# Pneumonia Detection by Binary Classification: Classical, Quantum and Hybrid Approaches for Support Vector Machine (SVM)

**Sai Sakunthala Guddanti** [1], **Apurva Padhye** [2], **Anil Prabhakar** [3] **and Sridhar Tayur** [*4]

[1] *Centre for Quantum Information Communication and Computing, Indian Institute of Technology Madras, Chennai, India*
[2] *Department of Electrical and Computer Engineering, University of Maryland, College Park, USA*
[3] *Dept. of Electrical Engineering, Indian Institute of Technology Madras, Chennai, India*
[4] *Tepper School of Business, Carnegie Mellon University, Pittsburgh, USA*

Correspondence*:
Sridhar Tayur
stayur@andrew.cmu.edu

## ABSTRACT

Early diagnosis of pneumonia is crucial to increase the chances of survival and to reduce the recovery time of the patient. Chest X-ray images, the most widely used method in practice, are challenging to classify. Our aim is to develop a machine-learning tool that can accurately classify images as belonging to normal or infected individuals. A support vector machine (SVM) is attractive because binary classification can be represented as an optimization problem, in particular as a Quadratic Unconstrained Binary Optimization (QUBO) model, which, in turn, maps naturally to an Ising model, thereby making annealing – classical, quantum and hybrid – an attractive approach to explore.

In this paper, we offer a comparison between different methods: (1) a classical state-of-the-art implementation of SVM (LibSVM); (2) solving SVM with a classical solver (Gurobi), with and without decomposition; (3) solving SVM with simulated annealing; (4) solving SVM with quantum annealing (D-Wave); and (5) solving SVM using Graver Augmented Multi-seed Algorithm (GAMA). GAMA is tried with several different numbers of Graver elements and a number of seeds, using both simulating annealing and quantum annealing. We find that simulated annealing and GAMA (with simulated annealing) are comparable, provide accurate results quickly, competitive with LibSVM, and superior to Gurobi and quantum annealing.

**Keywords: Quantum Annealing, Quantum Machine Learning, Binary Classification, Graver Augmented Multi-seed Algorithm, Support Vector Machine**

## 1 INTRODUCTION

Pneumonia is a major disease prevalent across the globe. Caused by the bacteria and viruses in the air we breathe, the illness affects one or both of the lungs, creating difficulty in breathing. Pneumonia accounts for more than 15% of deaths in children under the age of five (World Health Organization, 2022). Early and accurate diagnosis of pneumonia, therefore, is crucial to prevent deaths and ensure better treatment.

There are many widely used tests to diagnose pneumonia, such as chest X-rays, chest MRI, and needle biopsy of the lung. Chest X-ray imaging is the most commonly used method, as it is relatively inexpensive and non-invasive. Figure 1 shows examples of healthy and pneumonic lung X-rays. However, the examination of chest X-rays is challenging and sensitive to subjective variability. Machine learning

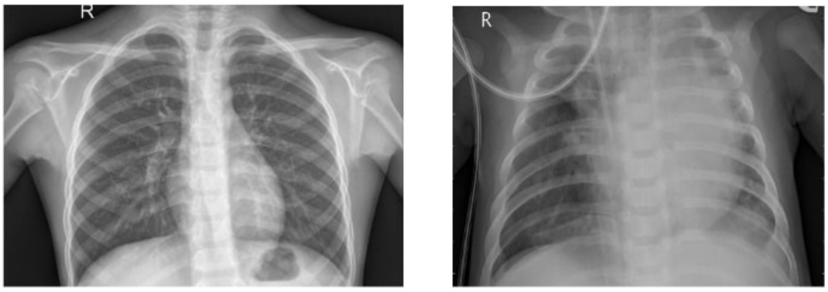

**Figure 1.** The image on the left shows a normal chest X-ray whereas the one on the right shows lungs with pneumonia opacity (Breviglieri, 2021).

(ML) techniques have gained popularity for solving the image classification problem, and so have found their use in pneumonia diagnosis as well. Support Vector Machine (SVM) is a widely used method for classification. We have the added advantage of being able to reframe the SVM as a Quadratic Unconstrained Binary Optimization (QUBO) problem, making it especially suitable for studying annealing methods. In this work, we computationally evaluate a variety of SVM methods, in the context of X-ray imaging for pneumonia, and compare our results against LibSVM, a state-of-the art implementation of SVM. Our main contributions include:

1. Studying a QUBO formulation of an SVM using simulated annealing (SA) and quantum annealing (QA).
2. Solving a QUBO with Gurobi, and comparing with annealing methods.
3. Combining multiple weak SVMs to get a strong classification model to accommodate fewer qubits on NISQ quantum annealers.
4. Studying a hybrid quantum-classical optimization heuristic technique, Graver Augmented Multi-seed Algorithm (GAMA).

## 2 RELATED WORK WITH CNNS AND SVMS

Nagashree and Mahanand (2023) compared the performance of an SVM with a few other classification algorithms, namely, decision tree, naïve Bayes, and $K$ nearest neighbour. The comparison results indicate a better performance of SVMs for diagnosing pneumonia. Darici et al. (2020) and Kundu et al. (2021) developed an ensemble framework and implemented it with deep learning models to boost their individual performance.

Many researchers have explored, using different data sets, comparing between classical and quantum machine learning algorithms. Willsch et al. (2020) introduced a method to train an SVM on a D-Wave quantum annealer and studied its performance in comparison to classical SVMs for both synthetic data and real data obtained from biology experiments. Wang et al. (2022) implemented an SVM, enhanced with quantum annealing, for two fraud detection data sets. They observed a potential advantage of using an SVM with quantum annealing, over other classical approaches, for bank loan time series data. Delilbasic et al. (2021) implemented two formulations of a Quantum Support Vector Machine (QSVM), using IBM quantum computers and D-Wave quantum annealers, and compared the results for Remote Sensing (RS) images. Bhatia and Phillipson (2021) compared classical approach, simulated annealing, hybrid solver and fully quantum implementations for public Banknote Authentication dataset and the Iris Dataset.

Researchers have also studied Convolutional Neural Networks (CNN) in this context. Although it is not the focus of our paper, we mention related literature. Sirish Kaushik et al. (2020) implemented four models of CNNs, and reached an accuracy of 92.3%. Youssef et al. (2020) and Nakrani et al. (2020), implemented deep learning models (different types of CNNs) to classify the data. Madhubala et al. (2021) extended the classification to more than two types of pneumonia. They used CNNs for classification and later performed augmentation to obtain final results. Ibrahim et al. (2021), considered bacterial pneumonia, non-COVID viral pneumonia, and COVID-19 pneumonia chest X-ray images. They performed multiple experiments with binary and multi-class classification and achieved a better accuracy in identifying COVID-19 (99%) than normal pneumonia (94%).

# 3 BACKGROUND INFORMATION

## 3.1 QUBO formulation of SVM

Recall that an SVM is a supervised machine learning model. The hyper-plane produced by the SVM maximizes its distance between the two classes. Figure 2 shows the support vectors and the hyper-plane that classifies data into two classes (labels +1 and -1).

