = \sum_{i<j}^{N} J_{ij} s_i s_j + \sum_{i=1}^{N} h_i s_i. \qquad (7)$$

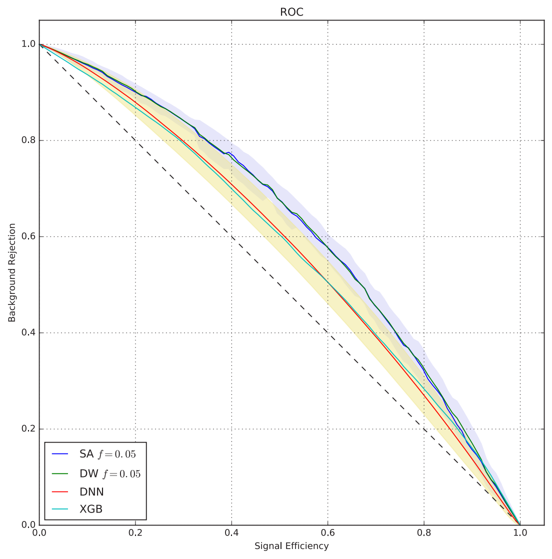
In the main text and hereafter, when we refer to λ it is measured in units of $\max_i(C_i)$ (e.g., $\lambda = 0.05$ is shorthand for $\lambda = 0.05 \max_i(C_i)$).

4 Robustness of QAML to MCMC mismodelling

Two essential steps are involved in the construction of the weak classifiers in our approach. First, we remove information about the tails of the distributions of each variable and use the corresponding truncated single-variable distributions to construct weak classifiers. Second, since the single-variable classifiers do not include any correlations between variables, we include additional weak classifiers built from the products/ratios of the variables, where after taking the products/ratios we again apply the same truncation and remove tails. That is, our weak classifiers account only for one and two-point correlations and ignore all higher order correlations in the kinematic variable distribution. The particular truncation choice to define the weak classifiers as a piecewise linear function defined only by a central percentile (30th or 70th, chosen during construction) and two percentiles in the tails (10th and 90th) means that the MC simulations only have to approximately estimate those four percentiles of the marginals and the correlations between the variables. Any MC simulation which is unable to approximate the 10th, 30th, 70th, and 90th percentiles of the marginal distribution for each dimension of the dataset and the products between them would surely not be considered acceptably similar to the target distribution for use in HEP data analyses, as it is effectively guaranteed to be wrong in the higher order correlations and thus in its approximation of the true distribution. Meanwhile, typical machine learning approaches for this problem use arbitrary relationships across the entire training dataset, including the tails and high-order correlations, and so are likely to be more sensitive to any mismodelling.

5 Receiver operating characteristic (ROC)

Any classifier may be characterized by two numbers: the true positive and true negative rates, in our case corresponding to the fraction of events successfully classified as signal or background, respectively. Since our classifiers all return floating point values in $[-1, 1]$, to construct a binary classifier we introduce a cut in this range, above and below which we classify as signal and background, respectively. Since this cut is a free parameter, we vary it across



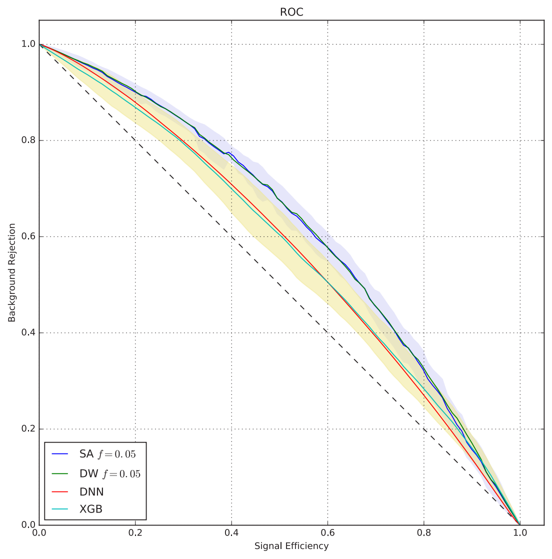


Figure 1. The ROC curves for the annealer-trained networks (DW and SA) at $f = 0.05$, DNN, and XGB. Error bars are defined by the variation over the training sets and statistical error. Both panels show all four ROC curves. Panel (a) [(b)] includes 1σ error bars only for DW and DNN [SA and XGB], in light blue and pale yellow, respectively. Results shown are for the 36 variable networks at $\lambda = 0.05$ trained on 100 events. The annealer trained networks have a larger area under the ROC curve

the entire range and plot the resulting parametric curve of signal acceptance (true positive, ε_S) and background rejection (true negative, r_B), producing a receiver operating characteristic (ROC) curve.[42]

More explicitly, consider a labeled set of validation events, $\mathscr{V} = \{\vec{x}_v, y_v\}$, with $y_v = 0$ or 1 if $\vec{x}_v$ is background or signal, respectively, and a strong classifier $R_{\vec{w}}(\vec{x})$. The latter is constructed from a given set of weak classifiers and a vector of weights $\vec{w}$ previously obtained from training over a training set $\mathscr{T}$. The strong classifier outputs a real number $R_{\vec{w}}(\vec{x}_v) = \sum_i w_i c_i(\vec{x}_v)$. To complete the classifier, one introduces a cut, O_c, such that we classify event $\vec{x}_v$ as signal if $R_{\vec{w}}(\vec{x}_v) > O_c$ and background if $R_{\vec{w}}(\vec{x}_v) < O_c$. If we evaluate the strong classifier on each of the events in our validation set $\mathscr{V}$, we obtain a binary vector of classifications, $\vec{C} = \{C_v\}$, with entries 0 denoting classification as background and 1 denoting classification as signal. By comparing C_v to y_v for all v we can then evaluate the fraction of the events which are correctly classified as background, called the true negative rate or "background rejection" r_B (equal to the number of times $C_v = y_v = 0$, divided by the total number of actual background events), and the fraction of events correctly classified as signal or "signal efficiency" ε_S (equal to the number of times $C_v = y_v = 1$, divided by the total number of actual signal events). For a given strong classifier, these values will be a function of the cutoff O_c. Plotting $r_B(O_c)$ against $\varepsilon_S(O_c)$ yields a parametric curve, dubbed the "receiver operator characteristic" (ROC) curve, as shown in Fig. 1. Note that the cutoffs are trivial to adjust while all the computational effort goes into forming the networks, so one can vary O_c essentially for free to tune the performance of the network to suit one's purposes.

In other words, for a given strong classifier, i.e., solution/state, we can evaluate its output as a floating point number on each of the values in our data set, and for any value of a cut on $[-1,1]$ this results in a single classification of the test data, $\vec{C}$. One can then evaluate the true positive and true negative rates by computing $\vec{C}_v \cdot \vec{y}_k$ where $k \in [S,B]$ (signal, background), $y_k^i = 1$ if datum i is in ensemble k and is 0 otherwise.

When we take $f > 0$ and accept excited states with energy $E < (1-f)E_{GS}$ as "successes", we have a set of networks (labeled by f) for each training set. We simply take the supremum over the f-labeled set of values of r_B at each value of ε_S, to form the ROC curve for the classifier formed by pasting together different classifiers over various ranges of ε_S.

To estimate the error due to limited test sample statistics, we reweight each element of the test set with weights $\vec{w}$ drawn from a Poisson distribution with mean 1, effectively computing $\sum_i w_i p_i y_k^i$. The weights on the elements of the test set are determined for all elements at once and we evaluate all strong classifiers using the same weights. For a single weight vector, we evaluate many values of the cut, and use linear interpolation to evaluate it in steps of 0.01 in the region $[0,1]$. This gives us the true negative rate as a function of the true positive rate for a single weighting corresponding to a single estimated ROC curve. When constructing a composite classifier from multiple states, we are identifying regions of signal efficiency in which one should use one of the states rather than the others, namely, we take the maximum background rejection rate over the states for each value of signal efficiency.

Repeating for many reweightings we get many ROC curves all of which are consistent with our data, and thus the standard deviation across weights on a single training set at each value of signal efficiency serves as an estimate of the statistical uncertainty in our ROC curves.

To estimate variation due to the choice of the training set towards reproductions of the procedure and results, for a given training size we generate multiple disjoint training sets and use the standard deviation in mean performance across training sets as our estimate of the error on the model resulting from the particular choice of training set. When we compute the difference between two ROCs or AUROCs, we hold the training set and weight vector fixed, take the difference, and then perform statistics over the weights and training sets in the same manner as above. Errors in the AUROC were estimated similarly, taking the AUROC for each Poisson weight vector and training set (fold) instead of $r_B(\varepsilon_S)$. This is the procedure leading to Fig. 3 in the main text. An example of the ROC curves is given in Fig. 1. At the scale of that plot, it is virtually impossible to tell the detailed differences between SA and DW or the various values of f, so we use plots of differences of AUCs to extract more detailed information about the ROC curves. This leads to Fig. 4 in the main text. Additional difference plots are given in Sec. 10 below.

6 Quantum annealing and D-Wave

In QA, we interpret the Ising spins s_i in the Ising Hamiltonian (7) as Pauli operators σ_i^z on the i^{th} qubit in a system of N qubits. QA is inspired by the adiabatic theorem,[43] namely if the Hamiltonian is interpolated from an initial Hamiltonian $H(t=0)$ to a final Hamiltonian $H(t=t_a)$ sufficiently slowly compared to the minimum ground-to-first-excited state gap of $H(t)$, the system will be in the ground state of $H(t=t_a)$ with high probability, provided it was initialized in the ground state of $H(t=0)$. Thus, one can evolve from a simple, easy to initialize Hamiltonian at $t=0$ to a complicated Hamiltonian with an unknown ground state at $t=t_a$, where t_a is known as the annealing time. In QA, the initial Hamiltonian is a transverse field $H_X = \sum_i \sigma_i^x$, and the final Hamiltonian is the Ising Hamiltonian (7), with the time-dependent Hamiltonian taking the form $H(t) = A(t)H_X + B(t)H_{\text{Ising}}$, where $A(t)$ is monotonically decreasing to 0 and $B(t)$ is monotonically increasing from 0; these functions are known as the annealing schedule. QA can be seen as both a generalization and a restriction of adiabatic quantum computation[44] (for a review see[15]): as a restriction, QA typically requires the initial Hamiltonian to be a sum of σ^xs and the final Hamiltonian be diagonal in the computational basis (i.e., a sum of σ^z terms), while, as a generalization, it undergoes open-system dynamics and need not remain in the ground state for the entire computation.

