AFRL-RI-RS-TR-2020-222

STRONG-CLASSIFIER QUANTUM ANNEALING DEVICE (SQUAD)

BOSTON FUSION CORP.

DECEMBER 2020

FINAL TECHNICAL REPORT

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

STINFO COPY

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE

■ **AIR FORCE MATERIEL COMMAND** ■ **UNITED STATES AIR FORCE** ■ **ROME, NY 13441**

NOTICE AND SIGNATURE PAGE

Using Government drawings, specifications, or other data included in this document for any purpose other than Government procurement does not in any way obligate the U.S. Government. The fact that the Government formulated or supplied the drawings, specifications, or other data does not license the holder or any other person or corporation; or convey any rights or permission to manufacture, use, or sell any patented invention that may relate to them.

This report was cleared for public release by the 88th ABW, Wright-Patterson AFB Public Affairs Office and is available to the general public, including foreign nationals. Copies may be obtained from the Defense Technical Information Center (DTIC) (http://www.dtic.mil).

AFRL-RI-RS-TR-2020-222 HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION IN ACCORDANCE WITH ASSIGNED DISTRIBUTION STATEMENT.

FOR THE CHIEF ENGINEER:

/ S /
KRISTI T. MEZZANO
Work Unit Manager

/ S /
GREGORY J. HADYNSKI
Assistant Technical Advisor
Computing & Communications Division
Information Directorate

This report is published in the interest of scientific and technical information exchange, and its publication does not constitute the Government's approval or disapproval of its ideas or findings.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
DECEMBER 2020	FINAL TECHNICAL REPORT	SEP 2018 – JUL 2020

4. TITLE AND SUBTITLE

STRONG-CLASSIFIER QUANTUM ANNEALING DEVICE (SQUAD)

5a. CONTRACT NUMBER
FA8750-18-C-0166

5b. GRANT NUMBER
N/A

5c. PROGRAM ELEMENT NUMBER
62788F

6. AUTHOR(S)

Andrew Spisak
Bill Thistleton
Eduardo Barerra
Paul Hemphill

5d. PROJECT NUMBER
CYDT

5e. TASK NUMBER
BO

5f. WORK UNIT NUMBER
ST

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Boston Fusion Corp.
70 Westview St., Suite 100
Lexington MA 02421

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Air Force Research Laboratory/RITQ
525 Brooks Road
Rome NY 13441-4505

10. SPONSOR/MONITOR'S ACRONYM(S)
AFRL/RI

11. SPONSOR/MONITOR'S REPORT NUMBER
AFRL-RI-RS-TR-2020-222

12. DISTRIBUTION AVAILABILITY STATEMENT

Approved for Public Release; Distribution Unlimited. PA# AFRL-2020-0581
Date Cleared: Dec 22, 2020

13. SUPPLEMENTARY NOTES

14. ABSTRACT

Strong-classifier Quantum Annealing Device (SQUAD) is a system that creates a strong classifier from thousands of base classifiers by developing embedding-less implementations on an optimized D-Wave 2000Q machine. The SQUAD Final Report documents the design of our classifier system and discusses our results and analysis.

15. SUBJECT TERMS
Quantum Computing, Quantum Information Science, Coherent Ising Machines, Quantum Algorithms, Quantum Annealing

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			**KRISTI T. MEZZANO**
U	U	U	UU	30	19b. TELEPHONE NUMBER (*Include area code*) N/A

CONTENTS

LIST OF FIGURES

1 SUMMARY

Quantum computing can theoretically provide exponential speed-up for machine learning (ML) algorithms. In the last few years, small-scale quantum annealing devices comprising 2,000 or more qubits (such as the D-Wave 2000Q) have been developed that can be used to find solutions to high-dimensional NP-hard optimization problems. Initial proof-of-concept implementations of these NP-hard optimization problems on smaller quantum annealing (QA) devices for practical ML problems have shown inconclusive advantages of quantum algorithms over classical algorithms. One such problem is that of creating an ensemble of classifiers. Implementations of ensemble classifiers on small QA devices has been hampered by the overhead associated with computing a graph minor embedding, as well as the lack of optimization of the control parameters of the devices [1, 2].

The objective of the Strong-classifier QUantum Annealing Device (SQUAD) program is to create a strong classifier from hundreds of weak base classifiers on an optimized D-Wave 2000Q machine. We systematically combine multiple proposed techniques for D-Wave performance enhancement. This report documents the design of our classifier system, as well as our results and analysis.

2 INTRODUCTION

The SQUAD ensemble classifier consists of a series of base classifiers produced via classical computation, paired with binary weights computed on the D-Wave quantum annealer. The binary weights identify the specific base classifiers to include as voting members of the final ensemble result (Figure 1). The process for computing the base classifiers, weights, and final strong classifier result is detailed in the sections to follow.

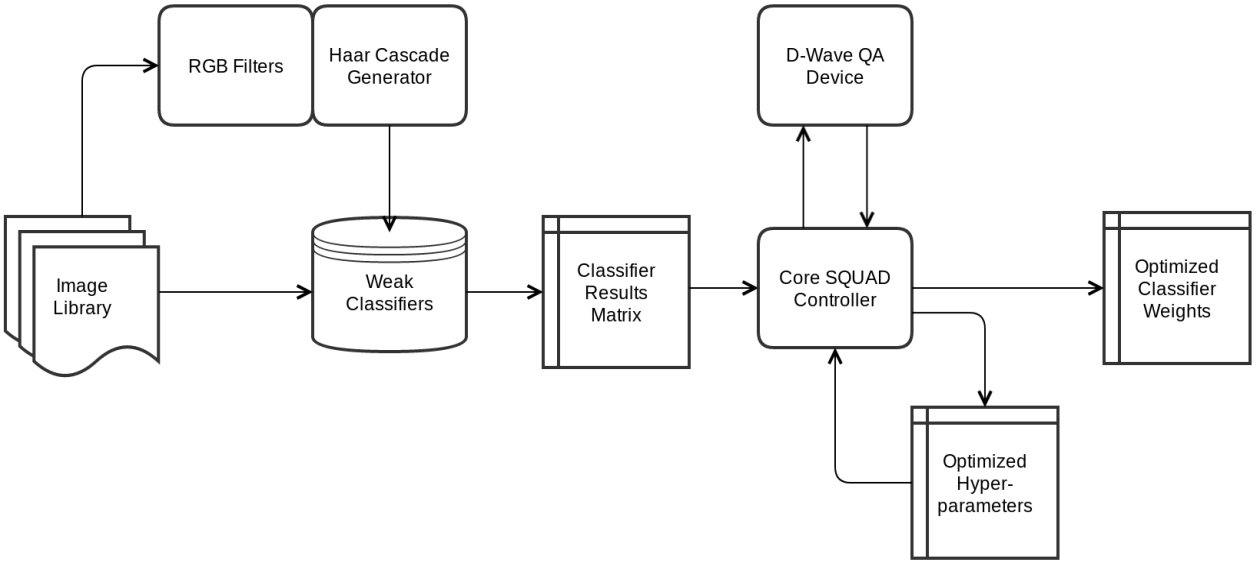

Figure 1: The SQUAD Quantum Ensemble Classifier Architecture

3 METHODS, ASSUMPTIONS, AND PROCEDURES

3.1 ENSEMBLE CLASSIFIER ARCHITECTURE

Ensemble methods seek to build a classifier by combining the insights of several weak base classifiers so that the aggregate result is more accurate and generalizable [3, 4].