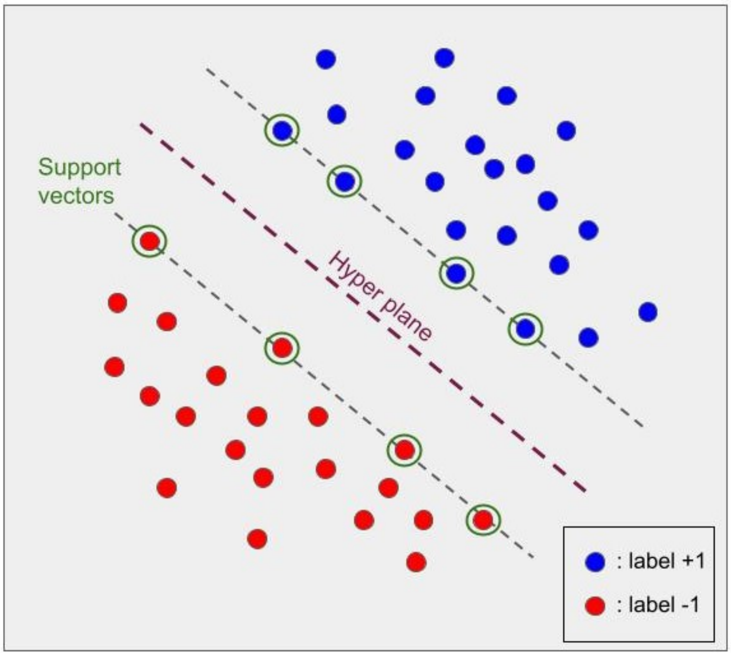


**Figure 2.** Representation of hyperplane in SVM separating two classes of data.

Given training data $X \in \mathbb{R}^{N \times d}$ and training labels $Y \in \{-1, +1\}^N$, where $N$ is the number of training data points, we look for a hyper-plane determined by weights, $w \in \mathbb{R}^d$, and bias, $b \in \mathbb{R}$, to separate the training data into two classes. Mathematically, the SVM is expressed as (Date et al., 2021):

$$\min_{w,b} \frac{1}{2}\|w\|^2, \tag{1}$$
$$\text{subject to } y_i(w^T x_i + b) \geq 1, \quad \forall i = 1, 2, \ldots, N.$$

Here, $x_i$ is the $i$-th row vector in $X$ and $y_i$ is the $i$-th element in $Y$. The Lagrangian function of this optimization problem is

$$\mathcal{L}(w, b, \lambda) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{N} \lambda_i [y_i(w^T x_i + b) - 1], \tag{2}$$

where $\lambda$ is the vector containing all the Lagrangian multipliers; that is, $\lambda = [\lambda_1, \ldots, \lambda_N]^T,$ with $\lambda_i \geq 0, \ \forall i$. Each Lagrange multiplier or support vector corresponds to one image and represents the significance of that particular image in determining the hyper-plane. Converting the above primal problem to its dual form yields a QUBO (Date et al., 2021)

$$\min_{\lambda} \mathcal{L}(\lambda) = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \lambda_i \lambda_j y_i y_j (x_i^T x_j) - \sum_{i=1}^{N} \lambda_i, \tag{3}$$

with the final weights determined as

$$w = \sum_{i=1}^{N} \lambda_i y_i x_i, \tag{4}$$
$$\sum_{i=1}^{N} \lambda_i y_i = 0, \tag{5}$$

and $\lambda_i, \lambda_j \geq 0, \forall i, j$. Since the data are linearly inseparable, we use a kernel function to plot the input data to higher dimensions and then use the SVM on the higher dimensional data. The kernel matrix is defined as

$$K_{ij} = \phi(x_i)\phi(x_j), \ \ \forall i, j, \tag{6}$$

where $\phi(x_i)$ is some function of the input vector $x_i$. We have used the radial basis function (RBF), in this paper, as it can project data efficiently. Mathematically, the RBF is defined as

$$K(x_1, x_2) = \exp\left(-\frac{\|x_1 - x_2\|^2}{2\sigma^2}\right). \tag{7}$$

The value of $\sigma$ was chosen as 50 by trial. Substituting the RBF from (7) in (3) yields the QUBO,

$$\min_{\lambda} \mathcal{L}(\lambda) = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \lambda_i \lambda_j y_i y_j (K_{ij}) - \sum_{i=1}^{N} \lambda_i. \tag{8}$$

The Lagrange multipliers should also satisfy the condition in (5). Writing (8) as a matrix yields

$$\min_{\lambda} \mathcal{L}(\lambda) = \frac{1}{2}\lambda^T (K \odot YY^T)\lambda - \lambda^T \mathbb{1}_N,\ \lambda \geq \mathbb{0}_N. \tag{9}$$

Here, $K$ is the kernel matrix whose elements are defined by (6). $\mathbb{1}_N$ and $\mathbb{0}_N$ represent $N$-dimensional vectors of ones and zeros, respectively, and $\odot$ is the element-wise multiplication operation. This QUBO matrix becomes the input to an annealer (Ising solver), that solves the minimization objectives and returns the Lagrange multipliers (binary), or the support vectors.

The precision vector is introduced to have integer support vectors instead of only binary and the dimension of the precision vector depends on the range of integer values for the support vector. The precision vector has powers of 2 as elements, and here we use $p = [2^0, 2^1]$ to get the final QUBO matrix. Now the dimensions of the QUBO have doubled and our support vectors can be four integers (0,1,2,3) instead of just being binary. Let $\hat{\lambda} = [\lambda_{11}, \lambda_{12}, \ldots, \lambda_{N1}, \lambda_{N2}]$ be the expanded Lagrange multipliers vector, which gives us our final QUBO. We pass the QUBO matrix to an annealer (Ising solver). The final $\hat{\lambda}$ vector obtained minimizes the QUBO,

$$\min_{\hat{\lambda}} \mathcal{L}(\hat{\lambda}) = \frac{1}{2}\hat{\lambda}^T P^T (K \odot YY^T) P\hat{\lambda} - \hat{\lambda}^T P^T \mathbb{1}_N, \tag{10}$$

where $P = I_n \otimes p$ and $\lambda = P\hat{\lambda}$. The annealer returns expanded Lagrange multipliers $\hat{\lambda}$, which we use to calculate support vectors $\lambda$. We can predict the labels for unseen data, using $\lambda$, as.

$$\mathsf{label}(x) = \text{sign}\left(\sum_{i=1}^{N} \lambda_i y_i (K_{xi}) + b\right), \tag{11}$$

$$b = \mathsf{mean}(y_i - w^T x_i), \quad \text{where } i \in [0, \ldots, N], \tag{12}$$

$$w^T x_i = \sum_{j=1}^{N} \lambda_j y_j K_{ji},$$

with $K_{xi}$ being the kernel between the new test point $x$ and training data point $i$, as defined in (6).

## 3.2 Graver Augmented Multiseed Algorithm (GAMA)

Let our binary optimization problem be of the form:

$$\text{objective function: } \min f(x)$$
$$\text{constraints:} Ax = b.$$

Alghassi et al. (2019a) introduced a novel fusion of quantum and classical methodologies for computation of Graver basis. In (Alghassi et al., 2019b), where the heuristic was named GAMA - Graver Augmented Multiseed Algorithm - was further studied the application of Graver basis (computed classically) as a means to attain good solutions. In this article, we explore the performance of GAMA in the context of solving an SVM.

GAMA is a heuristic algorithm, in which we compute a partial Graver basis and obtain many feasible solutions, using Ising solvers. The motivation for GAMA comes from the theoretical foundation that a complete Graver basis is a *Test-Set* for a wide variety of objective functions (Graver (1975) Murota et al. (2004) De Loera et al. (2009) Hemmecke et al. (2011) Murota et al. (2004) Lee et al. (2010)). Of course, finding a complete Graver basis (Pottier (1996)) is not realistic for most realistic size problems, but a partial Graver basis is significantly easier to obtain in certain situations, especially by solving QUBOs with Ising solvers. To make up for this incompleteness of the Graver basis, we rely on the availability of multiple feasible solutions. The GAMA heuristic, therefore, is performing a (partial) Graver walk from each of the feasible solutions as the seed (hence "multiseed"), and then picking the best among the (potentially) local optimal solutions. For finding the Graver bases, we consider the QUBO form of the constraints matrix $Ax = 0$. The Ising solver gives us many kernel elements, and performing conformal filtration on these kernel elements gives us the partial Graver bases. To get feasible solutions, we take the QUBO form of the constraints matrix $Ax = b$ (and solve it using an Ising solver). An alternative is to find kernel elements as differences of the feasible solutions and thus partial Graver bases and augment every feasible solution using the Graver bases to obtain solutions that are likely only a local optimum. To be clear, we have the following steps:

1. Find (partial) Graver basis (either by finding several kernel elements by solving a QUBO for $Ax = 0$ or by taking differences of feasible solutions found in step 2);
2. Find feasible solutions by solving a QUBO for $Ax = b$;
3. Augment the feasible solutions using partial Graver basis elements, computing the objective function value $f(x)$ at each step, and choosing the best solution among all (potentially) local optimal solutions.