Current and near-generation quantum annealers are naturally run in a batch mode in which one draws many samples from a single Hamiltonian. Repeated draws for QA are fast. The DW averages approximately 5000 samples per second under optimal conditions. We take advantage of this by keeping all the trial strong classifiers returned and not restricting to the one with minimum energy.[1] The DW has 1098 superconducting Josephson junction flux qubits arranged into a grid, with couplers between the qubits in the form shown in Fig. 3, known as the Chimera graph. The annealing schedule used in the DW processor is given in Fig. 4. The Chimera graph is not fully connected, a recognized limitation as the Ising Hamiltonian (7) is fully connected, in general. To address this, we perform a *minor embedding operation*.[45,46] Minor embedding is the process whereby we map a single logical qubit in H_{Ising} into a physical ferromagnetic ($J_{ij} = -1$) chain of qubits on DW. For each instance we use a heuristic embedding found via the D-Wave API, that is as regular and space-efficient as possible for our problem sizes.

Given a minor embedding map of logical qubits into a chain of physical qubits, we divide the local fields h_i equally among all the qubits making up the chain for logical qubit i, and divide J_{ij} equally among all the physical couplings between the chains making up logical qubits i and j. After this procedure, there remains a final degree of freedom: the chain strength J_F. If the strength of the couplers in the ferromagnetic chains making up logical qubits is defined to be 1, then the maximum magnitude of any other coupler is $\max\left(\max_i(\{|h_i|\}), \max_{i,j}(\{|J_{ij}|\})\right) = \frac{1}{J_F}$. There is an optimal value of J_F, generally. This is due to a competition between the chain needing to behave as a single large qubit and the problem Hamiltonian needing to drive the dynamics.[47] If J_F is very large, the chains will "freeze out" long before the logical problem, i.e., the chains will be far stronger than the problem early on, and

[1]The energy is effectively a function of error on the training set of the weak classifiers, hence is distinct from the measures used to directly judge classifier performance, such as the area under the ROC curve.

the transverse field terms will be unable to induce the large, multi-qubit flipping events necessary to explore the logical problem space. Similarly, if J_F is very weak, the chains will be broken (i.e., develop a kink or domain wall) by tension induced by the problem, or by thermal excitations, and so the system will generally not find very good solutions. Ideally, one wants the chains and the logical problem to freeze at the same time, so that at the critical moment in the evolution both constraints act simultaneously to determine the dynamics. For the results shown here, we used $J_F = 6$ with an annealing time $t_a = 5\mu$s. To deal with broken chains, we use majority vote on the chain with a coin-toss tie-breaker for even length chains. Detailed analysis of the performance of this strategy in the context of error correction can be found in the literature.[48,49]

Figure 5 shows the average minimum energy returned by the DW, rescaled by the training size (to remove a linear scaling), as a function of the chain strength and training size. We see that the smallest training size ($N = 100$) has a smaller average minimum energy than the rest of the training sizes, and that there is only a very slight downward tendency as the chain strength J_F increases for the larger training sizes.

Figure 6 plots the fractional deviation of the minimum energy returned by the DW relative to the true ground state energy, averaged over the training sets. While the DW's minimum energy returned approaches the true ground state, it seems to converge to $\approx 5\%$ (i.e. $f \approx 0.05$) above the ground state as we increase the chain strength, for all training sizes ≥ 1000. In this case, we were not able to find the optimal chain strength in a reasonable range of chain strengths, and instead simply took the best we found, $J_F = 6$. As discussed in Sec. 8, the DW processor suffers from noise sources on the couplers and thermal fluctuations, and it seems that this poses significant challenges for the performance of the quantum annealer. It is possible that even larger chain strengths may resolve the issue, but given the convergence visible in Fig. 6, it seems likely that $J_F = 6$ is already near the optimum.

7 Simulated annealing

In simulated annealing,[10] we initialize the vector $\vec{s}$ in a random state. At each time step, we create a trial vector $\vec{s}'$ by flipping one of the spins in $\vec{s}$, selected at random. We accept the trial vector using the Metropolis update rule:[50] the new state is accepted with probability 1 if $H(\vec{s}') < H(\vec{s})$ (i.e., lower energy states are accepted deterministically), whereas if $H(\vec{s}') > H(\vec{s})$, we accept the trial vector with probability $\exp\{-\beta(H(\vec{s}') - H(\vec{s}))\}$ for some inverse temperature β. After we have attempted N spin flips, which amounts to one sweep, we then increase the inverse temperature β according to some schedule. At first, $\beta \ll 1$ and the system quickly drifts through the space of possible states. As β grows the system settles into lower lying valleys in the energy landscape, and ultimately ceases to evolve entirely in the limit of infinite β (zero temperature). Our simulations used $\beta_{\text{init}} = 0.1$ and $\beta_{\text{final}} = 5$, and used a linear annealing schedule (i.e., if we perform S sweeps, we increase β after each sweep by $\frac{\beta_{\text{final}} - \beta_{\text{init}}}{S}$). These parameters have generally performed well in other studies.[33] All SA data in the main text and presented here is at 1000 sweeps, however we also tested SA at 100 sweeps, and found a negligible difference in overall performance, as seen in Fig. 7, where the integrated difference of the ROC curves is found to be statistically indistinguishable from 0.

8 Effect of noise on processor

Internal control error (ICE) on the current generation of D-Wave processors is effectively modeled as a Gaussian centered on the problem-specified value of each coupler and local field, with standard deviation 0.025, i.e., a coupler J_{ij} is realized as a value drawn from the distribution $N(J_{ij}, 0.025)$ when one programs a Hamiltonian. Figure 8 contains a histogram of the ideal values of the embedded couplers corresponding to connections between logical qubits across all 20 problem instances of 36 variables at 20000 training events. One can see that the ideal distribution has some structure, with two peaks. However, if one resamples values from the Gaussian distribution induced by ICE, one finds that many of the features are washed out completely. This suggests that the explanation for the flattening out of the performance of QA as a function of training size (recall Fig. 3 in the main text) is due to this noise issue. Thus we investigate this next.

Figures 9-11 tell the story of the scaling of the couplers with training size. Figure 9 shows linear scaling of the maximum Hamiltonian coefficient with training size. We observe wider variation at the smallest training sizes, but overall the precision scales linearly with training size. This is confirmed in Figure 10, which shows the maximum coefficient normalized by the training size. Since this value is constant for sufficiently large training sizes, the maximum value scales linearly with size. At first glance, this indeed suggests an explanation for why the performance of QA using the DW levels off as a function of training size: the coupling values pass 20 (half the scale of the errors which is $\approx 1/0.025 = 40$). However, absolute numbers are not necessarily informative, and Fig. 11 dispels this explanation.

Figure 11 shows the ratio of the median coefficient to the maximum coefficient, thereby showing the scale of *typical* Hamiltonian coefficients on the DW prior to rescaling for chain strength (which for the most of the data here would reduce the magnitude by a further factor of 6).

Since all the different types of coefficient ratios are constant with system size, we have effectively no scaling with training size of the precision of the couplers. This means that the scaling of precision with training size cannot explain the saturation of performance with increasing training size.

However, the magnitudes here are quite small, and so once one accounts for rescaling the energies, typical couplers are expected to be subject to a significant amount of noise, even causing them to change sign. This effect likely explains, at least in part, the difficulties the DW has in finding the true ground state, as discussed above and seen in Fig. 6, where even at the largest chain strength we still find that the DW's typical minimum energy is $\sim 5\%$ above the ground state energy.

9 Sensitivity to variation of the parameters of weak classifier construction

When constructing the weak classifiers, we choose to define v_{cut} as the 70th percentile of the signal distribution. This choice is arbitrary. To test the effect of this value on the classifier performance we use identical training sets and values of both 60% and 80% and compare them to our primary estimate of 70%. The results for both the minimum energy returned ($f = 0$) and $f = 0.05$ for each are shown in Fig. 12.

Note that every training set has the same ground state configuration at 70% and 80%. The ROCs and AUROC are then invariant across a wide range of v_{cut} values.

Figure 2 reproduces Fig. 3 from the main text, but also shows the AUROC for SA's optimal classifier (by energy) for various values of the regularization parameter λ. We find no significant variation, with the major features of SA being stable, namely the advantage at small training size and the saturation at around an AUROC of ≈ 0.64.

10 Difference between ROC curves plots

We show differences between ROC curves for various algorithms in Figs. 13-18. These form the basis for Fig. 4 in the main text, which gives the integral of the difference over signal efficiency. Figures 13 and 14 show the difference in background rejection $r_B^{\text{DW}} - r_B^{\text{SA}}$ as a function of the signal efficiency for $f = 0$ and $f = 0.05$, respectively. For $f = 0$ DW and SA are indistinguishable to within experimental error. For $f = 0.05$ SA slightly outperforms DW in the range of low signal efficiencies for training sizes ≥ 5000. The primary conclusion to draw from these plots is that SA differs from DW by roughly one standard deviation or less across the whole range, even though DW for training sizes larger than 100 struggles to find states within less than 5% of the ground state energy. This suggests a robustness of QAML, which (if it generalizes to other problems) significantly improves the potential to exploit physical quantum annealers to solve machine learning problems and achieve close-to-optimal classifier performance, even in the presence of significant processor noise.

Figures 15 and 16 show the ROC difference between DW and DNN and DW and XGB at $f = 0$, respectively. The two cases have broadly similar shapes. One clearly sees that QAML on DW outperforms DNN and XGB at the smallest training size in a statistically significant manner, but that the trend reverses for sizes ≥ 5000. Note, that at the scale of these diagrams, the gap between $f = 0$ and $f = 0.05$ is negligible.

Figures 17 (SA) and 18 (DW) show the difference between $f = 0$ and $f = 0.05$. SA and DW exhibit broadly similar behavior, with an improvement with excited states of $\approx 0.4\%$ in background rejection for SA and of $\approx 0.2\%$ for DW. The improvement increases with training size and is slightly larger for SA than DW (though this difference is likely simply noise, as it is less than half the standard deviation of each distribution). It should be noted that since QAML's comparative advantage against other techniques appears to be in the realm of small training sizes. However, this is the same range where including excited states has no benefit.