We start with an ensemble of N base classifiers and build a table of individual classification outputs for a test or evaluation library of T images. A weak classifier can be any algorithm or model that outputs a classification judgment for an image, and generally does so better than random chance, i.e., with accuracy values between 50 and 60%, such that it is easy to generate many examples without extensive training data or optimization. (Section 3.3 describes our approach to base classifier generation.) We structure the matrix of classifier results, $\boldsymbol{C}$, such that each column represents the outputs of a single classifier (applied to the full library of images), and each row represents the results of a single image (evaluated by the full set of classifiers):

$$\boldsymbol{C} \equiv \begin{bmatrix} c_1(t_1) & c_2(t_1) & \cdots & c_N(t_1) \\ c_1(t_2) & c_2(t_2) & & c_N(t_2) \\ & \vdots & & \\ c_1(t_T) & c_2(t_T) & & c_N(t_T) \end{bmatrix}. \tag{1}$$

An ensemble classifier seeks to apply optimal weights w_j to the set of classifiers to obtain a single weighted sum. In general, these weights may be anywhere on the range $w_j \in [0,1]$ (as with, e.g., regression solutions with constrained coefficients); for QA solutions such as the ensemble classifier discussed here, the weights are restricted to the binary option $w_j \in \{0,1\}$ (for more detail see Section 3.2). The output $R(t_i)$ of the ensemble classifier for an image t_i is then

$$R(t_i) = sign\left(\sum_{j=1}^{N} w_j c_j(t_i)\right), \tag{2}$$

where the values c_j from $\boldsymbol{C}$ have been scaled according to

$$-\frac{1}{N} \leq c_j(t_i) \leq \frac{1}{N} \tag{3}$$

so that

$$-1 \leq \sum_{j=1}^{N} w_j c_j(t_i) \leq 1. \tag{4}$$

If the individual base classifiers are sufficiently uncorrelated, the performance of the strong classifier $R(t)$ is expected to be higher than any individual base classifier, both in terms of accuracy and generalizability. The problem becomes the computation of the optimal weight set w_j, which for fully-connected problems is NP-hard.

3.2 QUANTUM ANNEALING FORMALISMS

3.2.1 The QUBO Loss Function

In general, we follow the QBOOST approach [5]. Obtaining a solution via QA methods requires casting the problem as a quadratic unconstrained binary optimization (QUBO) problem. The standard QUBO loss function is usually written as [2]

$$L = \sum_{\tau} \left(y_\tau - R(\vec{x}_\tau)\right)^2, \tag{5}$$

where τ indexes images to be classified, $\vec{x}_\tau$ is a vector of features representing image τ, y_τ is the label for that image, and $R(\vec{x}_\tau)$ is the strong classifier function.

Expanding this loss function in terms of the base classifiers $c_i(\vec{x}_\tau)$ and the weights w_i gives the QUBO minimization problem as

$$\vec{w}_{\min} = \operatorname{argmin}\left[\sum_{i,j} \mathrm{C}_{ij} w_i w_j - 2\sum_{i} \mathrm{C}_i w_i\right]. \tag{6}$$

Here $\mathrm{C}_{ij} = \sum_\tau c_i(\vec{x}_\tau) c_j(\vec{x}_\tau)$ is the interaction term and $\mathrm{C}_i = \sum_\tau c_i(\vec{x}_\tau) y_\tau$ is the self-term. To solve this QUBO on a QA device, we must first convert it to an Ising problem, which is an equivalent formulation described directly in terms of the Hamiltonian of the quantum system [2]. The D-Wave Ocean software toolset[1] includes routines for converting problems back and forth between QUBO and Ising descriptions, and the SQUAD core module makes use of these routines.

In the Ising formalism, the above equation instead has the form

$$s_g = \operatorname{argmin}\left[-\sum_{i,j} \mathrm{J}_{ij} s_i s_j - \sum_{i} h_i s_i\right], \tag{7}$$

where J_{ij} and h_i are transformed versions of C_{ij} and C_i, respectively, and the $s_j \in \{-1,1\}$ are transformed versions of the weights w_i, now given physical interpretation as the spin states of individual qubits. The solution s_g is the ground state found by the QA system for the Hamiltonian defined by J_{ij} and h_i.

[1] https://ocean.dwavesys.com/

3.2.2 Embeddings and Coupling Rescaling

Solving the fully-connected Ising Hamiltonian requires a fully-connected physical qubit architecture. Due to physical limitations in geometry and manufacturing, the connectivity of the qubits of the D-Wave QA systems is given by a graph architecture known as a Chimera (Figure 2).

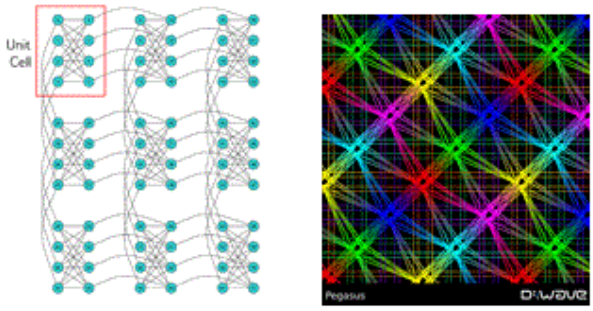

Figure 2: D-Wave Chimera connectivity architecture (left) utilized in this study and the more densely connected Pegasus architecture (right), courtesy of D-Wave Systems

Various embedding schemes have been used to compute a fully-connected embedding for a given problem within a Chimera architecture by linking multiple physical qubits into single logical qubits [6, 7]. The current D-Wave software provides a minor embedding utility to map an upper triangular QUBO onto the correct Ising model with appropriate connections. Newer architectures with enhanced connectivity are becoming available [8, 9]. Embedding may work for intermediate size problems (N<65 currently), but the process is slow and many qubits are lost to the embedding. We solve problems with at least 100 base classifiers, too large to compute a fully-connected graph embedding.

To obtain an embedding-less QUBO or Ising Hamiltonian, we must only include interaction terms C_{ij} that correspond to a physical connection in the chimera graph of the physical D-Wave qubits. We can achieve this by writing the interaction term as

$$\sum_i c_i w_i \left[\sum_{j \in G_i} c_j w_j + \sum_{j \notin G_i} c_j w_j \right], \tag{8}$$

with G_i representing the set of qubits physically connected to qubit i. The first term can be computed normally, as the physical connection between qubits allows a coupling to be set in the Hamiltonian of the D-Wave device, while the second term cannot be computed directly in an embedding-less architecture and must be approximated.

3.2.3 Regularization

One common issue with ensemble classifiers is that of overfitting; selecting too many base classifiers often leads to marginal improvements in accuracy on a test dataset at the cost of a loss of ability to generalize to unseen data. Including a hyperparameter regularization term λ produces a model with fewer nonzero weights by penalizing the loss function based on

the count of nonzero weights. This counters the tendency towards overfitting the training data. Hence, our loss function becomes

$$\vec{w}_{\min} = \operatorname{argmin}\left\{\sum_{i=1}^{T}\left(\sum_{j=1}^{N} y(t) - w_j c_j(t_i)\right)^2 + \lambda \sum_{j=1}^{N} w_j\right\} \tag{9}$$

and λ becomes a tunable hyperparameter to be optimized (see Section 3.4).

3.3 Base Classifier Generation

We selected as the exemplar dataset PlanesNet Version 9.0[2] due to its availability, simplicity, and potential for Air Force relevant interest. PlanesNet consists of 32,000 20 pixel by 20 pixel overhead images classified as either containing or not containing a plane, and a JSON-formatted file containing image names, labels, scene identifications, and location metadata. Figure 3 shows example images from the PlanesNet dataset.

To create a set of base classifiers, we used OpenCV[3] to train a set of Haar image cascade classifiers [10]. We used the Gentle AdaBoost algorithm for the Haar cascade boost, which is the OpenCV default.We restricted the maximum depth of the base decision trees to be 1, which is the case of "stumps." To create a large number of base classifiers, we generated new feature sets from the training images by filtering the images into red (R), green (G), blue (B), and grayscale channels.

OpenCV outputs the trained cascade classifier as an .xml file containing a set of decision trees and Haar features (rectangle boundaries). We wrote a custom .xml parser to extract the stumps and write individual .xml files for each base classifier.

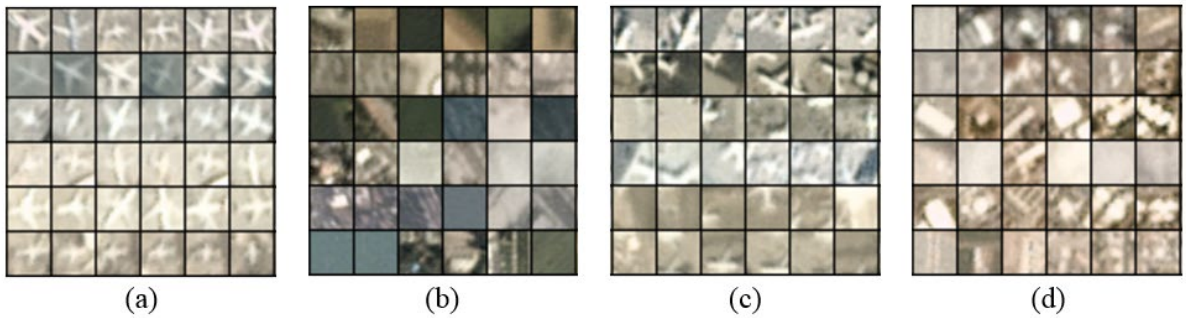

Figure 3: Example images from the PlanesNet dataset, labeled as follows: (a) "plane", (b) "no-plane", (c) "partial plane", and (d) "no-plane" that are known to often be misclassified as "plane" by ML algorithms.