## 4 DATA AND PRE-PROCESSING

The data set used is from Kaggle (Breviglieri, 2021) (Kaggle,RRID: SCR_013852): 1000 images from each of the normal and opacity classes are used for training the SVM, while 267 images from the normal class and 1000 images from the opacity class are used to test the trained model for evaluation of performance. Originally the images are of different sizes and dimensions. Therefore, the images are first resized to 200 $\times$ 200 pixels. The resized images are then flattened to give 1-dimensional arrays of 40,000 pixels.

Although the original data set in Kaggle contains more than 4000 images, we have considered only 2000 training images. In the dataset we observed 1082 normal images available for training, while there are more than 3000 images with signs of pneumonia. To get unbiased results from the ML models, we began our training with a balanced dataset. Thus we considered 1000 normal images and 1000 opacity images, as the data set in our studies.

## 5 METHODS

We begin with a discussion of each method.

### 5.1 Method 1: LibSVM (Benchmark)

LibSVM is a state-of-the-art library that implements support vector machines (Chang and Lin (2011)) using the input data sets directly, without going through the formulation of a QUBO. The results from LibSVM are typically considered to be a benchmark to compare other newer methods against.

### 5.2 Method 2: SVM using Gurobi

An SVM modelled as QUBO, as in (10), can be solved using a state of the art classical solver, such as Gurobi (version 9.5.0). This is implemented in two ways:

1. All 2000 training images are taken at once and incorporated in the QUBO. The solver returns expanded Lagrange multipliers as an array of 4000 elements, using which we construct 2000 support vector values and make predictions on test data.
2. The training set is divided into 40 sets, each of 50 images. Every set represents an SVM. The 40 SVMs are solved separately and combined using majority voting bagging (Kim et al., 2002). This approach is discussed in detail in section 5.3.

### 5.3 Method 3: SVM using Annealing

We used the D-Wave neal simulated annealer, digital annealer from IITM, and the Advantage_system 6.2 from D-Wave with 5614 qubits with the Pegaus connectivity between them (Dattani et al., 2019) as our three Ising solver options. Among these, the first two are simulated annealers, while the latter is a quantum annealer.

With additional lenience given for the Lagrange multipliers using a precision vector, the QUBO matrix for 2000 input images has a size of $4000 \times 4000$. This is beyond the processing capacity of simulated annealing using D-Wave neal and D-Wave quantum annealing. To overcome this, we opted to partition the images into 20 distinct sets, each comprising 100 images, giving a QUBO matrix of size $200 \times 200$, which can be solved with simulated annealers, while still remaining challenging for quantum annealing platforms.

Subsequently, we refined our strategy by further dividing the images into 40 sets, each encompassing 50 images (25 from each class). As a result, there are 40 SVMs (40 QUBO matrices) of size $100 \times 100$. These 40 SVMs are trained separately, and their outputs are combined using the majority voting bagging technique (Kim et al., 2002) to obtain the final decision boundary for classification. This framework is summarised in Figure 3.

#### 5.3.1 Method 3(a): Simulated Annealing

**Simulated annealing using the D-Wave neal package:**
The 40 QUBOs corresponding to 40 SVMs are solved individually using a simulated annealer, with 1000 iterations each per SVM. The output of the annealer is the set of expanded Lagrange multipliers for all the 1000 iterations. We filter the one which gives the minimum energy among 1000 iterations for every SVM, and thus obtain 40 sets of expanded Lagrange multipliers for 40 SVMs, using which we get our final support vectors. The 40 SVMs are combined using the majority voting bagging technique, and the prediction of unseen test data is done by (11). The simulated annealer was configured using the default parameter values specified by D-Wave neal in our study.

**Simulated annealing using the Digital Annealer of IITM:**
In the utilization of the Digital Annealer for simulated annealing, it was essential to designate parameter values, that is, the starting and ending temperature and iterations to perform at every temperature while descending. We converted all 40 QUBOs to Ising formulations and gave them as input to the digital annealer. The annealer performs one round of annealing from starting temperature to ending temperature with a specified number of iterations at every step. We took the initial temperature to be 6.4K, the final temperature to be 0.001K, and iterations at every step to be 20. The output we get would be the final spin values of the Ising formulation and its final energy value. We take the spin values output for all 40 SVMs

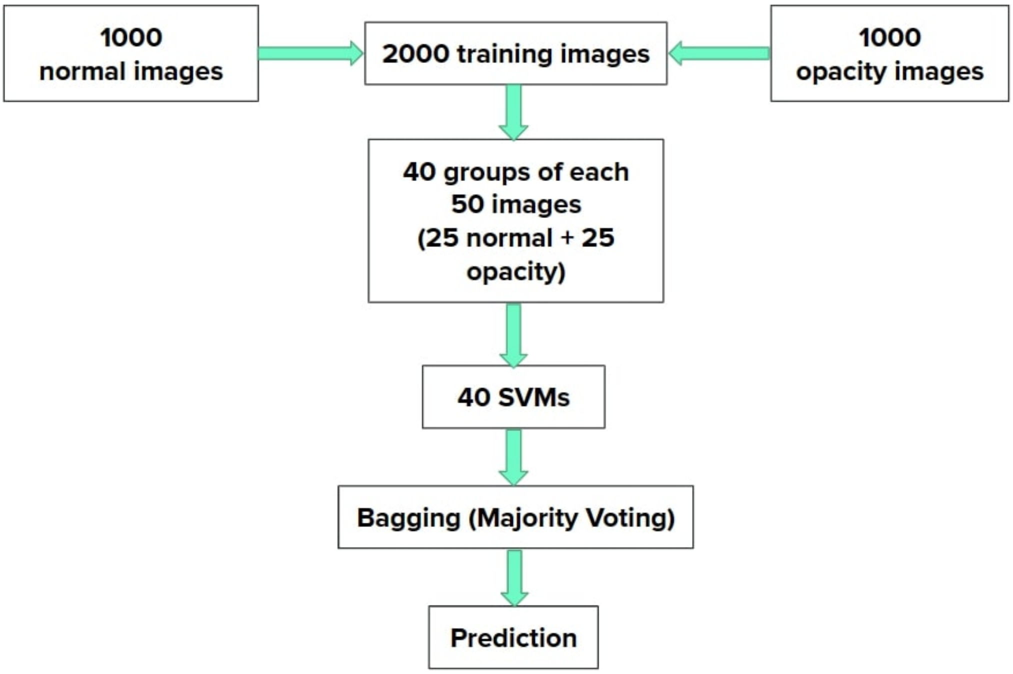


**Figure 3.** Flowchart of the steps involved in our proposed method for utilizing SVM using annealing.

which are expanded Lagrange multipliers and calculate support vectors. These are combined using majority voting and prediction for unseen test data is done by (11)

#### 5.3.2 Method 3(b): Quantum Annealing

The procedure employed resembles that of simulated annealing with D-Wave neal. Here, instead of 1000, we have taken 500 iterations of the D-Wave quantum annealer. It's important to note that, unlike simulated annealing, quantum annealers often have substantial queue times.

### 5.4 Method 4: SVM using GAMA

GAMA can be a very efficient method when the objective function is complex but the constraints are simple (Alghassi et al., 2019b). We give the simpler constraints to the annealer, obtain partial Graver elements and feasible solutions, and do a walkback using the initial objective function to obtain a final solution. The constraint equation is given by (5).

To ensure that the algorithm does not get stuck in a local minimum while performing augmentation, we implement a Metropolis-Hastings version of GAMA. In this case, we consider the probability of moving in any of the directions according to the ratio in the objective function value, and not just in the direction of improvement. We end the augmentation iterations if the change in objective function value remains constant for more than ten iterations.

