# Applications of Quantum Annealing in Machine Learning

Conor McCormack

Professor Todd Brun
EE 520

The current state of quantum computing has left many of the field's most promising applications unrealized as the extraordinary levels of precision and control they require remain out of reach for all but the most trivial of problems. As of this writing, the most sophisticated quantum computers are able to manipulate systems of only a few dozen stable qubits and high-fidelity quantum gates, which falls orders of magnitude short of the complexity necessary for executing meaningfully-sized factoring procedures like Shor's famous algorithm or simulating quantum systems that could enhance our understanding of chemical interactions in drug development. Given these shortcomings in achieving universal quantum computation, quantum annealing has risen as an alternative quantum computing paradigm that leverages quantum-mechanical phenomena to solve specific classes of problems that aim to sample from or find a global minimum of a given objective function.

D-Wave Systems, a private Canadian company, has partnered with research institutions, including the University of Southern California, to pioneer the development of quantum annealers, including their most recent system that supports annealing over 5,400 qubits. Though the architecture of these systems evolves dramatically with each generation, the essential formulation of the annealing process remains the same. In every anneal, we attempt to adiabatically evolve the Hamiltonian $H$ associated with the system in such a way that it is left in a ground state of $H$ that represents a minimum energy solution to an objective function. This Hamiltonian can be decomposed into the sum of two Hamiltonians representing the initial state and objective function which are mutually exclusive at the beginning and end of the anneal.

The initial state of the system is usually an easily-achievable quantum state like a superposition of $|0\rangle$ and $|1\rangle$ for all qubits, as represented by the summation term of the "Initial

Hamiltonian" in Figure 2. Conceptually, these superpositions imply that at the start of the anneal, all configurations of the system are equally likely.

The objective function is most commonly the Hamiltonian of an Ising model, represented in Figure 1 below. In the Ising model $J_{i,j}$ is an interaction coefficient or measure of qubits *i* and *j's* likelihood to align in similar or opposite spins and $h_i$ is a "bias" for qubit *i*, which can be set along with $J_{i,j}$ to describe the function to be optimized or sampled from. Much of the practical application of quantum annealing arises in how easily $H_{ising}$ can be translated to a quadratic unconstrained binary optimization problem (QUBO), as shown in Figure 1.1, where the up and down states of the standard Z basis for qubit *i* can be considered as the binary values {1, 0} for bit $x_i$, respectively. Finding minimal energy configurations of our qubit spins is equivalent to minimizing the objective function $C$ in Figure 1.1.

$$H_{ising} = \sum_{i} h_i \hat{\sigma}_z^{(i)} + \sum_{i<j} J_{i,j} \hat{\sigma}_z^{(i)} \, \hat{\sigma}_z^{(j)}$$

Figure 1. Hamiltonian of an Ising Model

$$C = \sum_{i} b_i x_i + \sum_{i<j} a_{i,j} x_i \, x_j$$

Figure 1.1. Conventional QUBO cost function with linear biases $b_i$ and quadratic terms $a_{i,j}$ for bit values $x_i$.

$$\mathcal{H} = \underbrace{-\frac{A(s)}{2}\left(\sum_{i} \hat{\sigma}_x^{(i)}\right)}_{\text{Initial Hamiltonian}} + \underbrace{\frac{B(s)}{2}\left(\sum_{i} h_i \hat{\sigma}_z^{(i)} + \sum_{i>j} J_{i,j} \hat{\sigma}_z^{(i)} \hat{\sigma}_z^{(j)}\right)}_{\text{Final Hamiltonian}}$$

Figure 2. Decomposition of the system Hamlitonian into the "initial" and "final" Hamiltonians.