Generating individual base classifiers according to this process yielded a total of 240 base classifiers, more than the 65 required to exhaust the possibility of a fully-connected minor embedding. The accuracies of the full classifier set are given in Figure 4, where bars are colored according to the feature set used to generate the feature.

[2] https://www.kaggle.com/rhammell/planesnet
[3] https://opencv.org/

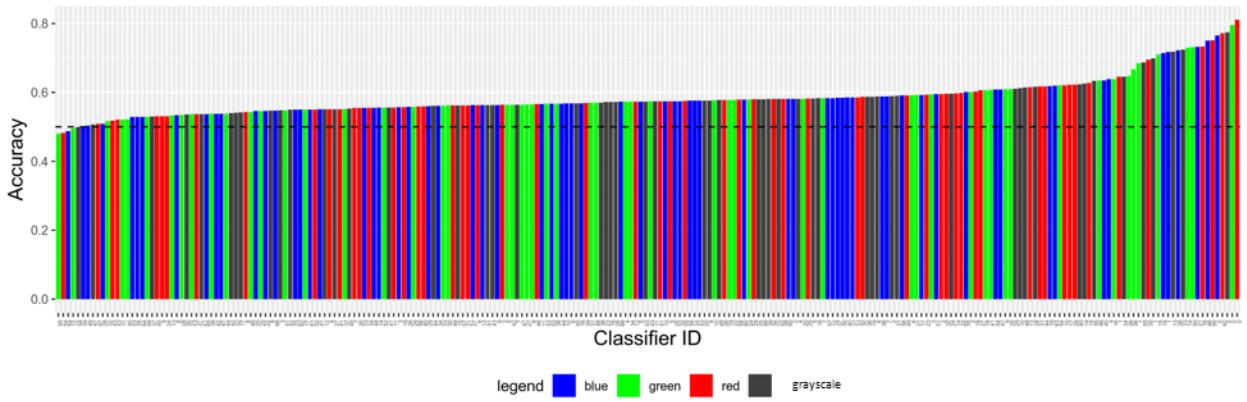

Figure 4: Sorted classifier accuracy, colored by generating feature. The dashed black line indicates 50% accuracy.

In addition to creating base classifiers for inputs to the quantum strong classifier (QSC), we created classical strong classifiers (CSCs) for comparison and evaluation of the QSC. To date, we have trained and optimized two CSCs, a deep convolutional neural network (CNN) with 98% accuracy on the data set, and the Haar cascade classifier [10] used to generate the base classifiers, which achieves 90% accuracy.

3.4 Computing Weights and Optimizations on a Quantum Annealer

3.4.1 Obtaining Weights

With the base classifier results matrix $\boldsymbol{C}$ defined as in Section 3, The SQUAD core uses the NumPy[4] library in Python to compute elements of a QUBO matrix Q as

$$Q = \boldsymbol{C}^T\boldsymbol{C} + \text{diag}(\lambda) - \frac{2}{N}\,\text{diag}(\boldsymbol{y}\cdot\boldsymbol{C}) \tag{10}$$

Lower triangular $(i > j)$ elements are set to 0 and upper triangular $(i < j)$ elements are defined as

$$Q_{i,j} = (\boldsymbol{C}^T\boldsymbol{C})_{i,j} + (\boldsymbol{C}^T\boldsymbol{C})_{j,j} \tag{11}$$

For submission to a D-Wave 2000Q quantum computer, we build the equivalent Ising model. Let $s_i \in \{-1,1\}$ and solve

$$\min_{s_1,s_2,\ldots s_n}\left(\sum_{i=1}^{n} h_i\cdot s_i + \sum_j\sum_{i<j} J_{i,j}\cdot s_i\cdot s_j\right). \tag{12}$$

Elements of the J matrix are stored as values in a Python dictionary with keys created as (i,j) tuples. Values are set as

$$J = \frac{1}{4}\cdot Q_{\text{upper}}\,. \tag{13}$$

[4] https://numpy.org/

In general, this approach leads to an Ising model where all the coupler weights $J_{i,j}$ are nonzero.

The h values are computed as

$$\boldsymbol{h} = -\left(-\text{diag}\left(\frac{\lambda}{2}\right) + \boldsymbol{C}\cdot\boldsymbol{y} - \frac{1}{2} * \boldsymbol{C}^{\boldsymbol{T}}\boldsymbol{C}\cdot\mathbf{1}\right). \quad (14)$$

Couplers are used if they are readily available in the Chimera architecture, and elements of the J matrix are discarded if they are not available.

Our approach for adapting a problem to the sparsely connected architectures include dropping most coupling terms but increasing the importance of those couplings which remain by using the scaling hyperparameter α (Section 3.4.3). We treat regularization (λ) and scaling (α) as hyperparameters and set them outside of the optimization call. This process is handled by the core SQUAD controller module, which takes in the base classifier results matrix as a stored NumPy array, creates the Q matrix, authenticates a live connection to a D-Wave device, and submits the problem for a solution.

We have found that calls to the D-Wave 2000Q quantum computer are sometimes lost (either to communication issues or an intermittent failure to embed), so we compute the minimization within an error-handling wrapper that only computes a quantum solution if the embedding is fulfilled, resubmitting any failed embeddings.

3.4.2 Optimizing Regularization Hyperparameter λ

As is typical when developing a general model from a specific data set, we introduce a penalty term to guard against overfitting. The number of base classifiers retained in the ensemble classifier is regulated with a parameter λ, introduced into the loss function with the L0-norm term [11]

$$\lambda\|w\|_0 = \lambda\sum_{j=1}^{N} w_j\,. \quad (15)$$

To determine optimal values of λ for the ensemble classifier, we set the parameter to values on a range and obtain a solution on the quantum device for each such value. We then select the optimal solution, thereby optimizing λ.

3.4.3 Optimizing Scaling Hyperparameter α

The quadratic loss approach to selecting an optimal ensemble of base classifiers results in a fully connected set of couplings and corresponding nonzero weights J_{ij}. Systems with such long range interactions require all-to-all connectivity to solve [12]. The chimera architecture employed in this study has only local connectivity and is severely restricted in the number of available connections, forcing the great majority of the weights to be zero. For smaller problems (N<65 in our simulations) an embedding may be found which is logically equivalent to the original, allowing for solution on the quantum device. For larger problems such an embedding is beyond the capabilities of the architecture. We follow other authors (e.g., [1], [2]) and perform a very simple embedding which sets to zero all connections not immediately found in the available

architecture. To compensate for the lost couplings, we increase the importance of the retained couplings in comparison with the local field strengths by multiplying the J_{ij} by a scale factor α. Similarly to the selection of regulation term λ, we determine optimal values of α in the validation part of the train-validate-test cycle.

4 RESULTS AND DISCUSSION

4.1 ADVERSARIAL DATASET DESIGN AND GENERATION

4.1.1 Background

One of the applications we explored in SQUAD is using the strong QA classifier to achieve improved performance in a case for which state-of-the-art image classification methods, such as CNNs, fail. The case we have chosen is that of an adversarial attack on the CNN detector trained on our dataset, which for non-adversarial images has a 98% accuracy. An adversarial image is one which has been specifically altered to fool a classifier, despite being an otherwise straightforward exemplar of a class.

4.1.2 Generating Adversarial Dataset

One of the most popular methods of generating adversarial datasets for neural network classifiers is the Fast Gradient Sign Method (FGSM) [13]. This attack relies on access to the network architecture and weights, which makes the method remarkably powerful. The attack is leveraged by reading the gradient data of a classifier's input, and using that gradient to apply a perturbation to the input data that will artificially maximize the loss of the perturbed image.