#### 5.4.1 Method 4(a): GAMA using Simulated Annealing

We tested simulated annealing from D-Wave and the Digital Annealer from IITM. Similar to method 3 (sec 5.3), the images are divided into 40 sets (40 SVMs). Recall that we use the constraint mentioned in (5) to get Graver bases and feasible solutions:

$$\sum_{i=1}^{N} \lambda_i y_i = 0. \tag{13}$$

The constraint matrix (QUBO matrix framed from the above equation) remains the same for all SVMs as the $Y$ vector (labels vector) remains the same for all 40 SVMs (each SVM first has 25 normal and then 25 opacity images). As the right-hand part of constraints is zero, kernel elements and the feasible solutions are also the same. This special structure implies that a single execution of the annealer is sufficient to address the optimization requirements for all 40 SVMs. Thus, the Graver bases and feasible solutions are obtained once and used for augmentation in all SVMs.

A total of 500 feasible solutions (also kernel elements) were obtained using simulated annealing using the D-Wave neal package (from dwave-ocean-sdk). For simulated annealing using the digital annealer of IITM, we have taken the QUBO of constraint mentioned above in (5) and converted it to an Ising formulation. The annealer performs one round of annealing at a time as mentioned in method 3(a). We took the initial temperature to be 6.4K, the final temperature to be 0.001K, and iterations at every step to be 20. The entire annealing is performed for 500 times. Here, also, 500 feasible solutions (also kernel elements) are obtained. When conformal filtration is performed we obtained 499 partial Graver bases.

Detailed experimentation of this method is done using D-Wave neal simulated annealing. We experimented with three different sets of Graver bases and feasible solutions. The following cases are considered for augmentation:

1. 50 Graver elements + 50 feasible solutions
2. 100 Graver elements + 100 feasible solutions
3. 200 Graver elements + 200 feasible solutions

We obtain 40 sets of Lagrange multipliers corresponding to 40 SVMs for each of the three cases above. The majority voting bagging is used to combine 40 SVMs, and the final output is tested on the test data set according to (11). Using the digital annealer from IITM, we have utilized all 499 partial Graver bases and feasible solutions and performed the augmentation.

### 5.4.2 Method 4(b): GAMA using D-Wave quantum annealing

The GAMA with quantum annealing process follows a methodology akin to that of GAMA involving simulated annealing. The number of feasible solutions was 127 (as compared to 500 in the earlier method). Note that out of 500 calls to D-Wave, only 127 gave the minimum energy solution. All 127 feasible solutions and corresponding (partial) Graver elements (computed via conformal filtration, which happened to also be 127, likely due to the fact that the kernel elements are short to begin with) were included in the augmentation process.

## 6 RESULTS AND ANALYSIS

The results of the various methods are compared through confusion matrix representation and associated metrics as we detail next. A confusion matrix is a tabular representation used to assess the performance of classification models. It provides a comprehensive overview of how well the model's predictions align with actual outcomes for different classes or categories. The matrix is constructed by comparing predicted

| Method | True $+$ve | False $+$ve | True $-$ve | False $-$ve | Time taken |
|---|---|---|---|---|---|
| LibSVM | 917 | 19 | 248 | 83 | 3 mins 30 sec |
| Gurobi1 | 712 | 11 | 256 | 288 | 30 mins |
| Gurobi2 | 860 | 111 | 156 | 140 | 2.44 sec |
| SimAnn-Dn | 927 | 22 | 245 | 73 | 6 mins 29 sec |
| SimAnn-Di | 884 | 20 | 247 | 116 | 20 sec |
| QuantumAnn | 924 | 46 | 221 | 76 | 12 sec |
| GAMA1 | 862 | 28 | 239 | 138 | 10s(anneal) + 7s(aug) |
| GAMA2 | 900 | 36 | 231 | 100 | 10s(anneal) + 36s(aug) |
| GAMA3 | 924 | 33 | 234 | 76 | 10s(anneal) + 153s(aug) |
| GAMA-Di | 885 | 67 | 200 | 115 | 256s(anneal) + 1196(aug) |
| GAMA-Q | 875 | 9 | 258 | 125 | 0.3s(anneal) + 92sec(aug) |

**Table 1.** Confusion matrix values and time taken for following methods respectively: LibSVM (Classical state-of-the-art implementation of SVM), Gurobi1 (Gurobi using all images at once), Gurobi2 (Gurobi with images split into 40 sets), SimAnn-Dn (Simulated Annealing using D-Wave neal), SimAnn-Di (Simulated Annealing using the Digital Annealer from IITM), QuantumAnn (Quantum Annealing with D-Wave), Simulated Annealing using D-Wave neal with GAMA (50 Graver + 50 feasible solutions), Simulated Annealing using D-Wave neal with GAMA (100 Graver + 100 feasible solutions), Simulated Annealing using D-Wave neal with GAMA (200 Graver + 200 feasible solutions), Simulated Annealing using the Digital annealer from IITM with GAMA (499 Graver + 499 feasible solutions), Quantum Annealing with GAMA run on D-Wave quantum annealer (127 feasible solutions + 127 Graver elements). In the table ”aug” represents augmenting time. Note: Quantum annealer time represents only quantum processor time. We are reporting the best of three runs for all annealing methods.

class labels with true class labels for data points. It represents a breakdown of the predictions into four categories: True Positives (TP) represent correctly predicted positive instances, True Negatives (TN) represent correctly predicted negative instances, False Positives (FP) represent instances that are incorrectly predicted as positive when they are actually negative, and False Negatives represent instances that are incorrectly predicted as negative when they are actually positive. The confusion matrix helps in evaluating metrics like accuracy, precision, recall, and F1-score, which help with a deeper understanding of the model's performance across various classes.

We evaluate the various methods on four metrics:

$$\text{Accuracy} = \frac{\text{TP+TN}}{\text{TP+TN+FP+FN}}, \tag{14}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP+FP}}, \tag{15}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP+FN}}, \tag{16}$$

$$\text{F1 score} = \frac{2\,\text{TP}}{2\,\text{TP+FP+FN}}. \tag{17}$$

For all the methods, the results are noted from the confusion matrix, which is as shown in Table1 (Recall that $+$ve means opacity, and $-$ve is normal). For quantum annealing, the annealing time including queue time and post-processing for 40 SVMs is 3 hours 16 minutes. In the table, we have removed all these and only provided annealing time. The metrics of comparison for all the methods are in Table2.

| Method | Accuracy | Precision | Recall | F1 score |
|---|---|---|---|---|
| LibSVM | 91.9 | 97.9 | 91.7 | 94.6 |
| Gurobi1 | 76.4 | 98.4 | 71.2 | 82.6 |
| Gurobi2 | 79.8 | 88.2 | 86 | 87 |
| SimAnn-Dn | 92.5 | 97.6 | 92.7 | 95 |
| SimAnn-Di | 89.2 | 97.7 | 88.4 | 92.8 |
| QuantumAnn | 90.3 | 95.2 | 92.4 | 93.7 |
| GAMA1 | 86.8 | 96.8 | 86.2 | 91.2 |
| GAMA2 | 89.2 | 96.1 | 90 | 92.9 |
| GAMA3 | 91.3 | 96.5 | 92.4 | 94.4 |
| GAMA-Di | 85.6 | 92.9 | 88.5 | 90.6 |
| GAMA-Q | 89.4 | 98.9 | 87.5 | 92.8 |

**Table 2.** Accuracy, Precision, Recall, and F1 score for all methods. We have highlighted, in red, the maximum values in each column for easy comparison.

