Quantum-Classical Sentiment Analysis

Mario Bifulco[1] Luca Roversi[1]

[1]University of Turin, Computer Science Department

EQAI 2024

Abstract

We first experiment with applying a hybrid classical-adiabatic quantum classifier (HCAQC) for sentiment analysis and compare it to the classical classifier and state-of-the-art Transformer architecture. HCAQC is worse than Transformer in classification, but it converges to a good solution much more quickly. Secondly, the work explores how to address a bottleneck of HCAQC. The results show how effective HCAQCs can be developed, making greater use of the quantum component.

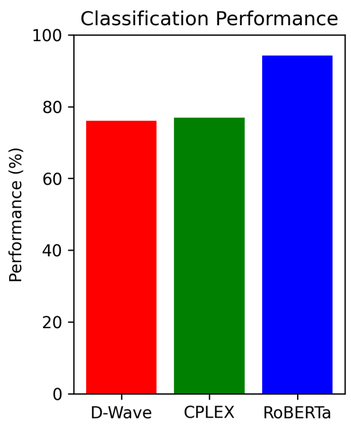
Quantum Support Vector Machine for Sentiment Analysis

We explore the potential of **Adiabatic Quantum Computing (AQC)** for **Natural Language Processing (NLP)**. The focus is on **Binary Sentiment Analysis (BSA)**. Our steps are:

1. Select the TweetEval dataset[1];
2. Transform the text into vectors using SentenceBert[2];
3. Encode the problem of learning a **Support Vector Machine (SVM)** using the D-Wave hybrid solvers;
4. Assess the quality. We compare it with:
   - An SVM implementation via CPLEX classical solver;
   - The Transformer RoBERTa[3].

QSVM Results

Legend of labels: **D-Wave** represents the quantum solution, **CPLEX** represents the classical implementation of SVM, **RoBERTa** represents the Transformer architecture.



Classification Performance

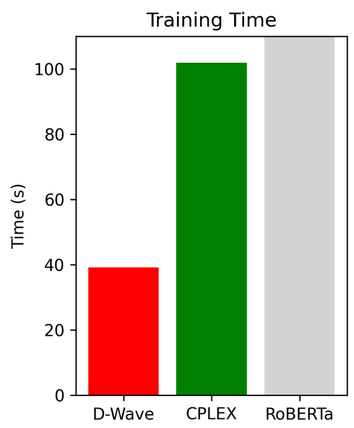
RoBERTa is almost 20% better than D-Wave and CPLEX, which classify 75% of cases correctly.

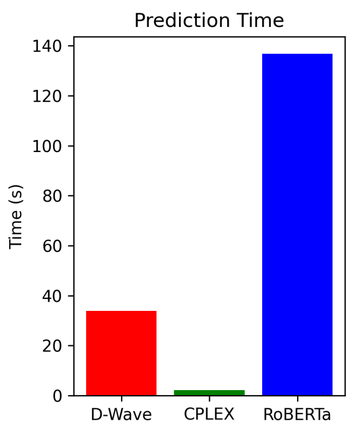
CPLEX is slightly better than D-Wave due to some limitations of the current hybrid solvers, which restrict the domain of the optimisation variables from real numbers to integers.

Training Time

The time required by D-Wave to find an optimal assignment is 60% less than that of CPLEX.

Although not available, it is reasonable to expect an even better result if compared to the time required by RoBERTa, which we can fairly expect to amount to several hours of training on high-performance machines.





Prediction Time

The complexity of the RoBERTa architecture also affects the time required for prediction.

The longer time D-Wave requires is due to the resolution methodology, which returns a set of optimal solutions from which a majority vote is taken during inference.

Experimentally, we find that it is possible to use only the first of the optimal assignments produced, reducing the time to values comparable to those of CPLEX.

Maximizing Quantum Boost

Experiments show that the D-Wave solver makes marginal use of the QPU: average contribution of **0.08%**. For greater performance, we run the problem directly on the QPU: bypass the opaque workflow D-Wave proprietary technology hybrid solvers.

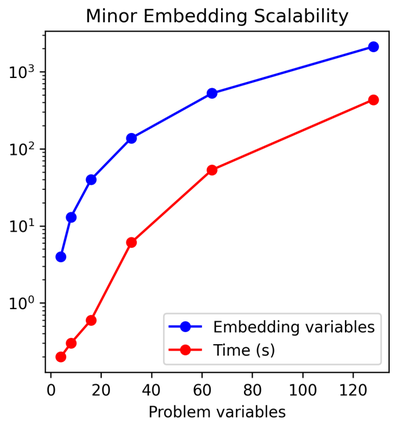
We transform the learning of an SVM into a problem that is solvable directly by the QPU. Among the series of required steps, the only one impacting performance is the calculation of a minor embedding[4].

QPU Bottleneck

Empirically, we identify a sweet spot for problems with 32 variables.

Beyond this size, the time required to find the minor graph can become dominant compared to the time needed to solve the problem.

Additionally, the number of qubits required to represent the problem graph becomes incompatible with the current QPU sizes (approximately 5600 qubits).



Homebrewing a Hybrid Solver

We design `QSplit`, a hybrid solver to increase the usage time of QPU. `QSplit` decomposes a QUBO problem into smaller QUBOS. `QSplit` relies on simple algebraic properties.