The perturbation acts as an added image tensor to the original input image, and can be varied in strength using a multiplicative constant ϵ. The attack can attempt to cause the resulting image to be misclassified in two ways, called *targeted* and *untargeted*. If an input image is of class A, a targeted attack will aim to convince the classifier that the image belongs to a specific class B instead. By contrast, an untargeted attack simply attempts to convince the classifier that the image is NOT A. Since PlanesNet dataset is a binary dataset, there is no distinction between targeted and untargeted attacks.

Using our validation set as input, we leveraged in-house graphics processing unit (GPU) servers to generate 1.6M adversarial images at ϵ values ranging from 1.1% to 98.9%.

4.1.3 Detecting Adversarial Attacks

We use this adverserial dataset to evaluate the performance of the strong QA classifier in the adversarial regime. To do so, we first select the subset of images for which CNN performance is a minimum—this turns out to be ϵ values around 8% (see Figure 5). Then, we evaluate the base classifier output on this dataset, creating a new set of $\boldsymbol{C}$ matrices for analysis. With these output matrices, we can perform two complementary analyses. In one we use the adverserial $\boldsymbol{C}$ matrices to compute the defining weights for

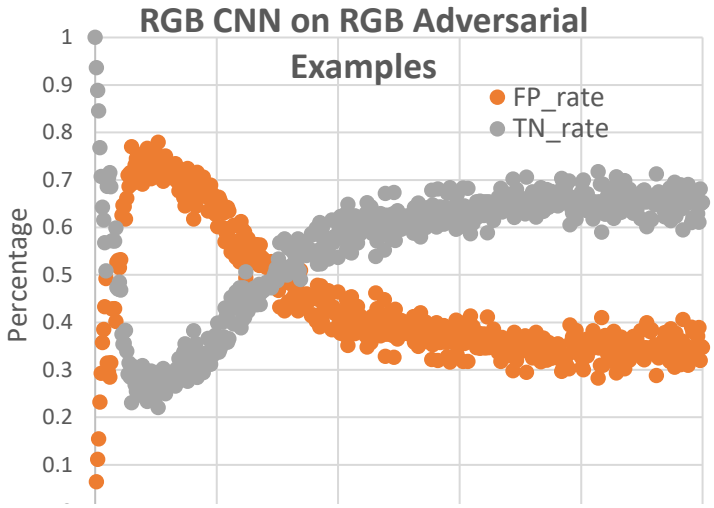

Figure 5: CNN False Positive (orange) and True Negative (gray) rates by epsilon value (perturbation strength)

base classifiers, following the procedure defined in Section 3.4.1, and measure the performance on the unperturbed dataset. In the other, we use the strong QA classifier with weights obtained from optimizing over the non-adversarial results, and measure the performance of that weight set on the adversarial images.

Finally, using the analysis results from testing the strong cascade classifiers and CNN against the perturbed datasets, we develop a method to flag CNN input as potentially adversarial. The method consists of using three non-similar strong cascade classifiers alongside a CNN and running an input across all our classifiers. If the majority of the strong cascade classifiers vote differently than our CNN, we flag the input as adversarial. We are able to do this since CNNs have high accuracy but low resistance to adversarial attacks, and strong cascade classifiers have accuracy that is not comparable to the CNN, but excellent resistance to adversarial attacks.

In addition to the original cascade classifier, we used OpenCV to train an additional two classifiers with different hyperparameters from the original so that we would have three dissimilar but equal-performance classifiers. Using these three well-trained and dissimilar Haar cascade classifiers we built an adversarial attack detector (see Figure 6). The purpose of adversarial attack detector is to determine whether an input to the CNN was designed to give an intentional misclassification.

The detection method is based on a few key observations:

- CNNs have very high potential max accuracy, but are very susceptible to small changes in their input image tensors—meaning small changes to an input image of a given class with random-pixel noise perturbations that are indiscernible to the human eye can easily fool a CNN's internal calculations leading to an incorrect classification.
- Haar Cascade classifiers have lower potential max accuracy, but are very robust against non-feature changes in their input images—meaning small changes to an input image of a given class with random-pixel noise perturbations that are indiscernible to the human eye do not generally affect with the overall classification accuracy too drastically.

The adversarial detector uses the following algorithm:

The detector is split into two groups of classifiers, where group 1 is composed solely of the CNN and group 2 is composed of the three Haar Cascade classifiers. An input image is then passed through each of our classifier groups and each group is given a binary attribute for the input image in question—1 if the classification was correct, 0 otherwise.

If the CNN agrees with the Haar Cascade group, then the input image is likely to have been correctly classified and not an adversarial attack. If there is disagreement between the two groups we have two possibilities: the CNN has correctly classified the input image while the Haar classifiers have not; or the CNN incorrectly classifies the image and the Haar classifiers agree. The final case is that in which both our classifier groups are fooled, which we label an undetected attack that eluded our detector.

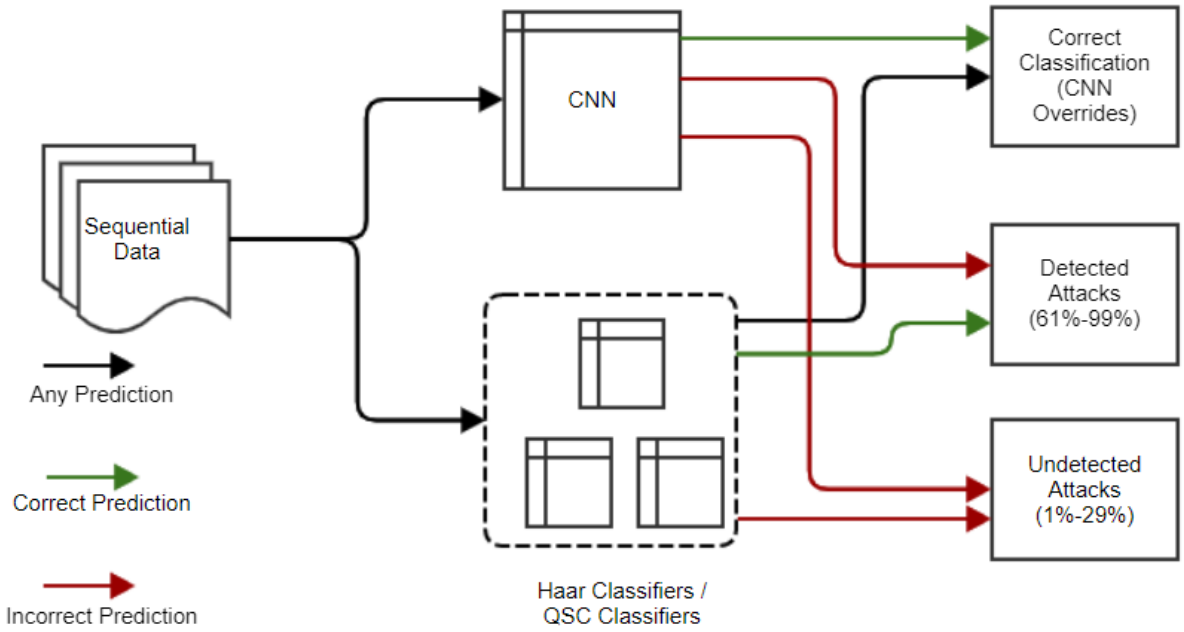

Figure 6: System diagram for detecting adversarial attacks

The adversarial attack detector's performance accuracy ranges from 62% to 99% for different attack perturbation strengths (see Figure 7); the case of 62% is when the adversarial attacks are optimally tuned to achieve the lowest possible performance accuracy for the CNN, and the 99% case is when input images are technically classified as adversarial attacks but do not fool the CNN often.

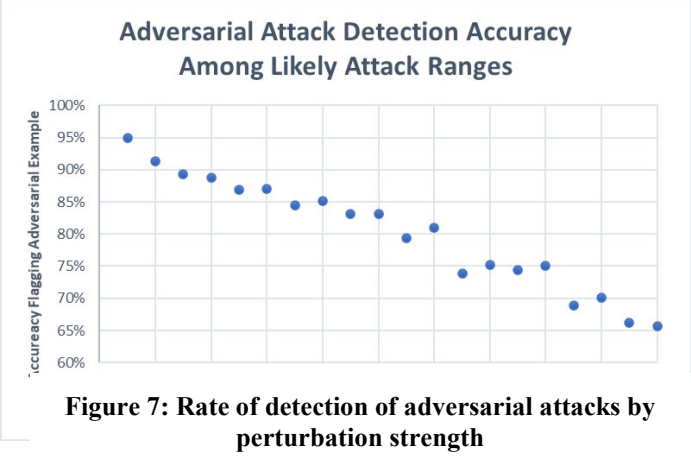

Figure 7: Rate of detection of adversarial attacks by perturbation strength

The adversarial attack is expected to perform just as well given a sequence of real-time images with real-time perturbations, such as would be the case during an adversary attempting to crash a self-driving vehicle or similar video-input situations.

