Robust Classification with Non-Convex Loss and Regularization designed for Adiabatic Quantum Computation

Vasil S. Denchev\textsuperscript{1,2} \ William G. Macready and Team\textsuperscript{3} \ Hartmut Neven\textsuperscript{1}

\textsuperscript{1}Google
\textsuperscript{2}Purdue University
\textsuperscript{3}D-Wave Systems
"I argue that all progress, both theoretical and practical has resulted from a single human activity: the quest for what I call good explanations."

"The field of artificial (general) intelligence has made no progress because there is an unresolved philosophical problem at its heart: we do not understand how creativity works."

Training a Binary Classifier
Training a binary classifier
Binary Classifier

\[ x \mapsto y = H_\mathbf{w}(x) \]

with \( x \in \mathbb{R}^M \), \( y \in \{-1, 1\} \) and \( \mathbf{w} \in \mathbb{R}^M \)
Many machine learning methods formulate training as optimization problem

- Goal is to form classifier: $y = H_w(x)$
  that has minimal generalization error

- Given a set of $S$ training examples $\{(x_s, y_s) | s = 1, \ldots, S\}$
- Training: $w^{opt} = \arg\min_w (L(w) + R(w))$
- Loss $L(w)$ controls how well the classifier separates the classes
- Regularization $R(w)$ controls the complexity of the classifier
Regularization

From Bishop, Pattern Recognition and Machine Learning, 2006
Natural choices of loss and regularization render training formally NP-hard

For computational efficiency typically a convex objective $L(w) + R(w)$ is constructed

But that comes at a cost

Convex losses are not as robust to incorrectly labeled examples
Solutions obtained with convex regularization are not as sparse
Indication that tighter generalization bounds can be obtained with non-convex losses
Minimum of objective function does not correspond to minimal training error
We studied non-convex versions of $L(w)$ and $R(w)$ rendering the problem formally NP hard.
Treated training as an integer program working with weights $w$ of low bit depth.
Adiabatic quantum optimization (AQO)

$H(t)$: slowly varying Hamiltonian for evolution from $t = 0$ to $t = T$

$H(0) = H_B$: initial Hamiltonian with known and easily preparable ground state

$H(T) = H_P$: Hamiltonian whose ground state encodes the solution to a given instance of an optimization problem

System remains in ground state during quantum evolution:

$$i \frac{d}{dt} |\psi(t)\rangle = H(t) |\psi(t)\rangle$$
Implementation of AQO in the D-Wave architecture

D-Wave implements the Ising model in hardware:

$$H_{\text{Ising}} = \sum_{i,j} J_{ij} \sigma_z^{(i)} \sigma_z^{(j)} + \sum_i h_i \sigma_z^{(i)}$$

$J_{i,j}$ are coupling strengths between pairs of qubits. $h_i$ are biases of individual qubits. $\sigma_z$ are Pauli matrices.

Finding the ground state of $H_{\text{Ising}}$ amounts to solving a QUBO

$$w^* = \arg \min_w \{ w' Q w \}$$

Equivalent to Weighted MAX-2-SAT

Simplest many body system!
Illustration of computation in D-Wave processor

Qubits connected via Chimera graph
Linear binary classifier

\[ y = H_w(x) = \text{sign} \left( \sum_{i=1}^{N} w_i h_i(x) + b \right) \]

\( x \in \mathbb{R}^M \): input patterns to be classified
\( y \in \{-1, 1\} \): output of the classifier
\( h_i : x \mapsto \{-1, +1\} \): weak classifiers
\( w_i \in [0, 1] \): set of weights to be optimized
\( b \in \mathbb{R} \) is the bias
\( H(x) \): strong classifier
Weights with low bit depth are sufficient, are even beneficial

Argument why weights with low bit depth are sufficient

Each training example creates hyper plane in weight space
Those create solution regions
As long as each region contains a least one vertex of the hyper lattice we are good

Lower bound: \( \text{bits} \geq \log_2(f) + \log_2(e) - 1 \)

Training with Convex Loss and Non-Convex Regularization

\[ \begin{align*} L(w) &= \sum_{s=1}^{S} L_{\text{square}}(m_s) \\ L_{\text{square}}(m_s) &= (m_s - 1)^2 \\ m_s &= y_s (w^T x_s + b) \quad \text{is the margin of example } s \\ R(w) &= \lambda \| w \|_0 \\ O(N) \text{ qubits are spent representing the weights and threshold} \end{align*} \]
Efficient use of qubits allows for flexible (heuristic) extensions

Assume first round of optimization yields

\[ y = \text{sign} \sum_{t=1}^{T} w_t h_t(x) + b \]

Assume \( T < Q \)

Re-run optimization by adding new weak classifiers \( \{ h_i(x) | i = 1, \ldots, Q - T \} \) selected via boosting

If large classifier is needed with \( T > Q \)

Freeze weights obtained in previous optimization rounds and concatenate

\[
 y = \text{sign} \left( \sum_{t=1}^{T_{\text{prev}}} w_t h_t(x) + \sum_{t=1}^{Q} w_t h_t(x) + b \right)_{\text{frozen}}
\]