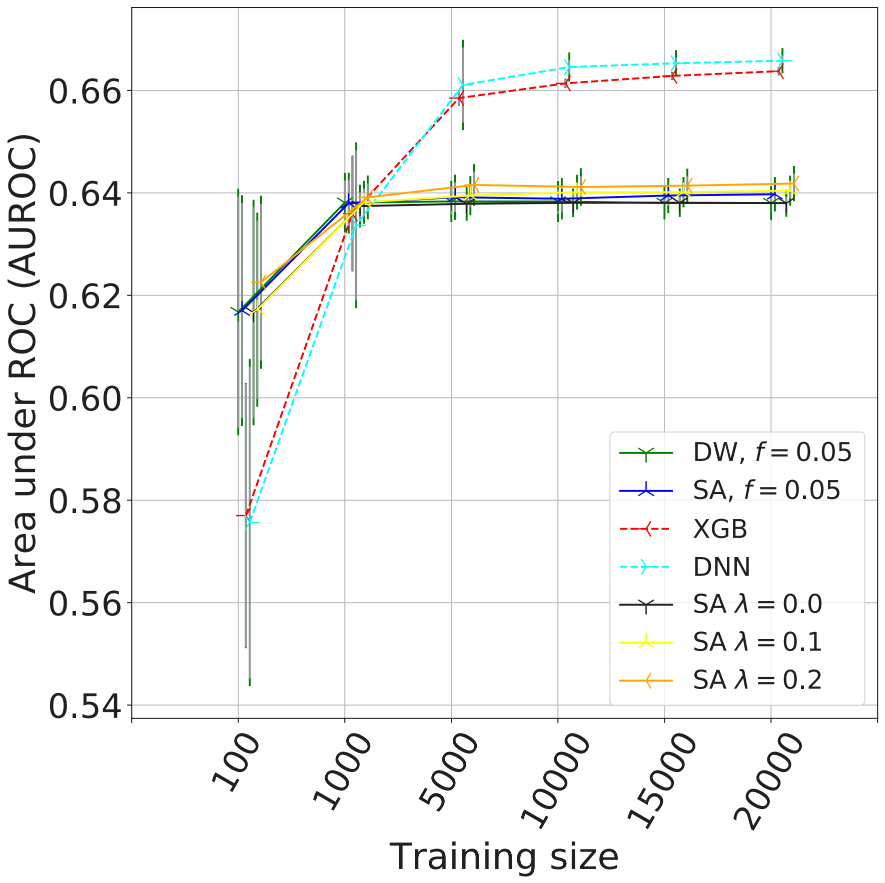


Figure 2. A reproduction of Fig. 3 from the main text, now including the optimal strong classifier found by SA at $f = 0$ for various values of the regularization parameter $\lambda = 0., 0.1, 0.2$. We find that this parameter has negligible impact on the shape of the AUROC curve, and that performance for SA always saturates at ≈ 0.64, with an advantage for QAML (DW) and SA over XGB and DNNs for small training sizes.

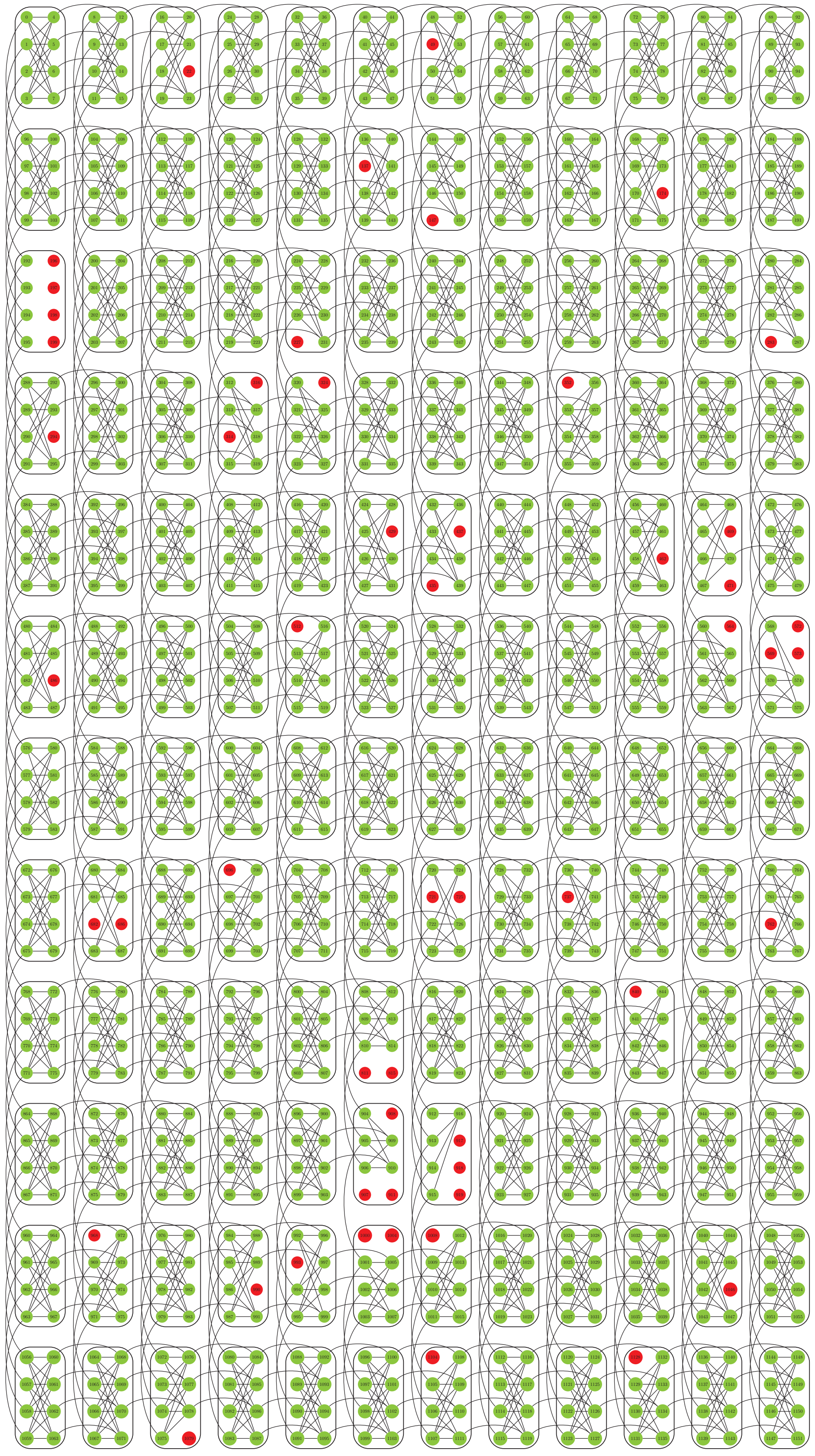


Figure 3. An 1152 qubit Chimera graph, partitioned into a 12×12 array of 8-qubit unit cells, each unit cell being a $K_{4,4}$ bipartite graph. Inactive qubits are marked in red, active qubits in green. There are a total of 1098 active qubits in the DW processor used in our experiments. Black lines denote active couplers.

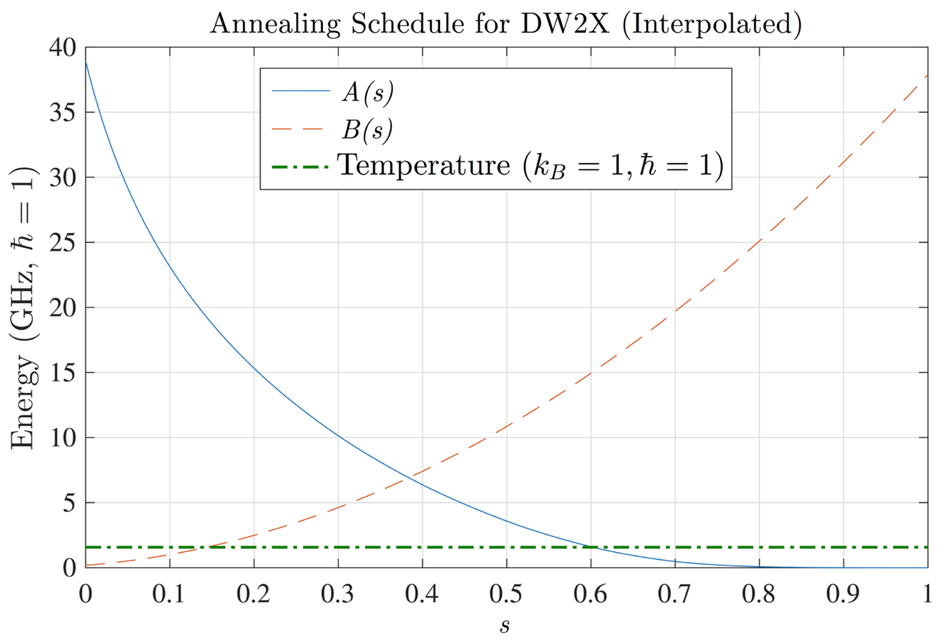


Figure 4. Annealing schedule used in our experiments.

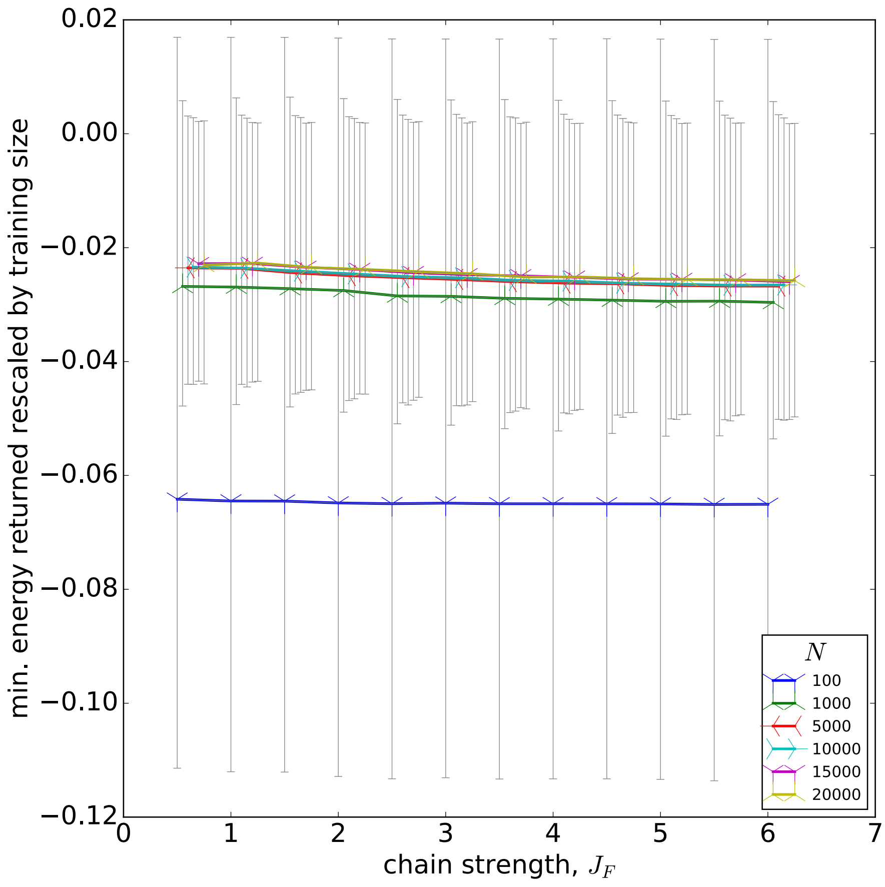


Figure 5. A plot of the minimum energy returned by the DW as a function of chain strength, rescaled by the number of training samples. I.e., for training size N, we plot E_m/N for minimum return energy E_m, where N is given in the legend.