We have demonstrated the potential for flagging adversarial inputs from mixed-technique systems such as these, including inputs from CNNs combined

with other quantum- and classical-based classifiers. Additional work is necessary to determine the correct tunings and highest-performing mix of classifiers.

4.2 Reverse Annealing Classifier Design

4.2.1 Motivation

A notable disadvantage of the D-Wave's quantum annealing process for SQUAD's ensemble classifier is that the qubits themselves are binary-valued. This restricts our ensemble classifer to using only the *predictions* of the weak classifiers, ignoring any information about the confidence of those predictions. The D-Wave's reverse annealing (RA) mode presents a promising avenue to incorporate this confidence information.

In reverse annealing, the D-Wave system is placed into some prepared state, with the spins set to known values (*cf.* the usual approach, when the qubits are initially in a quantum superposition of states). The annealing process is then run in reverse *(increasing* the transverse field, instead of decreasing it) to bring the spins into a quantum superposition of states, followed by another forward annealing run to bring the system back to the classical regime. Ordinarily, reverse annealing is used to pull the system out of a local minimum and make the optimization search more global.

With the D-Wave's control over the external magnetic field h_i applied to each qubit, we can also use RA to incorporate the confidence information from the weak classifiers. Having $h_i \neq 0$ modifies the i^{th} qubit's probability to be found in a given spin state after the RA process completes, as the qubit will tend to align to the external magnetic field. With some knowledge of how the specific value of h influences this probability, we can encode the confidence of the classifier in the magnetic field strength.

4.2.2 Reverse Annealing Classifier Design

4.2.2.1 Characterization of h vs. P_{flip} relationship

We carried out a series of experiments to characterize the spin-flip probability P_{flip} as a function of the magnetic field applied, h, and the annealing time, t_{RA}. In each experiment, the system was placed into an initial state with all qubits set to their -1 state, the external magnetic field was set to a uniform value across all qubits, and the reverse annealing process was run for a set period of time. In this manner, we scanned across h between -1.0 and 1.0 in steps of 0.1 and with annealing times between 3 and 12 μs. The results can be found in Figure 8. As expected, P_{flip} at $h = 0$ results in approximately 50% of the qubits reversing their spins. The probability that a given qubit reverses its spin during reverse annealing appears to be insensitive to the total

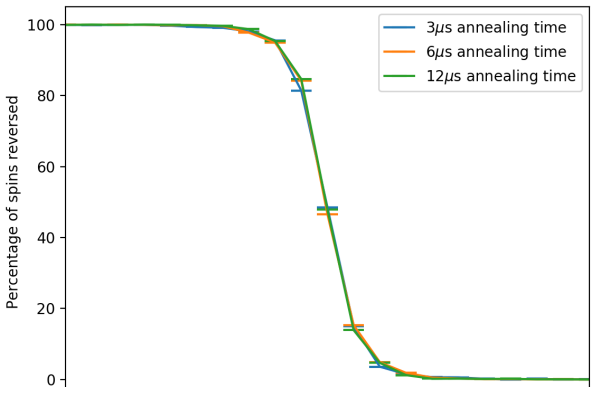

Figure 8: Percentage of qubits with reversed spins versus local magnetic field, for a variety of annealing times. Horizontal errors are rough estimates of the D-Wave's systematic and random uncertainty on h_i.

annealing time regardless of h. Further analysis is in progress to fully characterize this relationship.

4.2.2.2 Reverse annealing classifier

As can be seen in Section 4.2.1, if a strong magnetic field is applied, a qubit can have a strong resistance to changing its spin during the reverse annealing process. Conversely, with zero applied field and no couplings, the qubit will reverse its state half the time on average. When running a reverse annealing process on the outputs of an ensemble of base classifiers, we can thus lock in the results of the higher-confidence classifiers while allowing the lower-confidence classifiers to vary. Introducing couplings between classifiers then permits qubits to influence each other. The final total magnetization of the system is then interpreted as the total classifier output [14].

4.3 QUANTUM CLASSIFIER RESULTS AND ANALYSIS

4.3.1 Analysis of Base Classifiers and Potential Solutions

We consider three families of Adaboost base solvers: Gentle, Discrete, and Real. These correspond to base classifiers created according to Section 3.3 using different optimization algorithms (Gentle, Discrete, and Real, Adaboost) in OpenCV. We work within these families and to build strong classifiers by solving for the weights using the D-Wave according to Section 3.4 and comparing to classically-determined weights via various methods. We characterize the performance of these base solvers with both accuracy and Matthews Correlation when tested on classification of images from the PlanesNet data set. To study the effects of ensemble formation, we do not use all the base solvers, but rather select base solvers whose accuracies lie within a range of convenience. Base classifiers will all have accuracies exceeding 0.5 and below an upper threshold. Inclusion of base classifiers with arbitrarily high-performance leads to uninteresting ensembles dominated by small number of high-performing base classifiers. The performance of individual base classifiers for the three families are shown below.

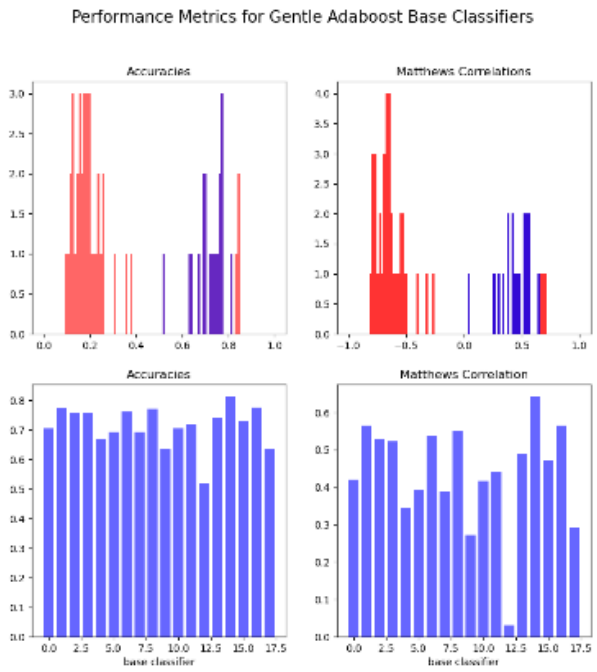

Figure 9: Base classifier properties for Gentle Adaboost classifiers. Top row: histograms of classifiers according to accuracy (left) and Matthews correlation (right). Classifiers called out in blue are selected for ensembles. Bottom row: Accuracy (left) and Matthews correlation (right) for selected base classifiers

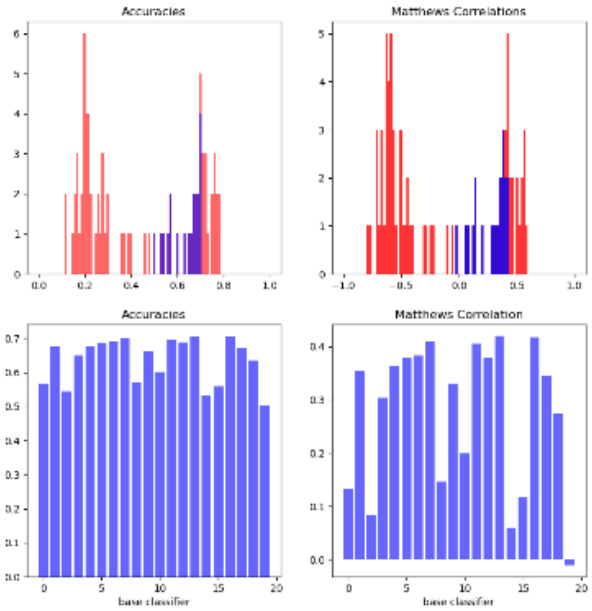

Figure 10: Base classifier properties for Discrete Adaboost classifiers. Top row: histograms of classifiers according to accuracy (left) and Matthews correlation (right). Classifiers called out in blue are selected for ensembles. Bottom row: Accuracy (left) and Matthews correlation (right) for selected base classifiers