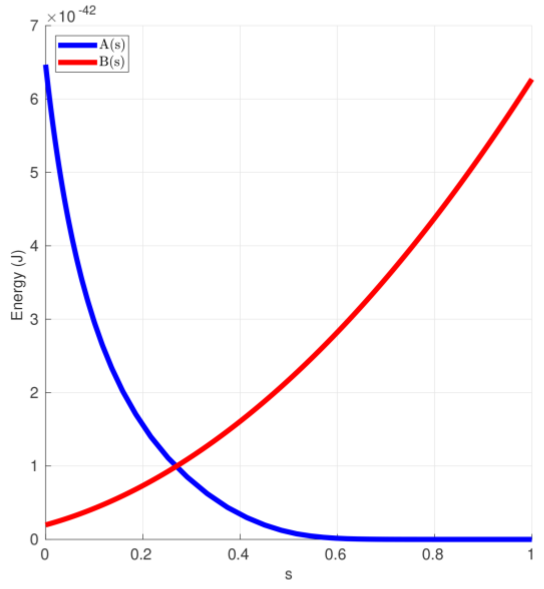


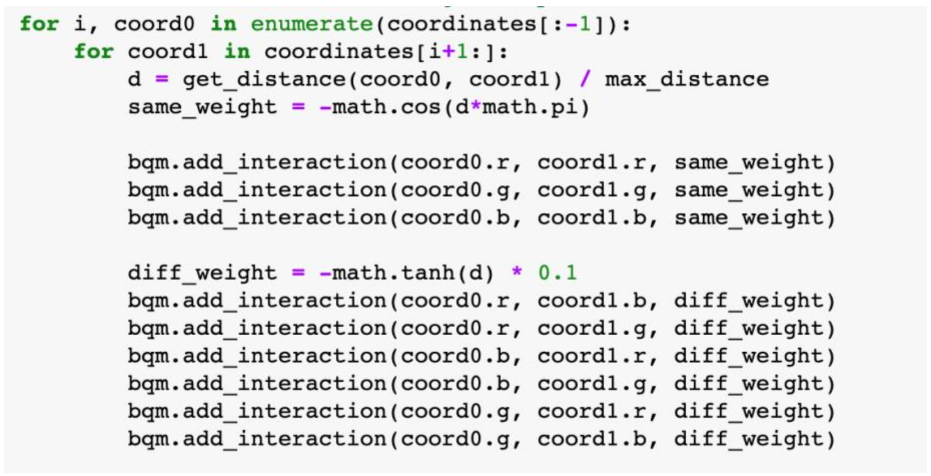
Figure 3. A real example of how the D-Wave system phases out the intial Hamiltonian and introduces the final Hamiltonian. S is a time scale determined by the adiabatic criterion which bounds how slowly *H* must change.

Associated with each of the initial and Ising, or “final” Hamiltonian is a function which controls the influence of each Hamiltonian throughout the anneal. Though many choices for these functions exist, it is required that A(1) and B(0) both approach 0 energy and A(0) and B(1) be approximately equal high energies. In this way, the initial Hamiltonian is replaced by the final Hamiltonian over the duration of the anneal. Given that this change meets the criteria for adiabaticity and the system is sufficiently isolated from the external world, we expect with decent probability that the final state of the system is left in the desired solution, a ground state of $H_{ising}$.

As the quality of quantum annealers improves, one of the most important challenges for researchers is the discovery of problems for which these machines might provide a considerable advantage compared to classical approaches. According to D-Wave, among the most popular of these are combinatorial optimization problems like those in the NP-hard complexity class, including the Traveling Salesman Problem, and selected problems in machine learning that either take random low-energy samples from the annealer to learn patterns in the cost function or those that can be mapped to a standard QUBO.

With my own academic interest in machine learning in mind, I explored the feasibility of cluster analysis using a simple QUBO mapping on the latest D-Wave system. Cluster analysis is commonly used to associate “instances” or observations in a dataset with each other in representative “clusters” that both group similar instances with each other and distinguish and separate dissimilar ones. More formally, we would like to minimize within group variance and maximize outgroup variance. As suggested by D-Wave, if interested in understanding a certain real estate market, one might be interested identifying similar types of homes or neighborhoods from a dataset containing quantitative records of houses in that market. Without any predetermined groupings, using a cluster analysis technique would hopefully return clusters of similar homes from your data that could inform your decision making.