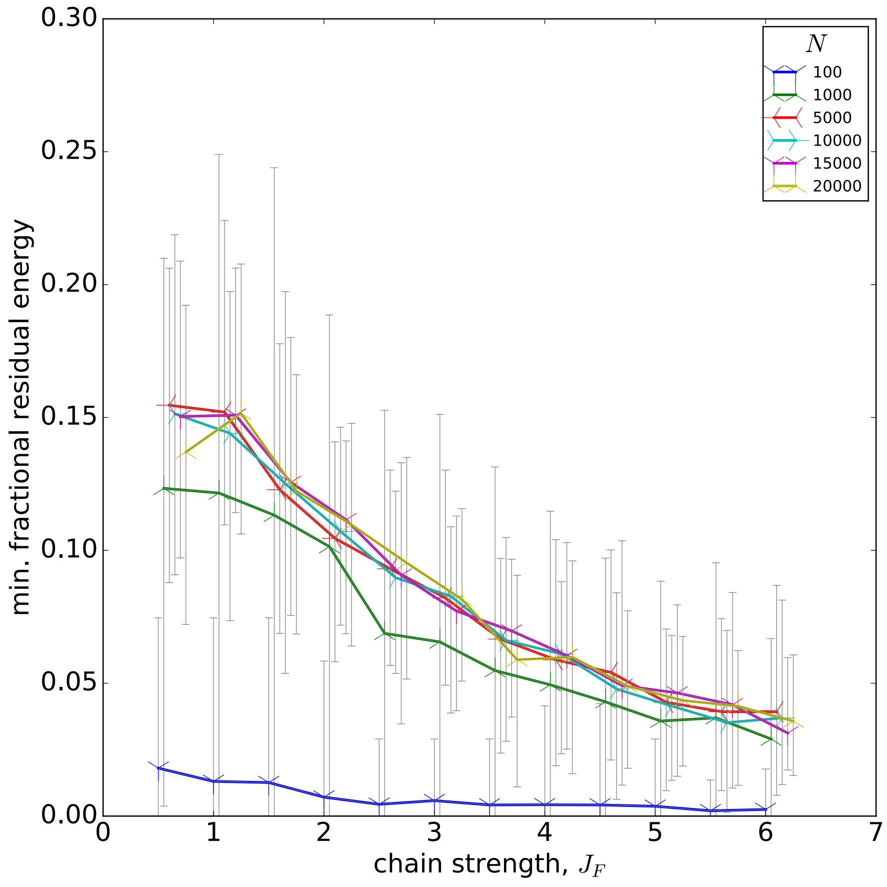


Figure 6. Plot of $(E_m - E_0)/E_0$ for minimum energy returned E_m and true ground state energy E_0, i.e., the minimum fractional reserve energy, averaged over the training sets, for each size and chain strength.

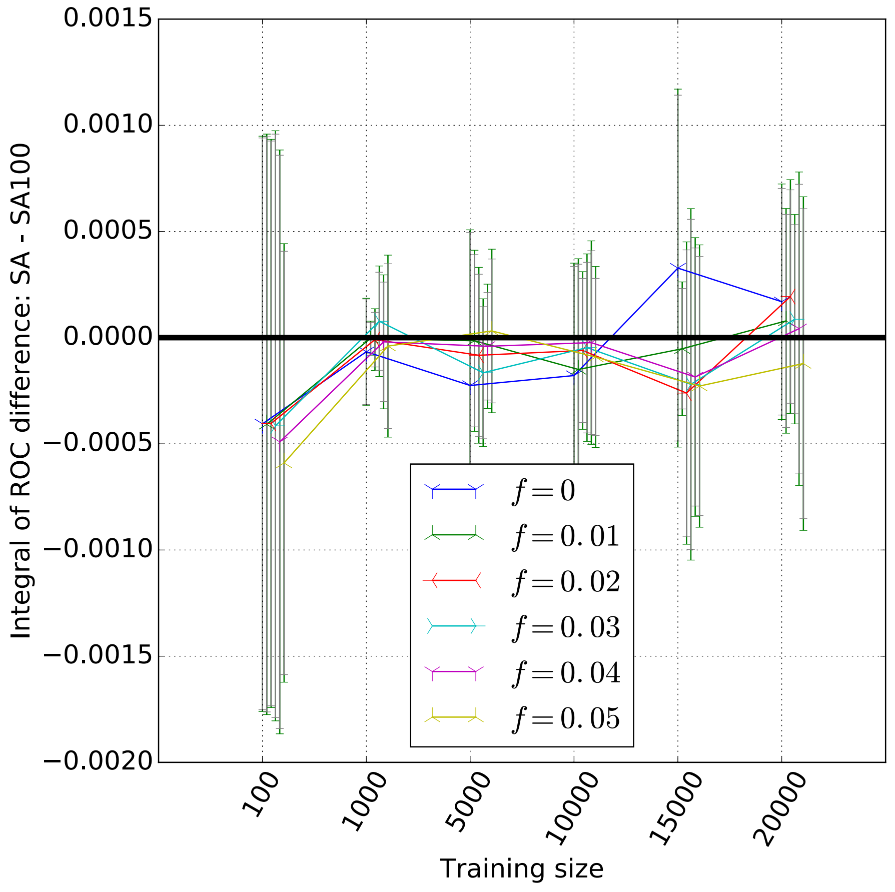
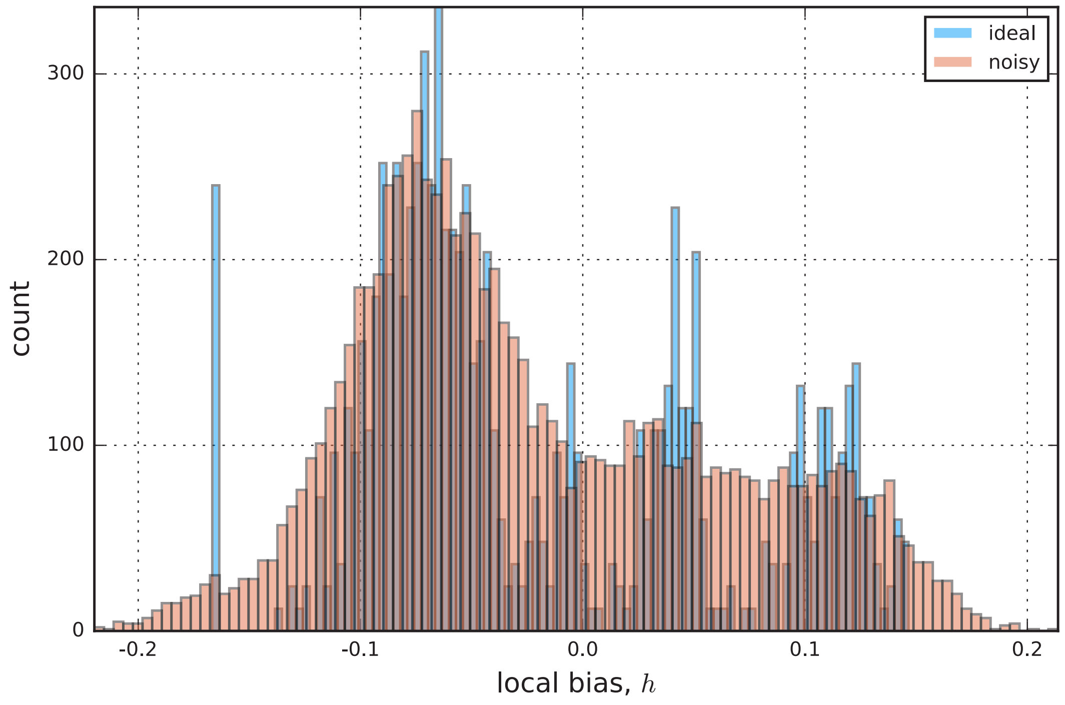


Figure 7. The integral of the difference of the ROC curves, i.e., the area between the ROC curves, for SA and SA100 for various thresholds of the energy and training size. SA at 100 and 1000 sweeps are effectively identical by this benchmark.



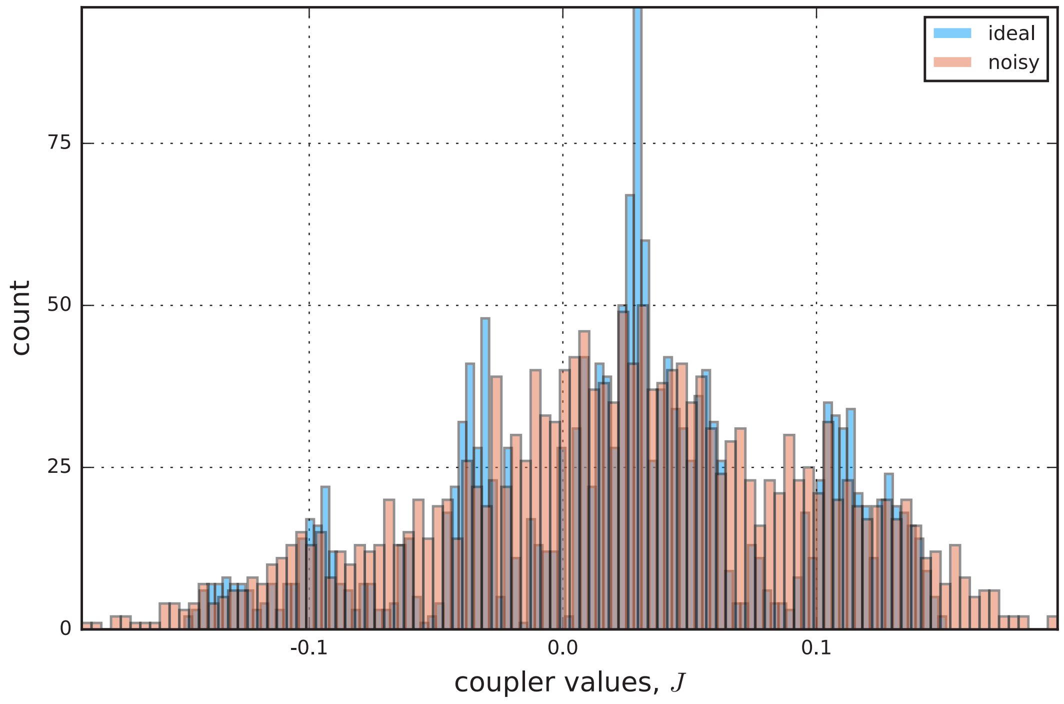


Figure 8. Histograms for the true (peaked) distribution of local biases and couplers, and the same distribution subject to point-wise Gaussian noise with zero mean and standard deviation 0.025, which is approximately the magnitude of errors on the DW couplers.