Given any QUBO matrix we recursively divide it into four parts as in Figure 1:

1. ULs and BRs are themselves QUBO matrices operating on a partition of the optimization variables;
2. UR retains the information linking the partitions UL_0 and BR_0 of the variables and we can safely transform it into an upper triangular matrix, namely a QUBO instance;
3. $\emptyset$ is a matrix composed entirely of zeros, so we ignore it.

The recursive subdivision can continue until the matrices reach a predetermined size, namely the Stopping dimension.

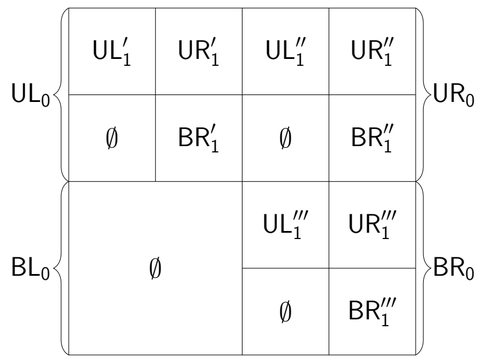


Figure 1. Two steps of recursive decomposition

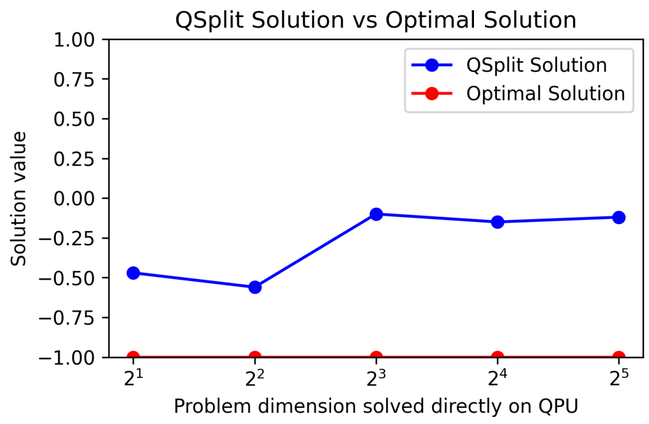
Example of one recursive step

Let us assume that we have classifications offered by UL_1', UR_1', BR_1'.

1. UL_1' and BR_1' are combined to generate conflict-free initial assignments called S_1;
2. The solutions from S_1 and UR_1' are combined, and conflicting variables are marked;
3. From the conflicting values, a QUBO problem is extracted, which is solved via QPU;
4. From the set of possible assignments, the k best distinct assignments are retained.

`QSplit` Results

We tested `QSplit` with problems from 128 variables, and the data collected showed the following behaviour **when decreasing the Stopping dimension**.

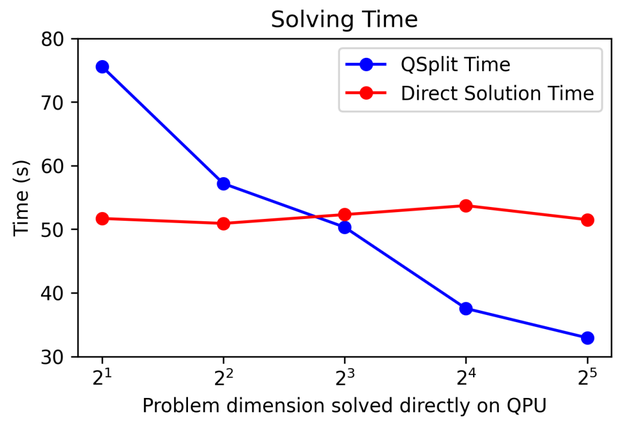


Performance

The quality of the solution improves, with the error decreasing from 45% to 25%.

Execution Time

The time required by `QSplit` increases, ranging from requiring 40% less time compared to direct resolution to requiring 50% more time.



Conclusion

We give evidence that quantum computing can bring tangible benefits over traditional methods to solve optimization problems. It is useful because:

- Converges much faster to the solution than classical methods;
- Requires limited hardware resources, allowing it to be used in embedded systems and personal computers.

`QSplit` is an experimental solution for handling large QUBO problems, different from those found in the literature[5]. The assignment produced can be improved by implementing more refined problem partitioning strategies[6], or by creating work pipelines capable of using a set of methods for finding the optimal assignment[7].

Bibliography

[1] Rosenthal Sara et al., "SemEval-2017 task 4: Sentiment analysis in Twitter".

[2] Reimers Nils et al., "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks".

[3] CardiffNLP, "Twitter-roBERTa-base for Sentiment Analysis".

[4] Jun Cai et al., "A practical heuristic for finding graph minors".

[5] Tameem Albash et al., "Solving large optimization problems with restricted quantum annealers".

[6] Sanavio Claudio et al., "Hybrid Classical–Quantum Branch-and-Bound Algorithm for Solving Integer Linear Problems".

[7] Michael Booth et al., "Partitioning Optimization Problems for Hybrid Classical/Quantum Execution".

Mario Bifulco

Email: mario.bifulco@edu.unito.it
LinkedIn: `linkedin.com/in/mario-bifulco/`
GitHub: TheFlonet/qsvm4sentanalysis

Luca Roversi

Email: luca.roversi@unito.it
Website: `di.unito.it/~rover`
ORCID: 0000-0002-1871-6109