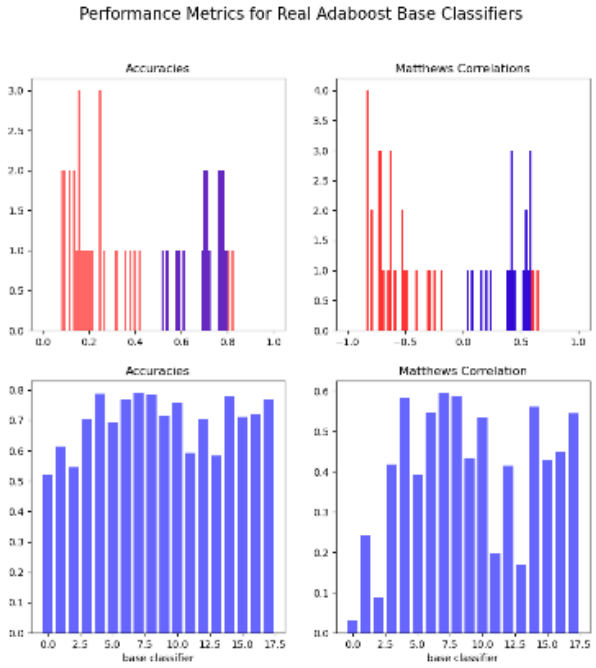

Figure 11: Base classifier properties for Real Adaboost classifiers. Top row: histograms of classifiers according to accuracy (left) and Matthews correlation (right). Classifiers called out in blue are selected for ensembles. Bottom row: Accuracy (left) and Matthews correlation (right) for selected base classifiers

For small numbers of base classifiers (<25 or so) we can run exact solvers which form all possible 2^n ensembles, giving via exhaustion an exact solution to compare with classical annealing and quantum annealing (QA) solutions. Table 1 presents the max and minimum possible accuracies for contstrained base classifier subsets; the minimum value represents the random chance floor for our balanced datset, while the maximum value provides a point of comparison for QA solutions on the constrained subset.

Table 1: Strong Classifier accuracy ranges for all possible solutions using a constrained base classifier set

	Gentle	Discrete	Real
Max Accuracy	86%	86%	88%
Min Accuracy	51%	50%	51%

Figure 12, Figure 13, and Figure 14 present the histograms of exact solutions for Gentle, Discrete, and Real Adaboost classifiers, respectively.

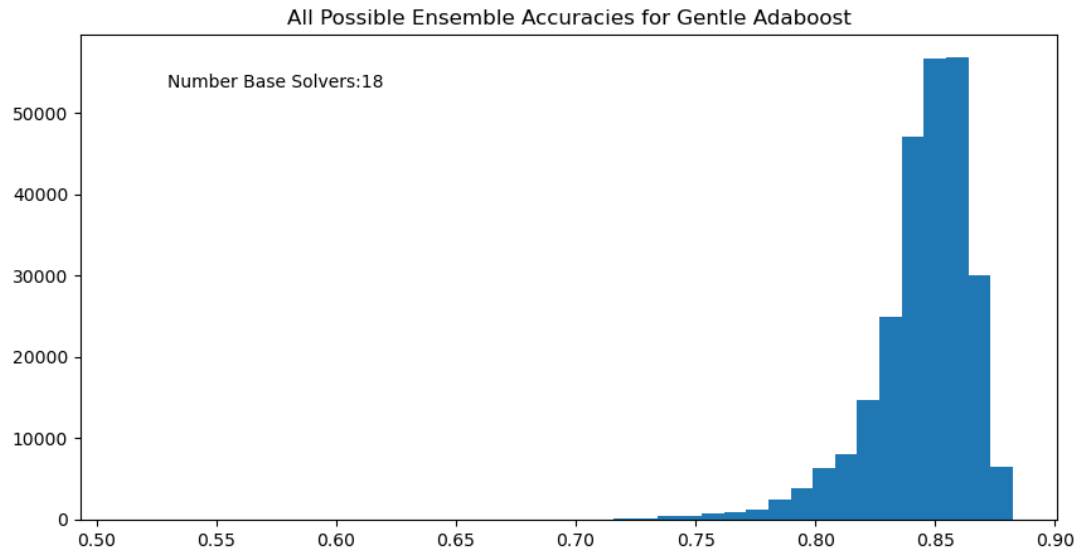

Figure 12: Histogram of accuracy results for all possible strong classifier ensembles with 18 base Gentle Adaboost classifiers

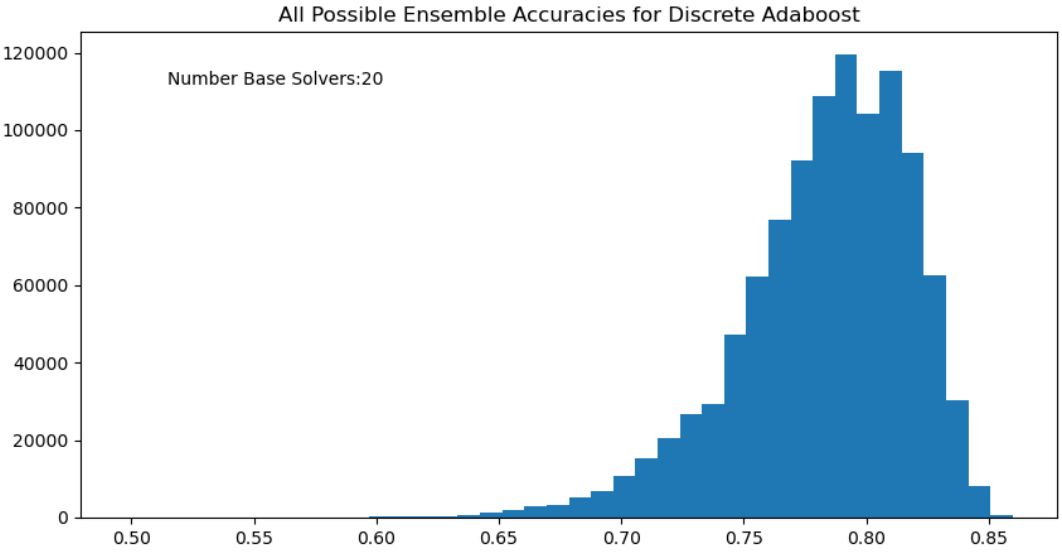

Figure 13: Histogram of accuracy results for all possible strong classifier ensembles with 20 base Discrete Adaboost classifiers

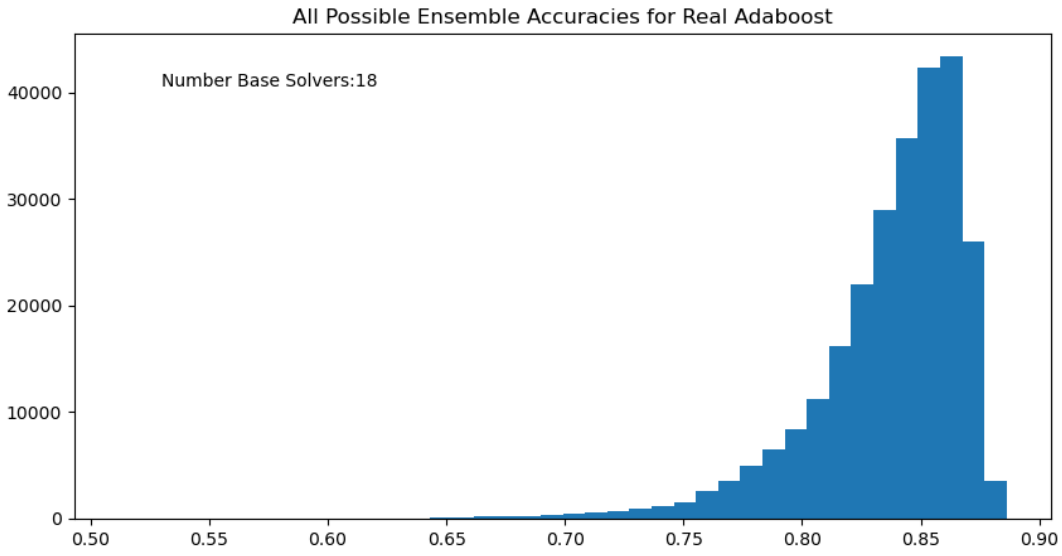

Figure 14: Histogram of accuracy results for all possible strong classifier ensembles with 18 base Real Adaboost classifiers

4.3.2 Results and Analysis

We note here that the QA device offers solutions that are stochastic in nature and that, while designed to be near optimal, there remains a good deal of variability in the lowest energy solution returned by a single run of the solver. We submitted the same problem to the D-wave QA device many times and recorded the lowest energy solution on each problem submission (Figure 15). The most frequently returned solution matches that of the `dimod` simulated annealing solver included in the D-Wave QA tools, though there is a tail of results that are worse. Of note is that neither solution performs as well as the exact solution provided via method of exhaustion; this phenomenon is problem-dependent and becomes irrelevant for optimizations with many base classifiers such that the method of exhaustion is intractable (precisely the regime in which QA can be expected to provide advantages over other solutions).