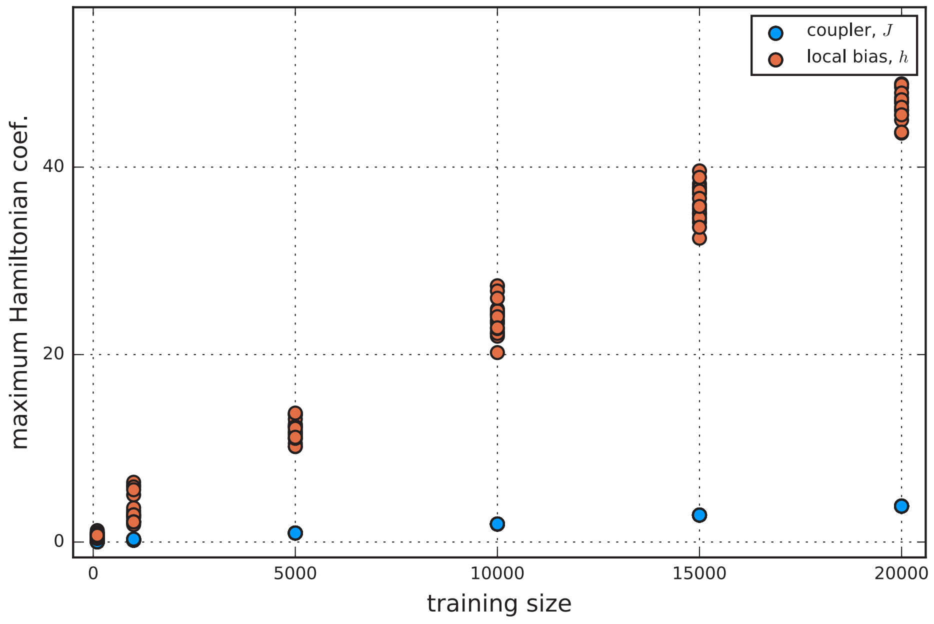


Figure 9. The maximum local bias and coupler term in the Hamiltonian across training sizes and training sets.

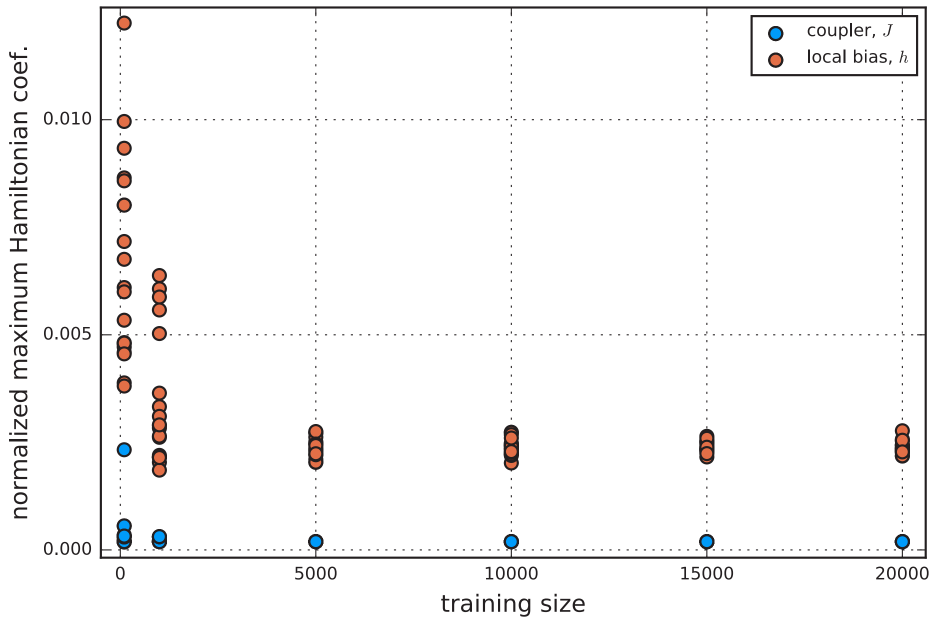


Figure 10. The maximum local bias and coupler term in the Hamiltonian across training sizes and training sets, normalized by the number of events in the training set. This makes it clear that the scaling of the Hamiltonian coefficients is linear in the training size, for training sizes ≥ 5000.

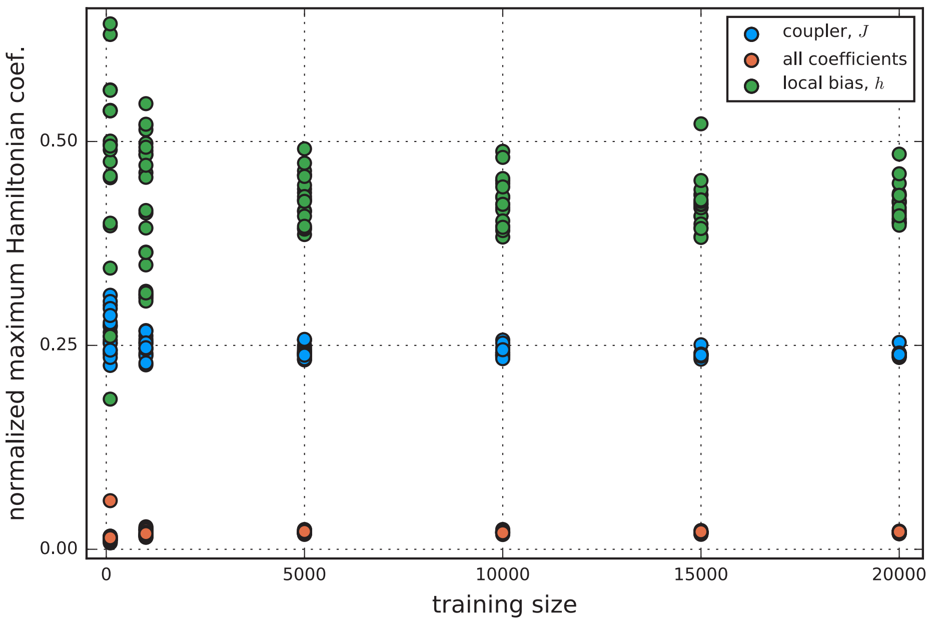


Figure 11. The ratio of the median coefficient by the maximum coefficient for the non-zero local biases, couplers, and both taken together.

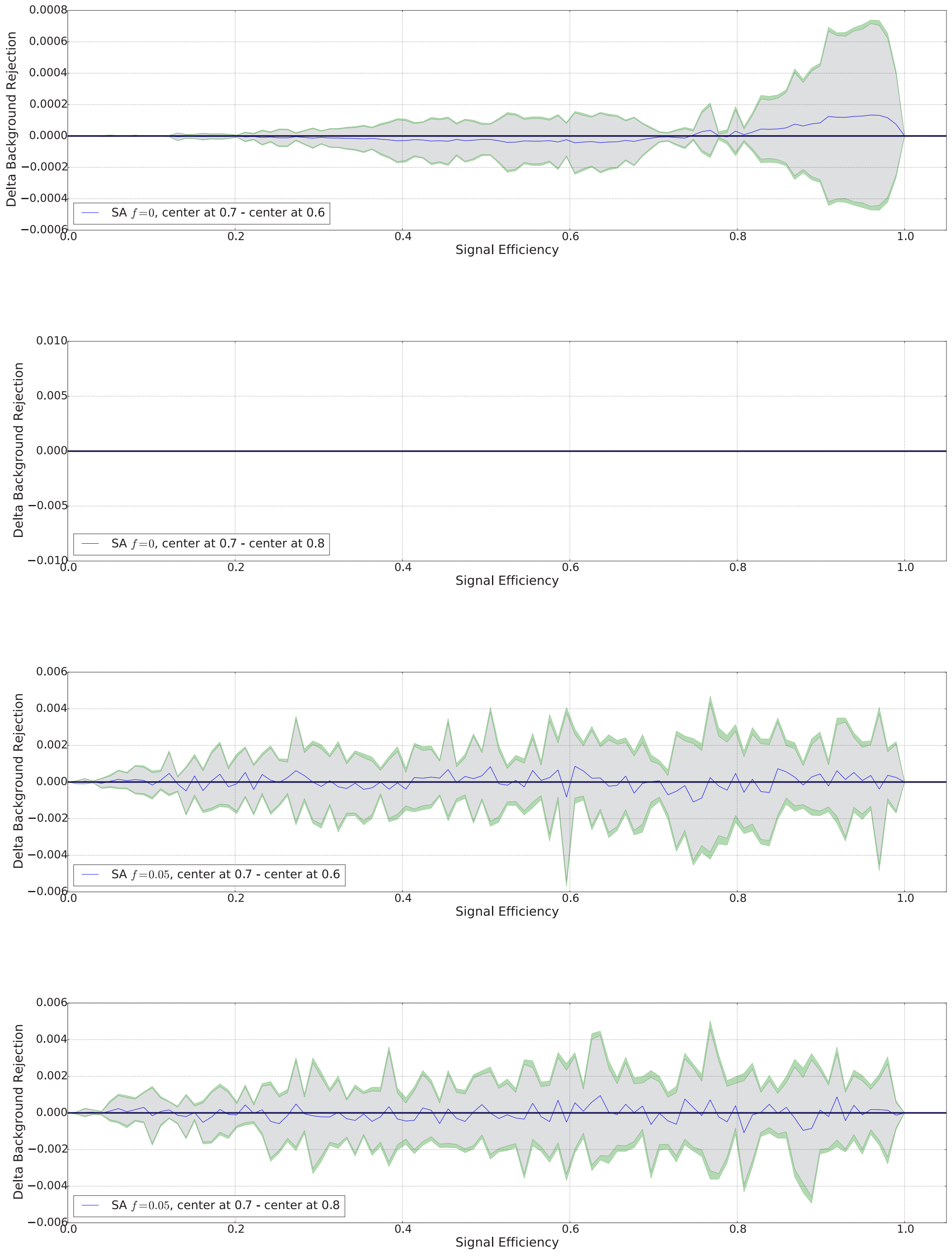


Figure 12. Difference between the ROC curve for SA at v_{cut} at the xth percentile during weak classifier construction and the curve using the yth percentile during the same for the ground state configuration. (a) $x=70$, $y=60$, $f=0$. (b) $x=70$, $y=80$, $f=0$. (c) $x=70$, $y=60$, $f=0.05$. (d) $x=70$, $y=80$, $f=0.05$.