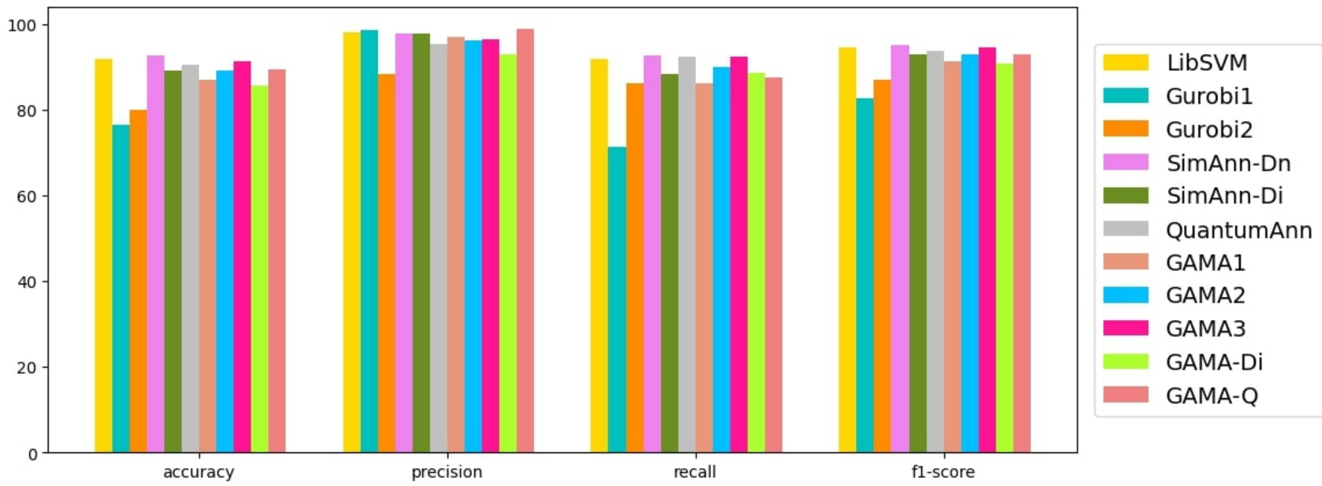


**Figure 4.** Comparative analysis of Accuracy, Precision, Recall, and F1 score for all methods.

### 6.1 Comparison of Methods

Since the running time for each method is different, we cannot draw direct comparisons based just on the values of the four metrics. However, Tables1 and 2 provide insight on some key points. All the metrics from Table 2 are plotted in the graph in Figure 4 for visual convenience. We use LibSVM as the classical solver to compare our SVM implementations against. As we observe in Table 2, the results from other methods, especially SimAnn-Dn, compare favourably against those from LibSVM.

- Gurobi, when given data divided into 40 SVMs, takes the least time (2.44 sec), but the performance is weak. When all images are input at once and trained for 30 minutes, there is no significant improvement in the performance.
- Simulated annealing performed using D-Wave neal takes around 6.5 minutes to run and the results obtained are good. The best accuracy (92.5%) and F1 score (95%) are achieved with simulated annealing.
- In case of GAMA, the performance improves as we increase the number of Graver elements taken for augmentation. The augmentation time taken also increases accordingly. (It reaches a threshold value of

performance as seen in Figures 10 and 12. See appendix.) Indeed, using 200 feasible solutions and 200 Graver elements appears sufficient to reach good performance relatively quickly.

- GAMA when implemented using quantum annealing takes around 8.5 minutes (including queue time) and provides accuracy similar to that of SVM using quantum annealing (Method 3(b)). Here, we can see a massive speed-up as method 3(b) takes more than 3 hours to run. Thus, despite limited connectivity, GAMA provides a significant time improvement for quantum annealing, without compromising on the metrics.
- Quantum annealers often have a lower precision for encoding QUBO coefficients. However, we found that this did not affect the results because the QUBO matrix elements ranged between 0 and 2, or between 0 and 4 when we used GAMA.

Among our approaches, for a given time budget (of training), the best methods are:

1. **5 minutes**: GAMA 3 (200 Graver elements + 200 feasible solutions).
2. **10 minutes**: Simulated annealing (method 3(a)) and GAMA 3 (200 Graver elements + 200 feasible solutions).
3. **20 minutes**: Simulated annealing (method 3(a)) and GAMA 3 (200 Graver elements + 200 feasible solutions).

Not much improvement is seen by increasing training time.

### 6.2 Bagging and Probability Distribution

Majority voting bagging (Kim et al., 2002), the method used to combine SVMs, also improves the performance of the combined SVM. The accuracy of annealing methods (method 3(a) and method 3(b)) without bagging and with bagging is compared in plots 5 and 6.

We can observe that the accuracy improved to 92.5% (Red line in 5) in the case of simulated annealing using D-Wave neal and to 90.3% (Red line in 6) in the case of D-Wave quantum annealing using majority voting bagging.

Many iterations of annealing are taken to find the Lagrange multipliers that best minimize the objective function value. It is instructive to know how often we might get the parameters that give the minimum objective function value. From Figure 5 and Figure 6, we also observe that some of the individual SVMs also give sufficiently good results. Thus, there maybe an opportunity to reduce computational time (by only solving a few SVMs rather than all 40) and obtain good results.

To understand the probability of obtaining the best solution, we plot the probability distribution for best performing SVMs (for simulated annealing using D-Wave neal and for quantum annealing, respectively). Figure 7 shows the probability distribution for all obtained solutions over 10000 iterations of simulated annealing for SVM number 31, which gave us the best individual SVM accuracy. We can see that although our desired low-energy solution occurred with low probability, the median solutions also give good accuracy. Figure 8 shows the probability distribution for all obtained solutions over 8000 iterations of D-Wave for SVM number 27, which gave us the best individual SVM accuracy. The distribution is similar to that of simulated annealing, but did not reach the quality of solutions of simulated annealing.

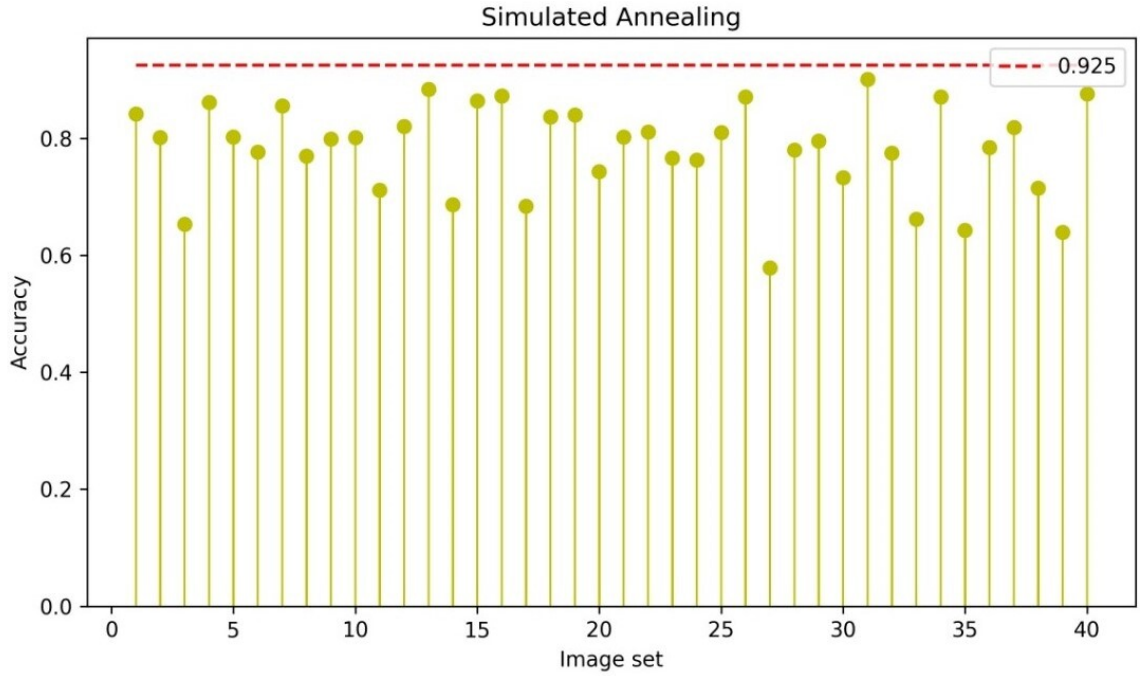


**Figure 5.** The yellow lines represent the accuracy metric for all the 40 SVMs we divided the data into. The red line shows the maximum accuracy achieved using weighted average bagging as 92.5%. All the SVMs are solved with simulated annealing using D-Wave neal.