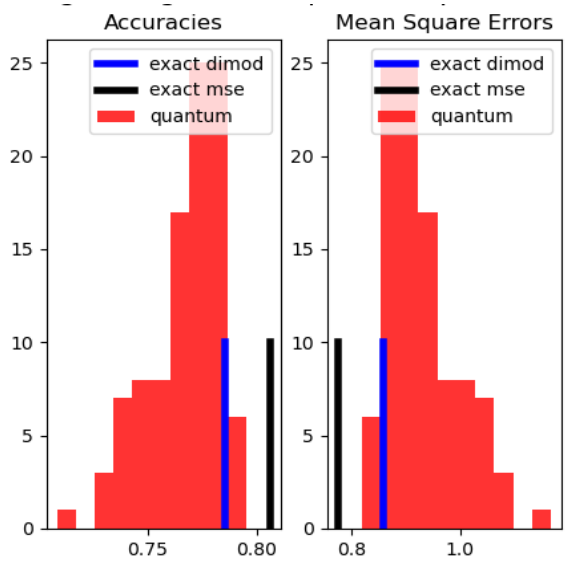

Figure 15: Histogram of lowest-energy solutions returned by multiple QA runs of the same problem. The blue bar represents the solution returned by the simulated annealing `dimod` package, while the black bar represents the exact best results, computed via method of exhaustion. Left panel reports accuracies, right panel reports the MSE.

4.3.2.1 Regularization

We studied the effect of the regularization constant λ on strong classifier performance for solutions directly embedded on the physical working graph of the device, and compared to classical solutions with the same Chimera graph structure of the D-wave. Increasing regularization has the effect of penalizing the inclusion of many base classifiers in the ensemble solution, forcing the solution to have fewer and fewer active base classifiers. The purpose of such a term is to reduce the effect of overfitting on the training data. Results are displayed in Figure 19 and Figure 17.

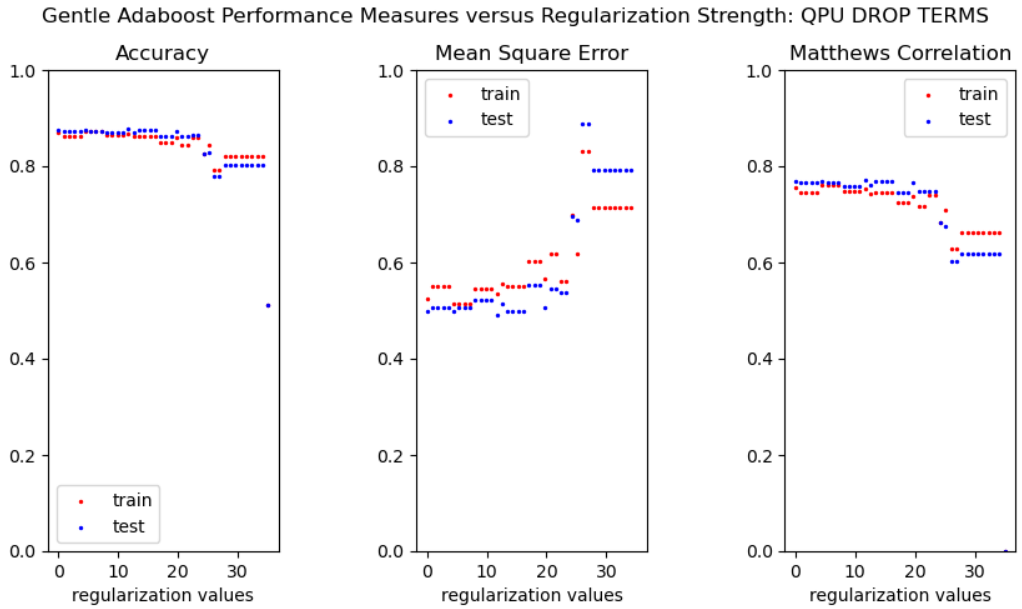

Figure 16: Performance metrics for strong classifier weights determined via direct embedding on the D-Wave QA device for increasing values of regularization.

For solutions obtained on the QA device (Figure 16), we do not observe an overtraining trend for low regularization values. Performance remains fairly constant across a range of values (suggesting the presence of many redundant/highly correlated base classifiers) until regularization increases enough to begin enacting a performance penalty, as useful information is discarded when marginal classifiers are removed from the solution set. At sufficiently high regularization, accuracy drops to nearly 50%, approximately the performance level of an arbitrary base classifier.

Figure 17 displays the same information for classical solutions using an embedding that matches that of the D-Wave Chimera graph.The trends are qualitatively similar—the quantum solution neither significantly lags nor significantly outperforms the classical methods. Performance against the training set versus against the test set is more similar for the classically-derived solutions than for the QA solutions, a trend that may indicate additional robustness to overfitting for solutions obtained via the inherently stochastic QA method.

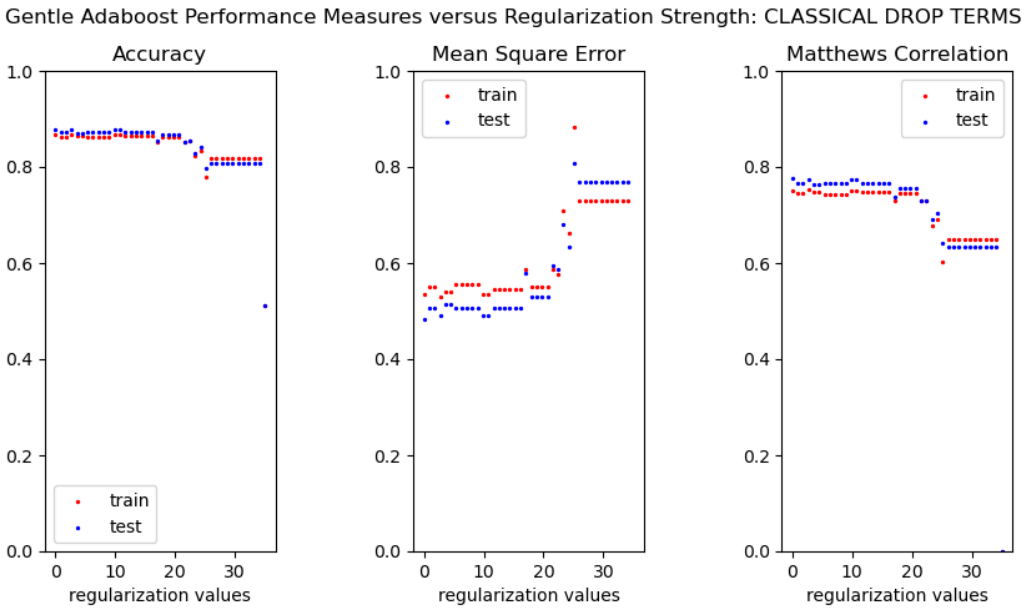

Figure 17: Performance metrics for strong classifier weights determined via classical solutions for a graph structure that matches the D-Wave Chimera embedding for increasing values of regularization.

4.3.2.2 Performance With and Against Adversarial Images

To determine if QA may provide specific advantages in regimes where stronger, more traditional methods (such as CNNs) fail, we computed a series of Quantum Strong Classifiers (QSCs) using base classifiers trained on real as well as adversarial input images. We wished to determine i) whether QSCs would be robust against image which fool a CNN, and ii) if composing an ensemble of base classifiers trained against adversarial images would improve classifications on real images.