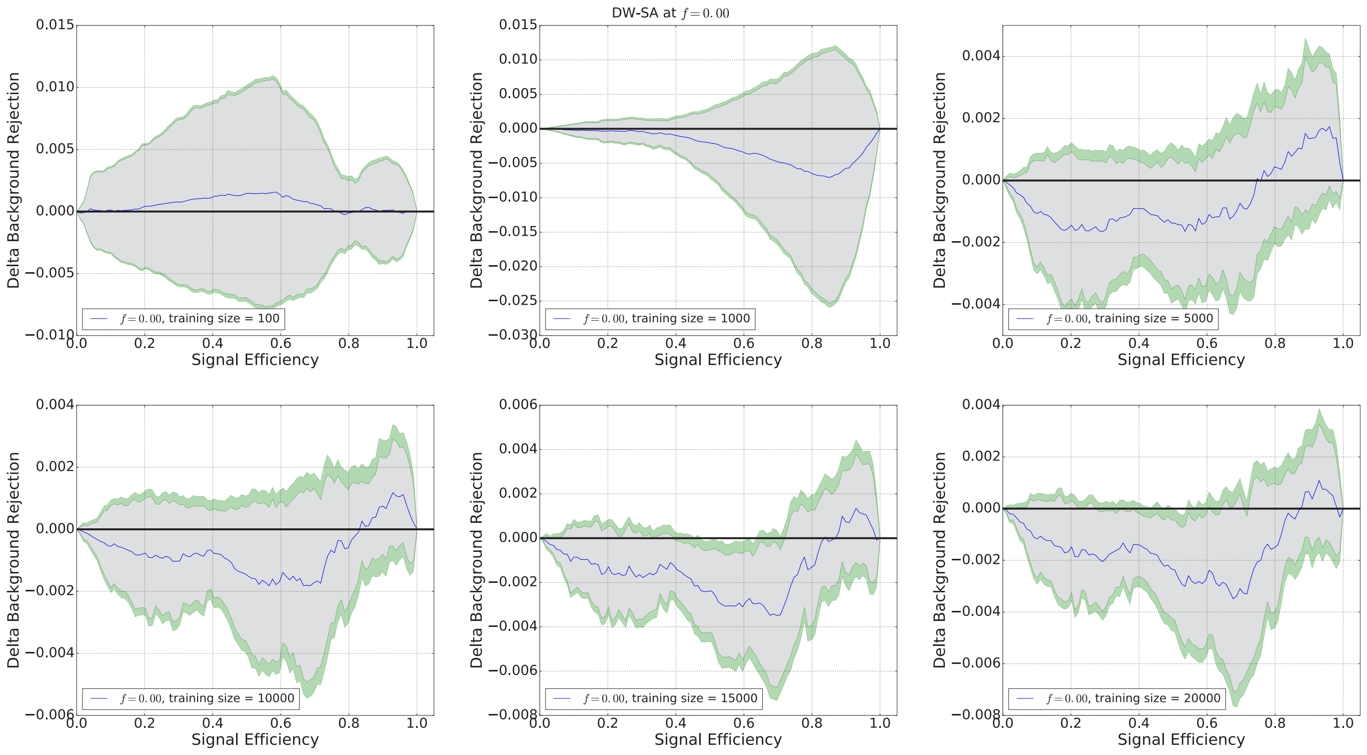


Figure 13. Difference between the ROC curves for SA and DW using the minimum energy returned.

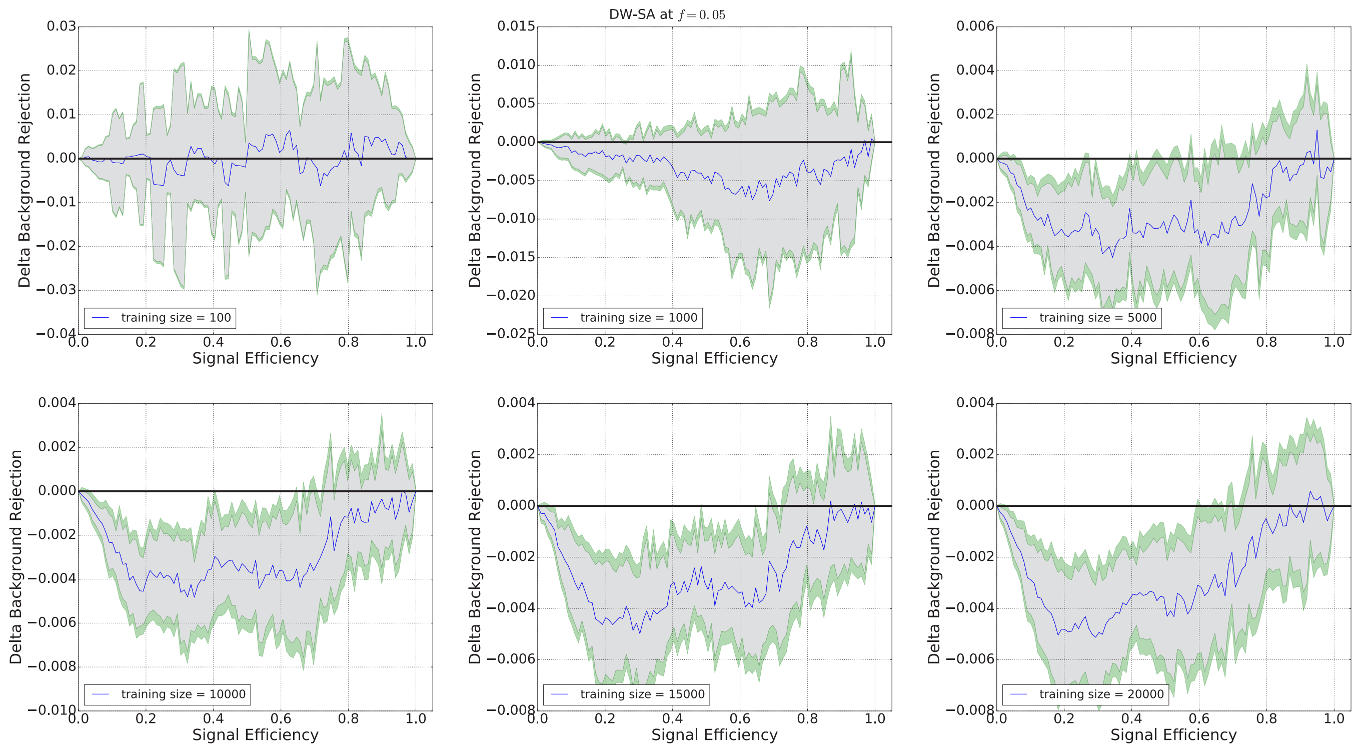


Figure 14. Difference between the ROC curves for SA and DW using all states within 5% of the minimum return energy.

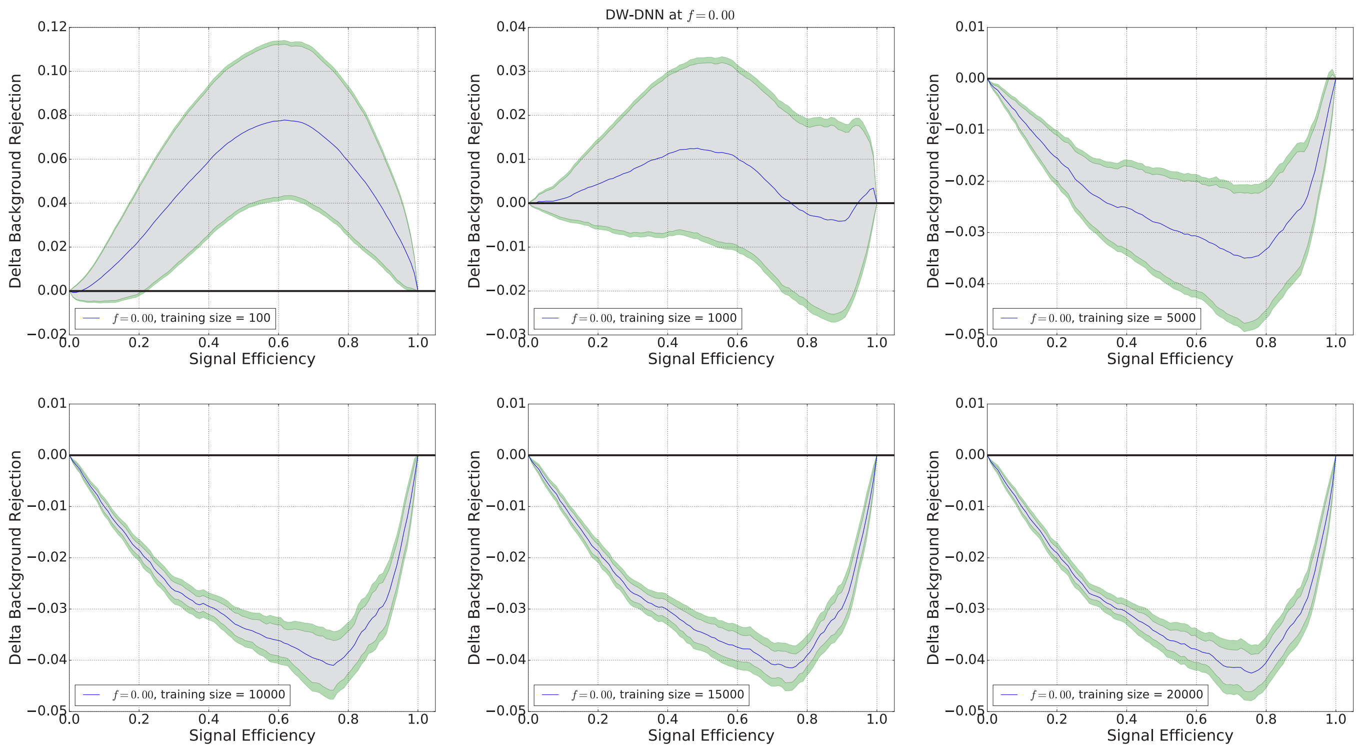


Figure 15. Difference between the ROC curves for DW and DNN using the minimum energy configuration from DW.