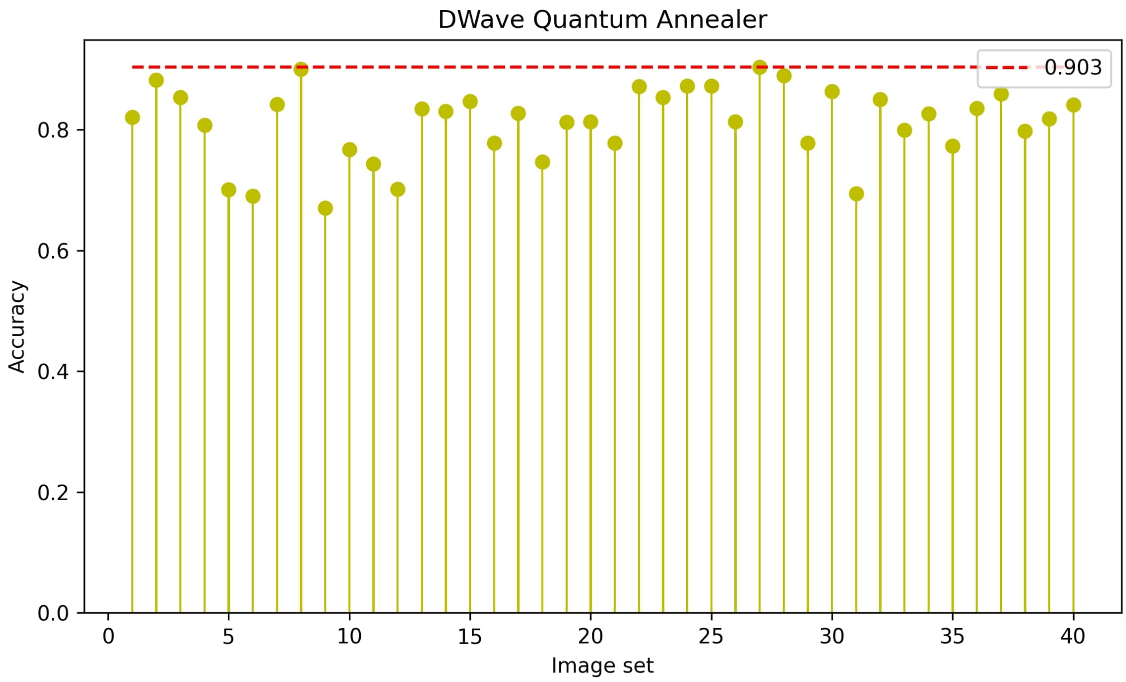


**Figure 6.** The yellow lines represent the accuracy metric for all the 40 SVMs we divided the data into. The red line shows the maximum accuracy achieved using weighted average bagging as 90.3%. All the SVMs are solved using quantum annealing.

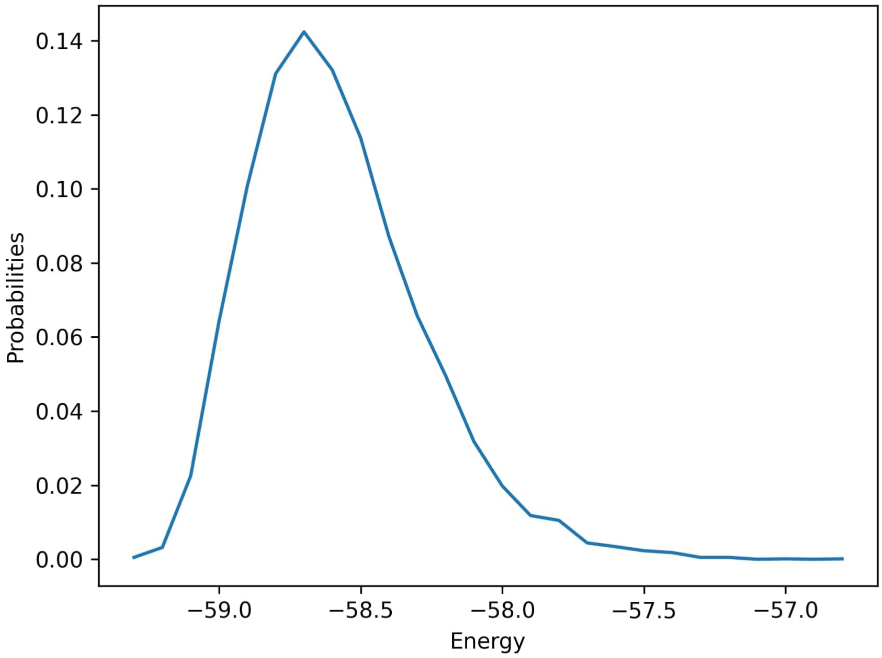


**Figure 7.** Probability distribution of simulated annealing solutions for SVM number 31. The best solution has energy around -59.

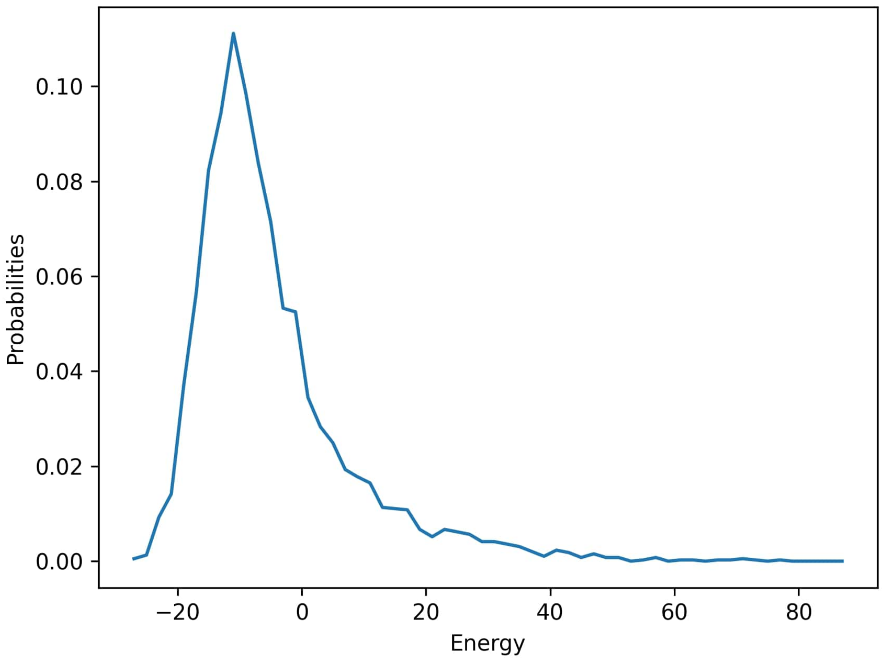


**Figure 8.** Probability distribution of D-Wave quantum annealing solutions for SVM number 27. Note that the best solution has energy of around -20, not as good as that found in simulated annealing.

## 7 CONCLUDING REMARKS

In this work, we explored binary classification through classical, quantum and hybrid methods, using X-ray imaging data for pneumonia, and used LibSVM as our benchmark. To have a balanced data set for SVM, we selected 1000 images, each, with and without pneumonia as our input data set. We partitioned the data into 40 sets. We formulated the SVM as a QUBO, and solved the QUBOs using simulated annealing, Gurobi and quantum annealing. Additionally, we studied GAMA heuristic, where the (different) QUBOs were solved using simulated annealing and quantum annealing. Each of our data sets yielded an SVM. We used bagging to combine the 40 SVMs, which improved the overall accuracy.

For binary classification of X-ray images, SVM can be an alternative to CNN, especially when considering pathways to implementations on a quantum annealer. The classical solver, LibSVM, shows a 92% accuracy in classification. However, Simulated Annealing using DWave neal (SimAnn-Dn) has comparable or better performance. GAMA provides a speed-up over quantum annealing with similar performance on metrics. Quantum annealing is not competitive in terms of time taken, but provides solutions of quality that are near the best obtained. We expect that the performance will improve as quantum annealers with more qubits and better connectivity come online, noting that the classical hardware and software also are expected to improve. This suggests that periodic comparisons should be encouraged. We hope that our work adds to the literature on the benchmarking of quantum, classical and hybrid approaches to solve a variety of important combinatorial optimization problems arising from practical applications (Metriq, 2023).