Dealing with Outliers
Convex losses are sensitive to label noise
Loss functions

![Loss functions graph](image_url)
Non-monotonous relationship between objective value and training error
Training with Non-Convex Loss and Convex Regularization

\[ L(w) = \sum_{s=1}^{S} L_q(m_s) \]

\[ L_q(m_s) = \min \left( (1 - q)^2, (\max(0, 1 - m_s))^2 \right) \]

\[ R(w) = \lambda \| w \|_2 \]

In addition to qubits needed to represent \( w \) and \( b \) we need \( O(S) \) ancillary qubits to represent non-convex loss!
Numerical experiments: non-convex regularization

<table>
<thead>
<tr>
<th></th>
<th>AdaBoost</th>
<th>Outer Loop Q = 64</th>
<th>Outer Loop Q = 128</th>
<th>Outer Loop Q = 256</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Test Error</strong></td>
<td>0.258 ± 0.006</td>
<td>0.254 ± 0.010</td>
<td>0.249 ± 0.010</td>
<td>0.246 ± 0.009</td>
</tr>
<tr>
<td><strong>Weak Classifiers</strong></td>
<td>257.8 ± 332.1</td>
<td>116.8 ± 139.0</td>
<td>206.1 ± 241.8</td>
<td>356.3 ± 420.3</td>
</tr>
<tr>
<td><strong>Reweightings</strong></td>
<td>658.9 ± 209.3</td>
<td>130.8 ± 65.3</td>
<td>145.6 ± 65.5</td>
<td>159.8 ± 63.7</td>
</tr>
<tr>
<td><strong>Training Error</strong></td>
<td>0.038 ± 0.054</td>
<td>0.185 ± 0.039</td>
<td>0.170 ± 0.039</td>
<td>0.158 ± 0.040</td>
</tr>
<tr>
<td><strong>Outer Loops</strong></td>
<td>11.9 ± 5.0</td>
<td>12.2 ± 4.6</td>
<td>12.6 ± 4.4</td>
<td></td>
</tr>
</tbody>
</table>
Improved generalization is expected from Vapnik-Chernovenkis theory

Vapnik-Chernovenkis dimension:

\[ VC_H = 2 \left( VC_{\{h_i\}} + 1 \right) (T + 1) \log_2 (e(T + 1)) \]

\[ H(x) = \sum_{t=1}^{T} h_t(x) \]

\( VC_{\{h_i\}} \) is the VC dimension of dictionary

Weak \( L0 \)-norm regularization with \( \lambda < \frac{2}{N} + \frac{1}{N^2} \) produces classifier with lower VC bound while not increasing the training error
Hardware experiments: Training a detector with D-Wave’s hardware

Car detector

Detect cars in images
Training data: 20,000 images with city street scenes
Calls to quantum hardware with 52-variable optimization problems

NIPS Demo 2009
Numerical experiments: non-convex loss

- Long-Servedio
- Mease-Wyner
- consistency
- covertype

Test error (%) vs. Label noise (%) for different datasets and loss functions:

- **Datasets:**
  - Covertype
  - Mushrooms
  - Adult9
  - Web8

- **Loss Functions:**
  - liblinear
  - logistic
  - square loss
  - t-logistic
  - sigmoid
  - probit
  - smoothed hinge

Graphs show the test error percentage across different label noise percentages for each dataset and loss function representation.
Numerical experiments: non-convex loss

- q liblinear
- logistic
- square loss
- t-logistic
- sigmoid
- probit
- smoothed hinge

**mushrooms**

**adult9**

**usps**

**web8**
Training with low-precision discrete weights yields significant levels of sparsity.
Challenges mapping to D-Wave hardware

**Problems**

- Objective needs to be quadratic.
- Qubit connectivity needs to adhere to Chimera graph.
- Parameter Warping

**Solutions**

- Higher order interactions can be reduced to quadratic by introducing auxiliary qubits.
- By stringing physical qubits together to form logical qubits that can be arbitrarily connected.
- Sculpt QUBO that can be represented by the hardware to fit samples from objective.
- Optimization becomes heuristic. Narrow minima may be overlooked.
- More qubits! 
- Indications that this may not scale!
- New approach!
Underexplored opportunity
Use D-Wave hardware as a sampling engine

Many machine learning methods are based on sampling from a probability distribution.

Non-zero temperature gives Boltzmann distribution:

\[ P(s) = \frac{\exp(-\beta E(s))}{Z} \]

Open quantum systems analysis: How does the probability distribution look if the adiabatic condition is violated?

Native hardware mode is sampling!

Approximate solutions by AQO not well studied

But post-PCP-theorem work indicates that approximate solutions can be used to optimally solve NP-hard problems in polynomial time (Dickson and Amin’11; Zuckerman, STOC’06)
Non-convex training of a binary classifier can be mapped to a QUBO format amenable to quantum optimization at "negative translation cost"

- Lower generalization error
- More compact classifiers
- Can better cope with outliers
- Less training cycles (when using boosting)

Experiments underway to run problem instances on D-Wave One quantum computer
Thank you!

Vasil S. Denchev\textsuperscript{1,2}  William G. Macready and Team\textsuperscript{3}
Hartmut Neven\textsuperscript{1}

\textsuperscript{1}Google
\textsuperscript{2}Purdue University
\textsuperscript{3}D-Wave Systems