We present these results in Figure 18, which displays an all-against-all performance summary across a range of CNN attack strengths from 0 (real data) to 0.016. Each column represents the evaluation output of a QSC whose weights were calculated for a different set of base classifiers; specifically, classifiers trained against adversarial images with the reported attack strength. The weights were computed on the QA using the direct Chimera embedding. Rows represent evaluation against a test set of specific attack strength.

Performance of our standard QSC against various adversarial attack strengths can be examined by considering the first column (for which the attack strength is 0.0). Notably, there is only minor variation across the range of attack strengths (cf. the CNN performance in Figure 5). Thus, the ensemble method is robust against adversarial attacks designed to fool a CNN. (See discussion in Section 4 for an examination of the potential applicability of this observation.)

By considering the uppermost row, we can examine whether any QSC composed of specific base classifiers trained at varying attack strengths performs exceptionally well against the real (non-adversarial) data. Notably, there is a QSC that performs better than the base QSC on real data, despite being trained against adversarial datasets. This QSC appears to represent an overall performance optimum across all test datasets. There is also QSC that performs poorly relative to other QSCs across all of the tests; intriguingly, this QSC sits at an attack strength value close to

the minimum of the CNN performance. It is important to note that the variation in performance is not large in absolute sense; about 3 percentage points total.

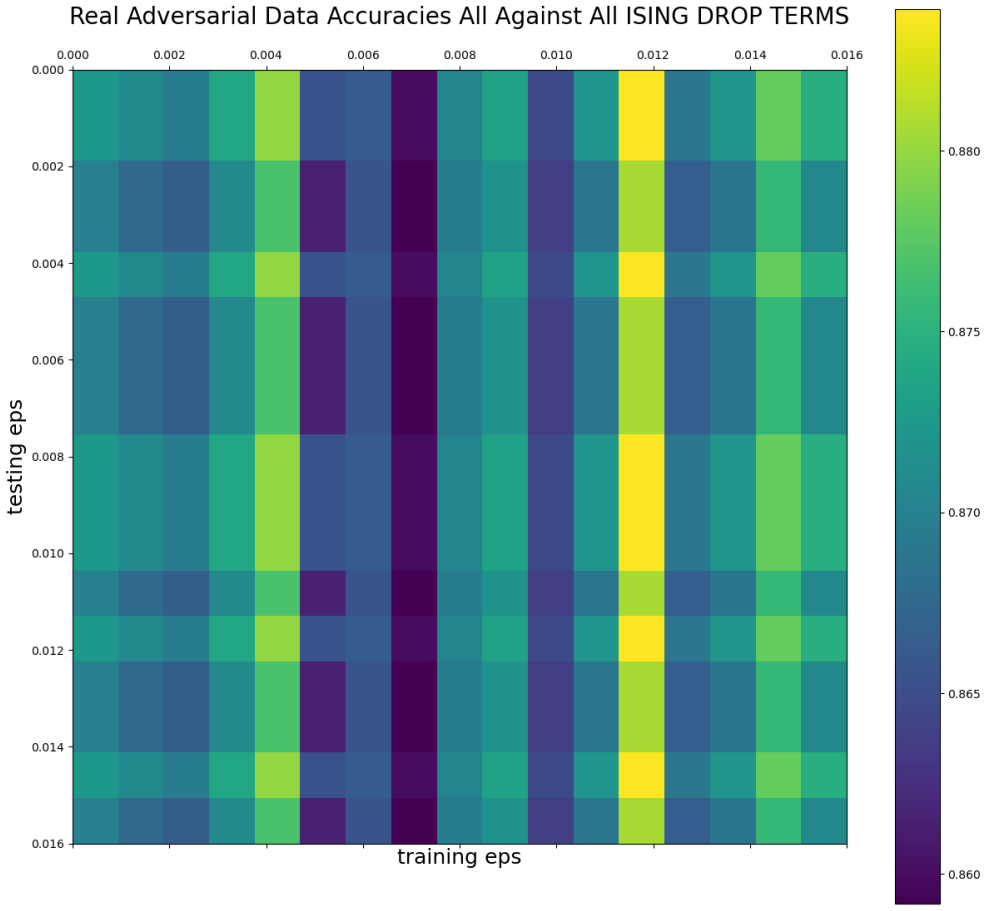

Figure 18: Results matrix for QSCs composed by solving for ensembles of base classifiers trained against different attack strengths. Columns represent results of a single QSC, while rows represent results of a given QSC against images of a specific attack strength.

5 CONCLUSIONS

We find that relative to a fully connected solution using a virtual embedding, use of direct embeddings on the D-Wave produces surprising high-quality results, given the stark differences in connectivity between full embedding and the Chimera graph structure. In addition, use of weights trained on the D-Wave for detecting adversarial examples represents an advantage relative to the CNN under attack, though this advantage holds for non-quantum Haar cascades as well. Finally, we propose further work on the use of total system magnetization in a consensus-based classifier making use of reverse annealing mode to set qubits directly according to base classifier results.

6 REFERENCES

[1] E. Boyda, S. Basu, S. Ganguly, A. Michaelis, S. Mukhopadhyay and R. R. Nemani, "Deploying a quantum annealing processor to detect tree cover in aerial imagery of California," *PLOS One,* vol. 12, no. 2, p. e0172505, 2017.

[2] A. Mott, J. Job, J.-R. Vilmant, D. Lidar and M. Spiropulu, "Solving a Higgs optimization problem with quantum annealing for machine learning," *Nature,* vol. 500, pp. 375-279, 2017.

[3] R. E. Schapire, "The Strength of Weak Learnability," *Machine Learning,* vol. 5, pp. 197-227, 1990.

[4] Y. Freund and R. E. Scapire, "A Short Introduction to Boosting," *Journal of the Japanese Society for Artificial Intelligence,* vol. 14, no. 5, pp. 771-780, 1999.

[5] H. Neven, V. S. Denchev, G. Rose and W. G. Macready, "QBoost: Large Scale Classifier Training with Adiabatic Quantum Optimization," *Journal of Machine Learning Research Workshop and Proceedings,* pp. 333--348, 2002.

[6] J. Cai, W. G. Macready and A. Roy, "A Practical Heuristic for Finding Graph Minors," 2014. [Online]. Available: https://arxiv.org/abs/1406.2741.

[7] T. Boothby, A. D. King and A. Roy, "Fast Clique Minor Generation in Chimera Qubit Connectivity Graphs," *Quantum Information Processing,* vol. 15, no. 1, pp. 495-508, 2016.

[8] N. Dattani, S. Szalay and N. Chancellor, "Pegasus: The Second Connectivity Graph for Large-Scale Quantum Annealing Hardware," 2019. [Online]. Available: http://arxiv.org/abs/1901.07636.

[9] N. Dattani and N. Chancellor, "Embedding Quadratization Gadgets on Chimera and Pegasus Graphs," 2019. [Online]. Available: http://arxiv.org/abs/1901.07676.

[10] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2001.

[11] H. Neven, V. Denchev and M. Drew-Book, "Binary Classification using Hardware Implementation of Quantum Annealing," *Quantum,* pp. 1-17, 2009.

[12] T. Albash, W. Vinci and D. A. Lidar, "Simulated Quantum Annealing Compairson Between All-to-All Connectivity Schemes," *Physical Review A,* vol. 94, no. 2, 2016.

[13] I. J. Goodfellow, J. Shlens and C. Szeg, "Explaining and Harnessing Adversarial Examples," 2014. [Online]. Available: arXiv:1412.6572.

[14] A. M. L. Vilela and H. E. Stanley, "Effect of Strong Opinions on the Dynamics of the Majority-Vote Model," *Scientific Reports,* vol. 8, no. 8709, 2018.

[15] K. Vangara, M. C. King, K. S. Krishnapriya, V. Albiero and K. Bowyer, "Characterizing the Variability in Face Recognition Accuracy Relative to Race," in *IEEE Conference on Computer Vision and Pattern Recognition*, Long Beach CA, 2019.

LIST OF ABBREVIATIONS

Convolutional neural network	CNN
Classical strong classifier	CSC
Fast gradient sign method	FGSM
Graphics processing unit	GPU
Javascript object notation	JSON
Machine learning	ML
Mean square error	MSE
Quantum anealing	QA
Quantum strong classifier	QSC
Quadratic unconstrained binary optimization	QUBO
Reverse annealing	RA
Strong-classifier quantum annealing device	SQUAD