To achieve this task as a QUBO, I followed and refactored a straightforward example from D-Wave’s GitHub account fit for forming a fixed number of three clusters. In this formulation of the problem, I allocate three qubits on the D-Wave system for every instance in the dataset, where each qubit represents membership in a cluster. If for example, the minimum energy anneal returned the states $\{1, 0, 0\}$ (equivalently $\{\uparrow, \downarrow, \downarrow\}$) for a data point, this would indicate that the instance belongs exclusively to the first cluster, as we should expect. To ensure this exclusivity, I also set a constraint on the valid arrangements of qubits in which exactly one of the three membership qubits is up and the others are down (i.e. ($\{1, 0, 0\}$, $\{0, 1, 0\}$, $\{0, 0, 1\}$).



```
for i, coord0 in enumerate(coordinates[:-1]):
    for coord1 in coordinates[i+1:]:
        d = get_distance(coord0, coord1) / max_distance
        same_weight = -math.cos(d*math.pi)

        bqm.add_interaction(coord0.r, coord1.r, same_weight)
        bqm.add_interaction(coord0.g, coord1.g, same_weight)
        bqm.add_interaction(coord0.b, coord1.b, same_weight)

        diff_weight = -math.tanh(d) * 0.1
        bqm.add_interaction(coord0.r, coord1.b, diff_weight)
        bqm.add_interaction(coord0.r, coord1.g, diff_weight)
        bqm.add_interaction(coord0.b, coord1.r, diff_weight)
        bqm.add_interaction(coord0.b, coord1.g, diff_weight)
        bqm.add_interaction(coord0.g, coord1.r, diff_weight)
        bqm.add_interaction(coord0.g, coord1.b, diff_weight)
```

Figure 4. Calculating pairwise distances and setting interaction terms to reward similar instance pairs and penalize dissimilar ones.

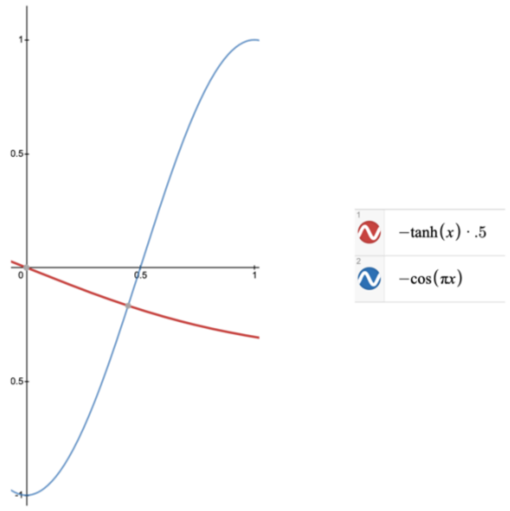


Figure 5. Weighting functions for interactions.

As shown in Figure 4, I then iterate over all pairs of instances in the dataset and calculate simple Euclidean distances for each pair. I use these distances as weights in the $a_{i,j}$ quadratic interaction terms in the QUBO problem. These distances are fed through $-\cos(d * \pi)$ and $-\tanh(d) * 0.1$ for same-cluster and different-cluster interaction terms respectively in order to reward similarity between instances with low energy values and penalize dissimilar pairs.

To test this on a small example, I sampled three values each from three 2D multivariate gaussians. With three qubits for each of the nine data points and fully connected interactions between all of them, we have 27 qubits with 301 couplings between them.