## AUTHOR CONTRIBUTIONS

Guddanti, S.S.: Writing – original draft, conceptualization, data curation, formal analysis, methodology, software; Padhye, A.: Writing – review and editing, conceptualization, data curation, formal analysis, methodology, software; Prabhakar, A.: Writing – review and editing, funding acquisition, supervision, validation; Tayur, S.: Writing – review and editing, funding acquisition, supervision, validation.

## FUNDING

We thank the Mphasis F1 Foundation and RAGS Foundation for supporting this work.

## DATA AVAILABILITY STATEMENT

The codes used for this study can be found in the [Pneumonia-Detection-by-Binary-Classification ] [https://github.com/Sai-sakunthala/Pneumonia-Detection-by-Binary-Classification]. The dataset of x-ray images are available as Kaggle,RRID: SCR_013852 (Breviglieri, 2021).

## REFERENCES

Alghassi, H., Dridi, R., and Tayur, S. (2019a). Graver bases via quantum annealing with application to non-linear integer programs. *arxiv preprint 1902.04215*

Alghassi, H., Dridi, R., and Tayur, S. (2019b). GAMA: A novel algorithm for non-convex integer programs. *arxiv preprint 1907.10930*

Bhatia, H. S. and Phillipson, F. (2021). Performance analysis of support vector machine implementations on the D-Wave quantum annealer. In *Computational Science – ICCS 2021*, eds. M. Paszynski, D. Kranzlmüller, V. V. Krzhizhanovskaya, J. J. Dongarra, and P. M. A. Sloot (Cham: Springer International Publishing), 84–97

[Dataset] Breviglieri, P. (2021). Pneumonia X-ray images. Https://www.kaggle.com/datasets/pcbreviglieri/pneumonia-xray-images

Chang, C.-C. and Lin, C.-J. (2011). LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology* 2, 27:1–27:27. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`

Darici, M. B., Dokur, Z., and Olmez, T. (2020). Pneumonia detection and classification using deep learning on chest X-ray images. *International Journal of Intelligent Systems and Applications in Engineering* 8, 177–183. doi:10.18201/ijisae.2020466310

Date, P., Arthur, D., and Pusey-Nazzaro, L. (2021). QUBO formulations for training machine learning models. *Scientific Reports* 11. doi:10.1038/s41598-021-89461-4

Dattani, N., Szalay, S., and Chancellor, N. (2019). Pegasus: The second connectivity graph for large-scale quantum annealing hardware. *arxiv preprint 1901.07636*

De Loera, J., Hemmecke, R., Onn, S., Rothblum, U., and Weismantel, R. (2009). Convex integer maximization via graver bases. *Journal of Pure and Applied Algebra* 213, 1569–1577. doi:https://doi.org/10.1016/j.jpaa.2008.11.033. Theoretical Effectivity and Practical Effectivity of Gröbner Bases

Delilbasic, A., Cavallaro, G., Willsch, M., Melgani, F., Riedel, M., and Michielsen, K. (2021). Quantum support vector machine algorithms for remote sensing data classification. In *2021 IEEE International Geoscience and Remote Sensing Symposium IGARSS*. 2608–2611. doi:10.1109/IGARSS47720.2021.9554802

Graver, J. E. (1975). On the foundations of linear and integer linear programming i. *Mathematical Programming* 9, 207–226

Hemmecke, R., Onn, S., and Weismantel, R. (2011). A polynomial oracle-time algorithm for convex integer minimization. *Mathematical Programming* 126, 97–117

Ibrahim, A. U., Ozsoz, M., Serte, S., Al-Turjman, F., and Yakoi, P. S. (2021). Pneumonia classification using deep learning from chest X-ray images during covid-19. *Cognitive Computation* doi:10.1007/s12559-020-09787-5

Kim, H.-C., Pang, S., Je, H.-M., Kim, D., and Bang, S.-Y. (2002). Support vector machine ensemble with bagging. In *Pattern Recognition with Support Vector Machines* (Springer Berlin Heidelberg). 397–408

Kundu, R., Das, R., Geem, Z. W., Han, G.-T., and Sarkar, R. (2021). Pneumonia detection in chest X-ray images using an ensemble of deep learning models. *PLOS ONE* 16, 1–29. doi:10.1371/journal.pone.0256630

Lee, J., Onn, S., Romanchuk, L., and Weismantel, R. (2010). The quadratic graver cone, quadratic integer minimization, and extensions. *Mathematical Programming* 136, 301–323

Madhubala, B., Sarathambekai, S., Vairam, T., Sathya Seelan, K., Sri Sathya, R., and Swathy, A. R. (2021). Pre-trained convolutional neural network model based pneumonia classification from chest X-ray images (May 24, 2021). In *Proceedings of the International Conference on Smart Data Intelligence (ICSMDI 2021)*

[Dataset] Metriq (2023). Community-driven quantum benchmarks. http://metriq.info

Murota, K., Saito, H., and Weismantel, R. (2004). Optimality criterion for a class of nonlinear integer programs. *Oper. Res. Lett.* 32, 468–472. doi:10.1016/j.orl.2003.11.007

Nagashree, S. and Mahanand, B. S. (2023). Pneumonia chest X-ray classification using support vector machine. In *Proceedings of International Conference on Data Science and Applications*, eds. M. Saraswat, C. Chowdhury, C. Kumar Mandal, and A. H. Gandomi (Singapore: Springer Nature Singapore), 417–425

Nakrani, N. P., Malnika, J., Bajaj, S., Prajapati, H., and Jariwala, V. (2020). Pneumonia identification using chest X-ray images with deep learning. In *ICT Systems and Sustainability*, eds. M. Tuba, S. Akashe, and A. Joshi (Singapore: Springer Singapore), 105–112

Pottier, L. (1996). The euclidean algorithm in dimension n. In *Proceedings of the 1996 International Symposium on Symbolic and Algebraic Computation* (New York, NY, USA: Association for Computing Machinery), ISSAC '96, 40–42. doi:10.1145/236869.236894

Sirish Kaushik, V., Nayyar, A., Kataria, G., and Jain, R. (2020). Pneumonia detection using convolutional neural networks (CNNs). In *Proceedings of First International Conference on Computing, Communications, and Cyber-Security (IC4S 2019)*, eds. P. K. Singh, W. Pawłowski, S. Tanwar, N. Kumar, J. J. P. C. Rodrigues, and M. S. Obaidat (Singapore: Springer Singapore), 471–483

[Dataset] World Health Organization (2022). Pneumonia in children. Https://www.who.int/news-room/fact-sheets/detail/pneumonia

Wang, H., Wang, W., Liu, Y., and Alidaee, B. (2022). Integrating machine learning algorithms with quantum annealing solvers for online fraud detection. *IEEE Access* 10, 75908–75917. doi:10.1109/ACCESS.2022.3190897

Willsch, D., Willsch, M., De Raedt, H., and Michielsen, K. (2020). Support vector machines on the D-Wave quantum annealer. *Computer Physics Communications* 248, 107006. doi:https://doi.org/10.1016/j.cpc.2019.107006

Youssef, T. A., Aissam, B., Khalid, D., Imane, B., and Miloud, J. E. (2020). Classification of chest pneumonia from X-ray images using new architecture based on ResNet. In *2020 IEEE 2nd International Conference on Electronics, Control, Optimization and Computer Science (ICECOCS)*. 1–5. doi:10.1109/ICECOCS50124.2020.9314567

# APPENDIX

## GAMA with larger set of Graver elements

We extend method 4(a) (D-Wave neal simulated annealing) by considering a larger number of Graver elements and feasible solutions for augmentation. This expanded approach aims to provide a more comprehensive and detailed understanding of the obtained results. We consider 250 and 499 feasible solutions and analyse results by increasing the number of Graver elements:

1. 500 Graver elements
2. 1000 Graver elements
3. 1500 Graver elements
4. 2000 Graver elements
5. 2500 Graver elements

As we can see in figure 9, all cases reach approximately the same final objective function value, although initial objective values for each case can be different. It can also be seen that the augmentation time varies linearly with the number of Graver elements considered. The metrics are almost flat with a larger number of Graver elements, and similar to method 4(a) with 200 Graver elements and 200 feasible solutions. In Figure11, with 499 feasible solutions instead of 250, all cases gave approximately the same final objective value, although initial objective values for each case differed, as before. The augmentation time increases non-linearly. However, the performance on the four metrics is not substantially improved.