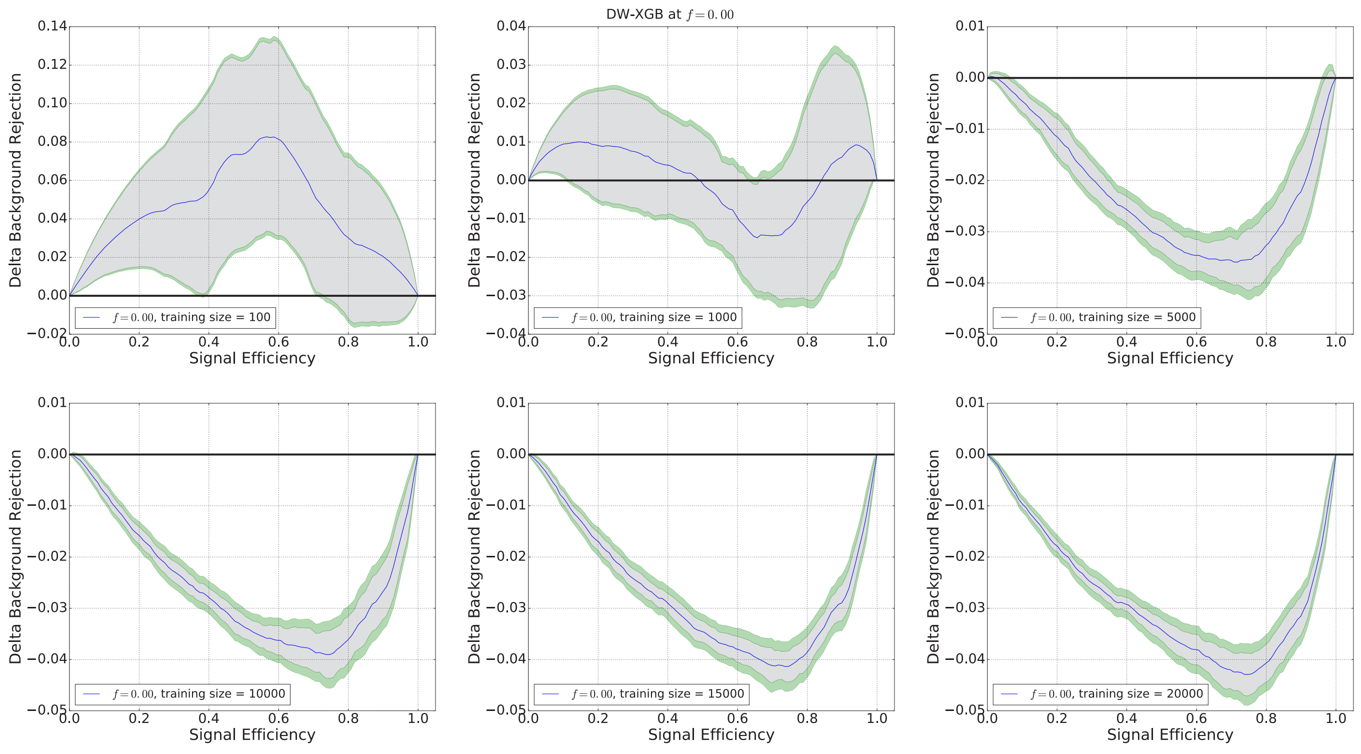
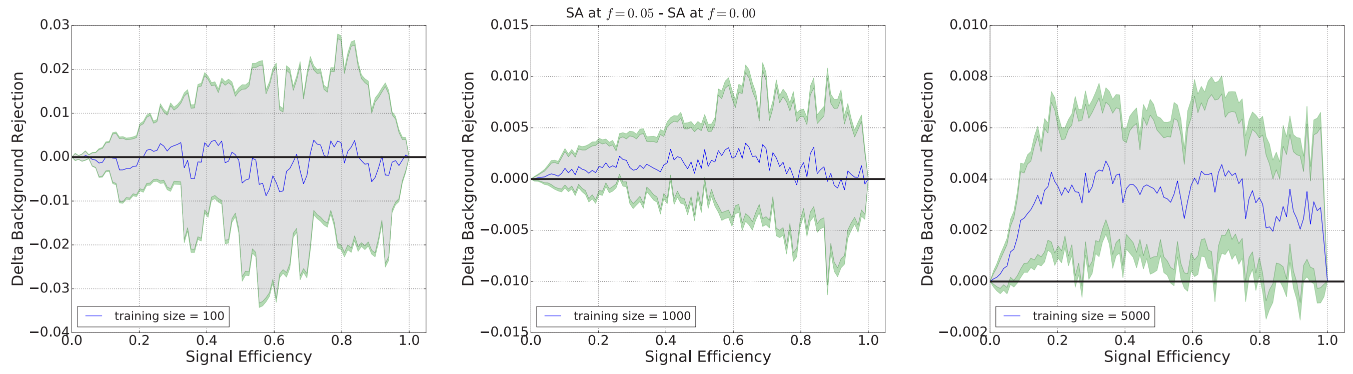


Figure 16. Difference between the ROC curves for DW and XGB using the minimum energy configuration from DW.



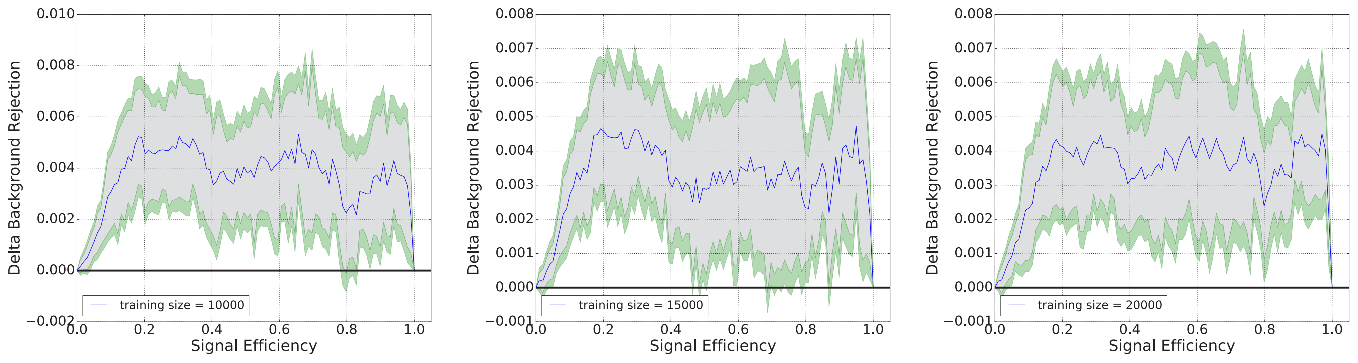
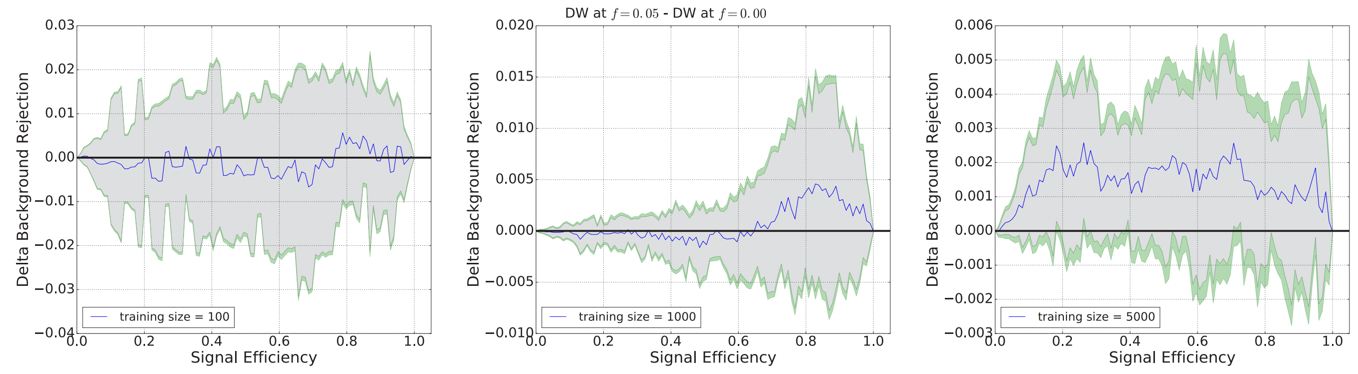


Figure 17. Difference between the ROC curves between the true ground state configuration and the $f = 0.05$ composite classifier from SA.



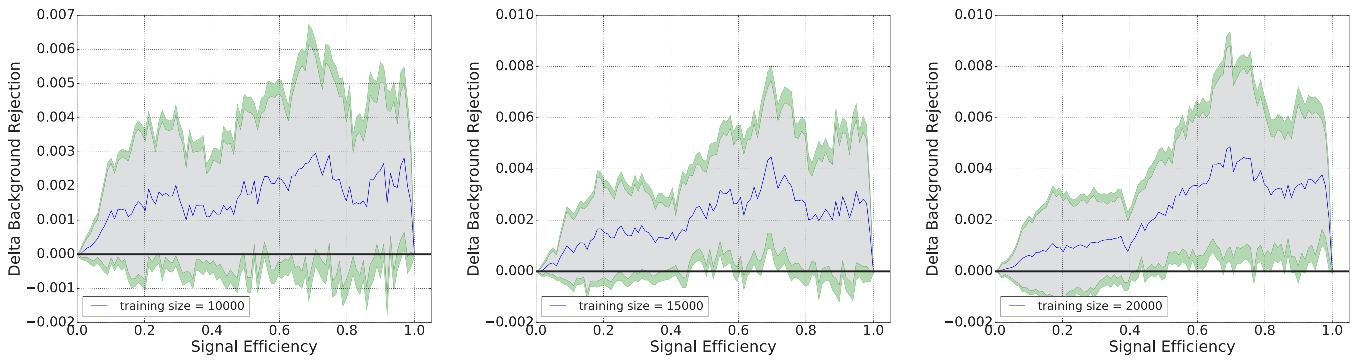


Figure 18. Difference between the ROC curves between the minimum energy state returned by DW and the $f = 0.05$ composite classifier from DW.