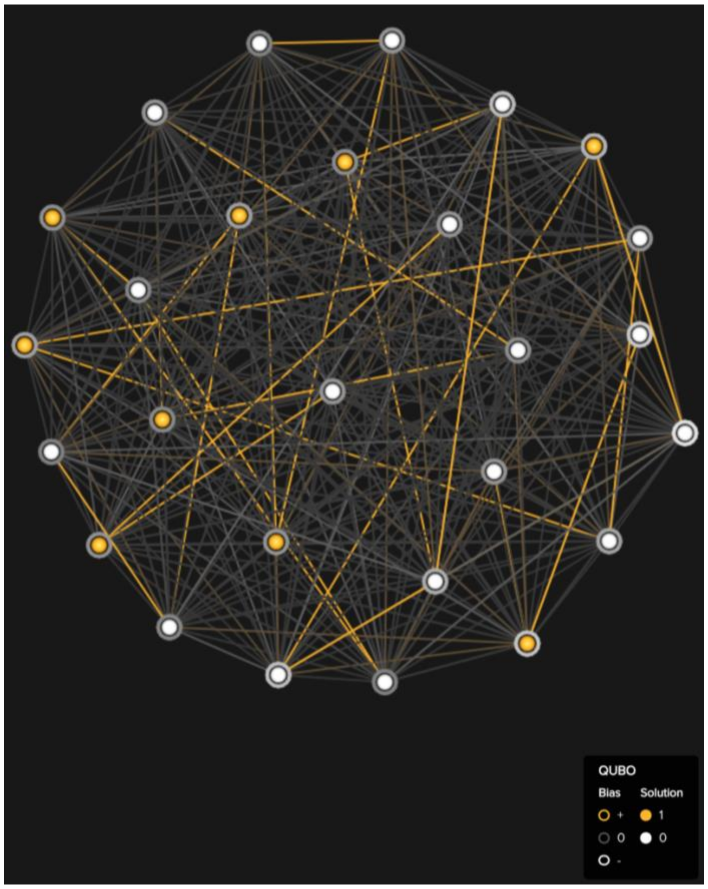


Figure 6. Full connected graph of 27 qubits from D-Wave's Problem Inspector tool. Each vertex is a qubit and every gold vertex indicates cluster membership.

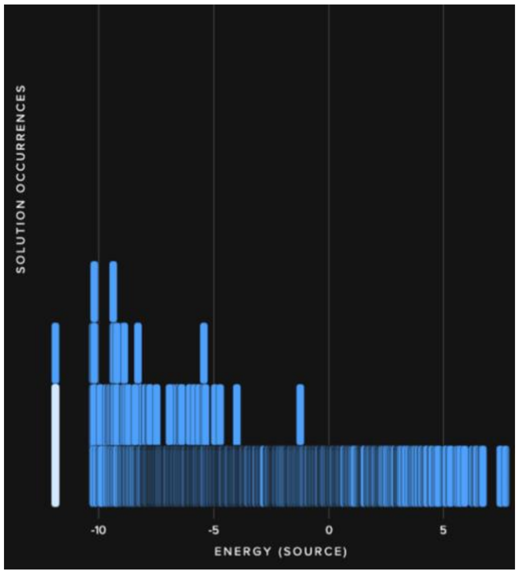


Figure 7. Histogram of energies from 1000 anneals.

I was rather pleased with this test, as the annealer seemed to create decent clusters that were fairly similar to the source distributions. However, success in this approach seemed to depend moderately on how close the means of each multivariate gaussian were placed to each other, with the three evenly spaced producing the most consistent results.

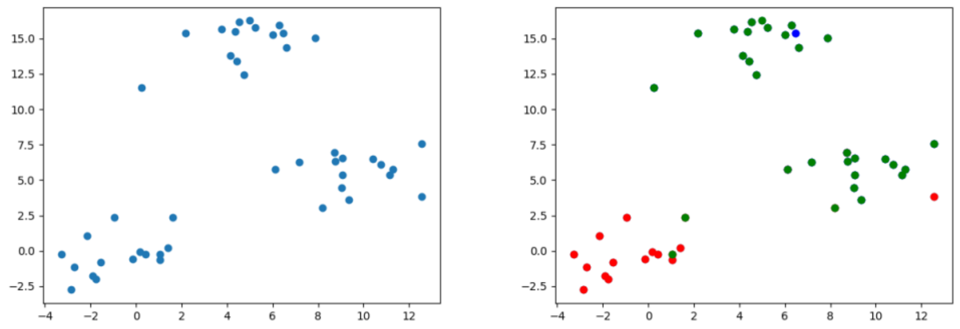


Figure 8. (Left) Unlabeled data from three multivariate Gaussians.
(Right) Clusters produced by D-Wave Advantage.