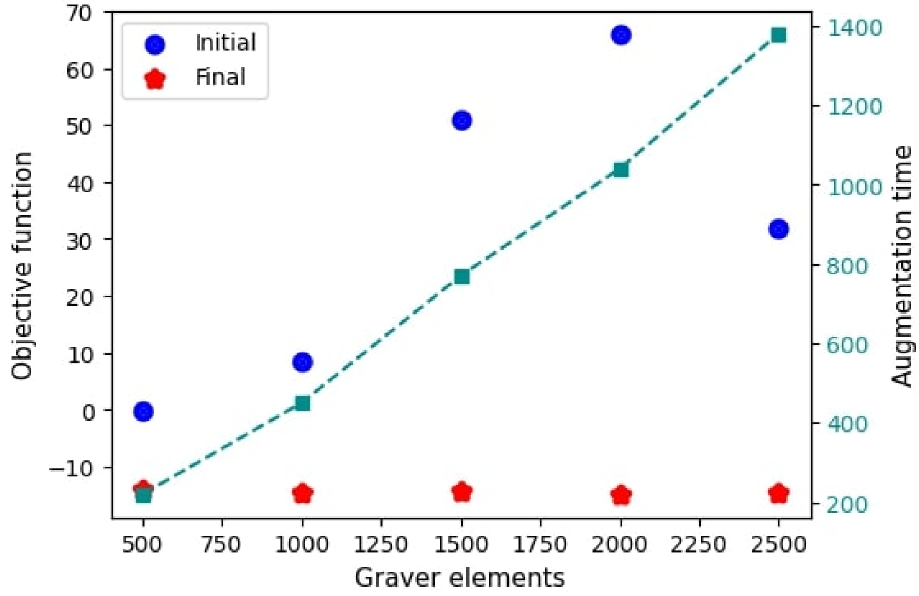


**Figure 9.** Graph shows how the objective function value changes when using method 4(a) (simulated annealing solutions using GAMA) with 250 feasible solutions and increasing number of Graver elements along with the augmentation time to reach the final solution.

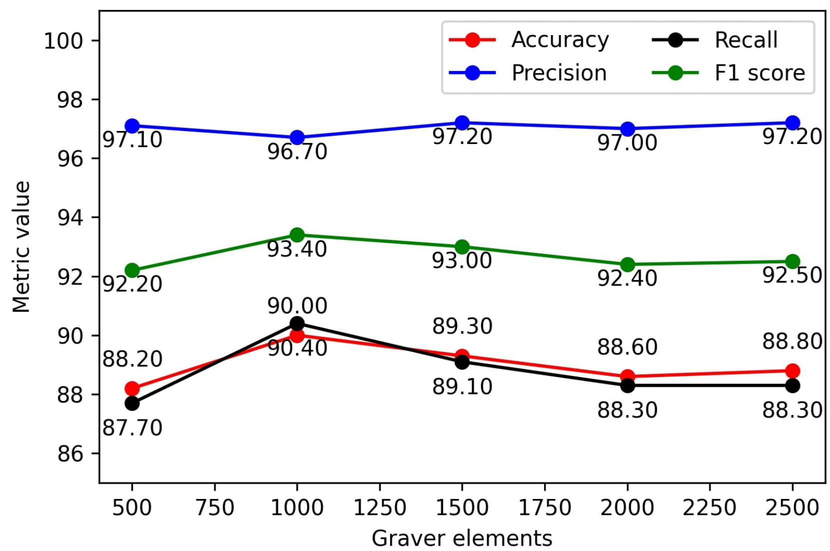


**Figure 10.** Graph shows the values of Accuracy, Precision, Recall, and F1 score using method 4(a) for 250 feasible solutions and increasing number of Graver elements.

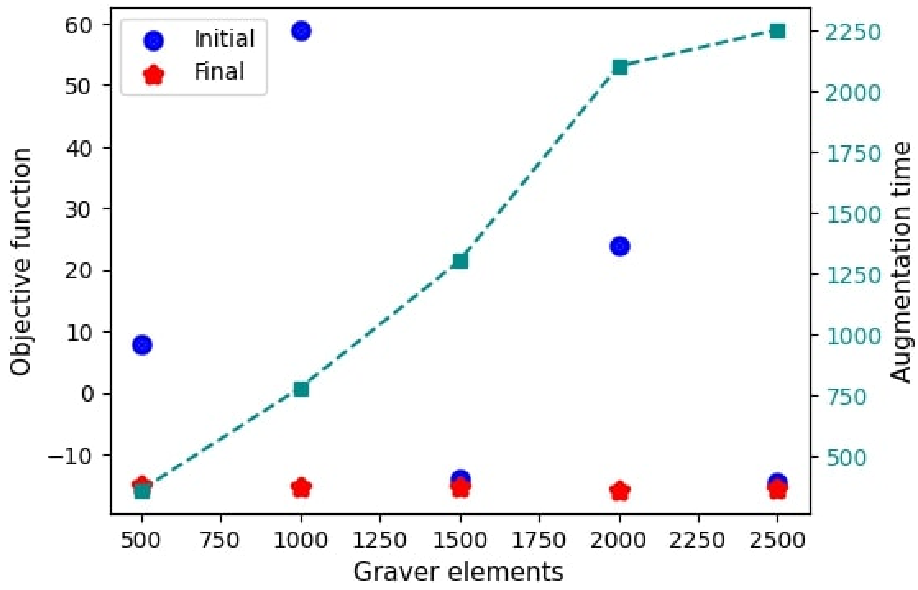


**Figure 11.** Graph shows how the objective function value changes when using method 4(a) (simulated annealing solutions using GAMA) with 499 feasible solutions and increasing number of Graver elements along with the augmentation time to reach the final solution.

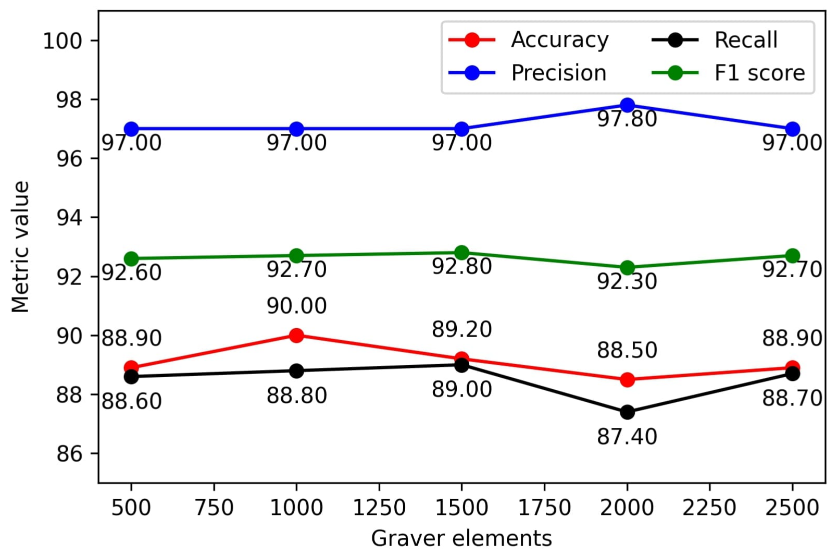


**Figure 12.** Graph shows the values of Accuracy, Precision, Recall, and F1 score using method 4(a) for 499 feasible solutions and increasing number of Graver elements.