References

1. Le, Q. V. Building high-level features using large scale unsupervised learning. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, 8595–8598 (IEEE, 2013).
2. Arthur, L. *Big Data Marketing* (Wiley, 2013).
3. Alwall, J. *et al.* The automated computation of tree-level and next-to-leading order differential cross sections, and their matching to parton shower simulations. *JHEP* **07**, 079 (2014). 1405.0301.
4. Agostinelli, S. *et al.* GEANT4: A Simulation toolkit. *Nucl. Instrum. Meth.* **A506**, 250–303 (2003).
5. Neven, H., Denchev, V. S., Rose, G. & Macready, W. G. Training a binary classifier with the quantum adiabatic algorithm. *arXiv:0811.0416* (2008). URL http://arXiv.org/abs/0811.0416.
6. Pudenz, K. L. & Lidar, D. A. Quantum adiabatic machine learning. *Quantum Information Processing* **12**, 2027–2070 (2013). URL dx.doi.org/10.1007/s11128-012-0506-4.
7. Long, P. M. & Servedio, R. A. Random classification noise defeats all convex potential boosters. *Machine Learning* **78**, 287–304 (2010).
8. Manwani, N. & Sastry, P. S. Noise tolerance under risk minimization. *Cybernetics, IEEE Transactions on* **43**, 1146–1151 (2013).
9. Denchev, V., Ding, N., Neven, H. & Vishwanathan, S. Robust classification with adiabatic quantum optimization. In Langford, J. & Pineau, J. (eds.) *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*, 863–870 (ACM, New York, NY, USA, 2012). URL http://icml.cc/2012/papers/461.pdf.
10. Kirkpatrick, S., Gelatt, C. D. & Vecchi, M. P. Optimization by simulated annealing. *Science* **220**, 671–680 (1983). URL http://science.sciencemag.org/content/220/4598/671.
11. Katzgraber, H. G., Trebst, S., Huse, D. A. & Troyer, M. Feedback-optimized parallel tempering monte carlo. *J. Stat. Mech.* **2006**, P03018 (2006).
12. Kadowaki, T. & Nishimori, H. Quantum annealing in the transverse Ising model. *Phys. Rev. E* **58**, 5355 (1998). URL http://journals.aps.org/pre/abstract/10.1103/PhysRevE.58.5355.
13. Das, A. & Chakrabarti, B. K. *Colloquium*: Quantum annealing and analog quantum computation. *Rev. Mod. Phys.* **80**, 1061–1081 (2008).
14. Farhi, E. *et al.* A Quantum Adiabatic Evolution Algorithm Applied to Random Instances of an NP-Complete Problem. *Science* **292**, 472–475 (2001). URL http://www.sciencemag.org/content/292/5516/472.
15. Albash, T. & Lidar, D. A. Adiabatic quantum computing. *arXiv:1611.04471* (2016). URL http://arXiv.org/abs/1611.04471.
16. Kaminsky, W. M., Lloyd, S. & Orlando, T. P. Scalable superconducting architecture for adiabatic quantum computation. *arXiv:quant-ph/0403090* (2004). URL http://arXiv.org/abs/quant-ph/0403090.
17. Johnson, M. W. *et al.* A scalable control system for a superconducting adiabatic quantum optimization processor. *Superconductor Science and Technology* **23**, 065004 (2010). URL http://stacks.iop.org/0953-2048/23/i=6/a=065004.
18. Bunyk, P. I. *et al.* Architectural considerations in the design of a superconducting quantum annealing processor. *IEEE Transactions on Applied Superconductivity* **24**, 1–10 (Aug. 2014).
19. Vinci, W. *et al.* Hearing the shape of the ising model with a programmable superconducting-flux annealer. *Sci. Rep.* **4** (2014). URL http://www.nature.com/articles/srep05703.
20. O'Gorman, B., Babbush, R., Perdomo-Ortiz, A., Aspuru-Guzik, A. & Smelyanskiy, V. Bayesian network structure learning using quantum annealing. *The European Physical Journal Special Topics* **224**, 163–188 (2015). URL http://dx.doi.org/10.1140/epjst/e2015-02349-9.
21. Rieffel, E. G. *et al.* A case study in programming a quantum annealer for hard operational planning problems. *Quantum Information Processing* **14**, 1–36 (2015). URL http://dx.doi.org/10.1007/s11128-014-0892-x.
22. Adachi, S. H. & Henderson, M. P. Application of quantum annealing to training of deep neural networks. *arXiv:1510.06356* (2015). URL http://arXiv.org/abs/1510.06356.
23. Amin, M. H., Andriyash, E., Rolfe, J., Kulchytskyy, B. & Melko, R. Quantum boltzmann machine. *arXiv:1601.02036* (2016). URL https://arxiv.org/abs/1601.02036.
24. Benedetti, M., Realpe-Gómez, J., Biswas, R. & Perdomo-Ortiz, A. Estimation of effective temperatures in quantum annealers for sampling applications: A case study with possible applications in deep learning. *Physical*

24. Benedetti, M., Realpe-Gómez, J., Biswas, R. & Perdomo-Ortiz, A. Estimation of effective temperatures in quantum annealers for sampling applications: A case study with possible applications in deep learning. *Physical Review A* **94**, 022308– (2016). URL http://link.aps.org/doi/10.1103/PhysRevA.94.022308.

25. Benedetti, M., Realpe-Gómez, J., Biswas, R. & Perdomo-Ortiz, A. Quantum-assisted learning of graphical models with arbitrary pairwise connectivity. *arXiv:1609.02542* (2016). URL http://arXiv.org/abs/1609.02542.

26. Boyda, E. *et al.* Deploying a quantum annealing processor to detect tree cover in aerial imagery of california. *PLOS ONE* **12**, e0172505– (2017). URL http://dx.doi.org/10.1371%2Fjournal.pone.0172505.

27. Boixo, S. *et al.* Evidence for quantum annealing with more than one hundred qubits. *Nat. Phys.* **10**, 218–224 (2014).

28. Shin, S. W., Smith, G., Smolin, J. A. & Vazirani, U. How "quantum" is the D-Wave machine? *arXiv:1401.7087* (2014). URL http://arXiv.org/abs/1401.7087.

29. Lanting, T. *et al.* Entanglement in a quantum annealing processor. *Phys. Rev. X* **4**, 021041– (2014).

30. Albash, T., Vinci, W., Mishra, A., Warburton, P. A. & Lidar, D. A. Consistency tests of classical and quantum models for a quantum annealer. *Phys. Rev. A* **91**, 042314– (2015). URL http://link.aps.org/doi/10.1103/PhysRevA.91.042314.

31. Boixo, S. *et al.* Computational multiqubit tunnelling in programmable quantum annealers. *Nat Commun* **7** (2016). URL http://dx.doi.org/10.1038/ncomms10327.

32. Rønnow, T. F. *et al.* Defining and detecting quantum speedup. *Science* **345**, 420–424 (2014). URL http://science.sciencemag.org/content/345/6195/420.

33. Hen, I. *et al.* Probing for quantum speedup in spin-glass problems with planted solutions. *Phys. Rev. A* **92**, 042325– (2015). URL http://link.aps.org/doi/10.1103/PhysRevA.92.042325.

34. King, A. D., Lanting, T. & Harris, R. Performance of a quantum annealer on range-limited constraint satisfaction problems. *arXiv:1502.02098* (2015). URL http://arXiv.org/abs/1502.02098.

35. Katzgraber, H. G., Hamze, F., Zhu, Z., Ochoa, A. J. & Munoz-Bauza, H. Seeking quantum speedup through spin glasses: The good, the bad, and the ugly. *Phys. Rev. X* **5**, 031026– (2015). URL http://link.aps.org/doi/10.1103/PhysRevX.5.031026.

36. Chollet, F. Keras. https://github.com/fchollet/keras (2015).

37. Al-Rfou, R. *et al.* Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints* (2016). URL http://arxiv.org/abs/1605.02688.

38. Kingma, D. P. & Ba, J. Adam: A method for stochastic optimization. *CoRR* **abs/1412.6980** (2014). URL http://arxiv.org/abs/1412.6980.

39. Snoek, J., Larochelle, H. & Adams, R. P. Practical Bayesian Optimization of Machine Learning Algorithms. *ArXiv e-prints* (2012). 1206.2944.

40. Snoek, J. Spearmint. https://github.com/HIPS/Spearmint (2012).

41. Chen, T. & Guestrin, C. Xgboost: A scalable tree boosting system. *arXiv:1603.02754* (2016). URL https://arxiv.org/abs/1603.02754.

42. Hanley, J. A. & McNeil, B. J. The meaning and use of the area under a receiver operating characteristic (roc) curve. *Radiology* **143**, 29–36 (1982).

43. Kato, T. On the adiabatic theorem of quantum mechanics. *J. Phys. Soc. Jap.* **5**, 435 (1950).

44. Farhi, E., Goldstone, J., Gutmann, S. & Sipser, M. Quantum Computation by Adiabatic Evolution. *arXiv:quant-ph/0001106* (2000). URL http://arxiv.org/abs/quant-ph/0001106.

45. Choi, V. Minor-embedding in adiabatic quantum computation: I. The parameter setting problem. *Quant. Inf. Proc.* **7**, 193–209 (2008). URL dx.doi.org/10.1007/s11128-008-0082-9.

46. Choi, V. Minor-embedding in adiabatic quantum computation: II. Minor-universal graph design. *Quant. Inf. Proc.* **10**, 343–353 (2011). URL dx.doi.org/10.1007/s11128-010-0200-3.

47. Venturelli, D. *et al.* Quantum optimization of fully connected spin glasses. *Phys. Rev. X* **5**, 031040– (2015). URL http://link.aps.org/doi/10.1103/PhysRevX.5.031040.

48. Vinci, W., Albash, T., Paz-Silva, G., Hen, I. & Lidar, D. A. Quantum annealing correction with minor embedding. *Phys. Rev. A* **92**, 042310– (2015). URL http://link.aps.org/doi/10.1103/PhysRevA.92.042310.

49. Mishra, A., Albash, T. & Lidar, D. A. Performance of two different quantum annealing correction codes. *Quant. Inf. Proc.* **15**, 609–636 (2015). URL http://dx.doi.org/10.1007/s11128-015-1201-z.

50. Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H. & Teller, E. Equation of state calculations by fast computing machines. *The Journal of Chemical Physics* **21**, 1087–1092 (1953). URL http://link.aip.org/link/?JCP/21/1087/1.