Moreover, as I scaled the problem to include more instances, I found that results became much noisier. Above in Figure 8, there are 45 points from three rather distinct gaussian distributions before and after clustering. The best result from the D-Wave system seemed to completely ignore one of the distributions in the clustering process.

Motivated by these initial results, I also attempted to apply this approach to cluster analysis to a classic statistical learning example, the Iris dataset, which is four-feature record (cm measures of sepal length, sepal width, petal length, petal width) collected by Ronald Fisher of 150 different flowers of three species: Iris Setosa, Iris Versicolour, and Iris Virginica. I began by standardizing each feature in order to give each a similar distribution with mean 0 and unit variance, which is important in a distance-based clustering method so as to insure no feature has an oversized influence on the result.

Applying the same constraints, distance metrics, weights, and interactions to this data, D-Wave returned decent results, though unfortunately they were nowhere near the performance of classical implementations like sci-kit's KMeans package. In assigning flowers to clusters in with their original species, this approach performed with an accuracy of around 0.56, which is almost acceptable. However, I found that the results were extremely noisy even on small subsets of the original data. Often, D-Wave failed to even return solutions that fit the exclusivity constraints, meaning that one flower was being assigned to multiple clusters or species. To address this, I attempted a similar test with the same subset of the data, but only considering two of the original features. Unfortunately, this failed to decrease the noise in any meaningful way and decreased the assignment accuracy along the way.

Overall, I was pleased with the both the results from each of my experiments as well as the ease of use in navigating the D-Wave platform. Although I was unable to achieve this myself, there seem to be a great opportunity for improving the performance of this approach by more appropriately fitting the data to the topology of D-Wave's QPU. The most common error I encountered when sending jobs to the D-Wave platform was related to excessive "chain length". Chains are D-Wave's workaround to embedding problems on their processor that don't naturally fit very well. Though it would be convenient, the architecture of their processor does not make it possible to couple every qubit to each other nor does it allow certain topologies, like a $K_3$ cycle without "chaining" together additional qubits. By relying on chains, we introduce multiplicative errors that affect the quality of the results

In my experimentation, I relied exclusively on the default embedding software that D-Wave provides. I suspect that additional noise from chaining and non-optimal embeddings are exactly what diminished the quality of the clusters as I increased the number of instances and the dimensionality of the data. Using the problem inspector software, I noticed that the problems I sent to the QPU often used chains of qubits with lengths in excess of 10 qubits. By moving away from fully-connected interactions or using a scheme that requires few qubits altogether, I believe mapping a clustering algorithm as a QUBO would be a totally feasible task for a quantum annealer to handle.

## Sources

Brun, Todd. *Quantum Annealing*. Nov. 2020. https://blackboard.usc.edu/bbcswebdav/pid-7169560-dt-content-rid-35507602_2/courses/20203_ee_520_30812/annealing.pdf. Asynchronous Lecture and PowerPoint Presentation.

D-Wave Systems. *What is Quantum Annealing?* D-Wave System Documentation. https://docs.dwavesys.com/docs/latest/c_gs_2.html.

D-Wave Systems. *Programming the D-Wave QPU: Setting the Chain Strength*. Apr. 2020. https://www.dwavesys.com/sites/default/files/14-1041A-A_Setting_The_Chain_Strength.pdf.

Jonson, Mark, and Lanting, Trevor. *Quantum annealing with manufactured spins*. Nature, May 2011.https://www.researchgate.net/publication/51117464_Quantum_annealing_with_manufactured_spins.

Fisher, Ronald. Iris Data Set. 1936. UCI Machine Learning Repository. https://archive.ics.uci.edu/ml/datasets/iris.

Hastie, Trevor, Tibshirani, Robert, and Friedman, Jerome. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Spring, Aug. 2008.

Neukart, Florian, Von Dollen, David, and Seidel, Christian. *Quantum-Assisted Cluster Analysis on a Quantum Annealing Device*. Frontiers in Physics, Jun. 2018. https://www.frontiersin.org/articles/10.3389/fphy.2018.00055/full