Introduction

Below you will find an overview of the projects you can chose from for the mini-project. At the end of the mini-project, **you will have to turn in a final report**. **The deadline for the final report is 6th of January 2020.**

The final report should in general be between 5-10 pages long (it is group work, so everybody writes a part of the final report) and should be concise and to the point. For each project we provide literature to study and understand in detail, we provide a set of objectives and task, and we give a schematic of what the final report should look like. The rest of the tutorials are dedicated to helping you with the mini-project. Assistance with the theory will be provided next tutorial (5 Dec) and assistance with helping you set up implementations will be the tutorial thereafter (12 Dec), of course you can also reach us directly via email.

Every group member is expected to know the basic ideas of the papers relevant to the project, i.e., the problem solved, the algorithm used, cost (complexity) and applicability. Some group members can focus on complexity-theoretical and algorithmic details and applicability, others on implementation.

Projects 1-3 are a combination of both theory and implementation/experiments. Projects 4-6 are reading projects and are theoretical. Finally, projects 7 is only for groups of up to two people.

Evaluation

The grade will be based on the final report and an optional meeting after the final report has been handed in (not mandatory for all). There is always the possibility for extra credit.

Project 1. (*Topological data analysis*).
This project is about learning and implementing the quantum algorithm for topological data analysis by Lloyd, Garnerone and Zanardi (LGZ), as treated during the lecture of week 46.

Literature:

- `https://www.nature.com/articles/ncomms10138`.
- `https://arxiv.org/abs/1906.07673`.
- **Extra:** `https://link.springer.com/article/10.1140%2Fepjst%2Fe2012-01655-6`

Objectives and tasks:

- Learn and understand the LGZ algorithm and the underlying details.
  - What is the problem that is being solved?

  - How does the algorithm and its subroutines work?
  - What are the strengths and weaknesses of this algorithm?
  - What is the the cost (i.e., complexity) of the algorithm?
  - What real-world problems can this algorithm be used for?

- Provide a basic implementation of the LGZ algorithm and perform experiments.
  - Implement the algorithm in Qiskit (in which case we will provide some subroutines) or Cirq (in which case you will have to start from scratch).
  - Provide some test runs on (easy) examplary Hermitian matrices.
  - Perform experiments on graphs and/or datasets.

The report should contain:

1. Description of the problem at hand.
2. Description and explanation of the algorithm and its subroutines.
3. Analysis of the complexity of the algorithm.
4. Description of your implementation and report of the test runs.
5. Discussion of experiments and applicability of the algorithm.

For a 10 and a paper: show whether measurement outcomes from the eigenvector register can be used for improved data analysis.

Project 2. (*Parameterized quantum circuits for supervised learning*). This project is about using parametrized quantum circuits (i.e., circuits where the gates depend on a set of parameters) as learning models in a supervised learning setting.

Remark. *There will be a short lecture on this topic in one of the next lectures.*

Literature:

- `https://www.nature.com/articles/s41586-019-0980-2`.
- `https://github.com/Qiskit/qiskit-community-tutorials/blob/master/artificial_intelligence/vqc.ipynb`
- `https://pennylane.ai/qml/app/tutorial_variational_classifier.html`

Objectives and tasks:

- Learn and understand the parametrized quantum circuit classifier.
  - What are the parametrized quantum circuits that are being used as the classifier?
  - How do we train the parametrized quantum circuit?
  - How do we use a (trained) parametrized quantum circuit to classify previously unseen data?
  - What are the strengths and weaknesses of this classifier?
- Provide an implementation of the parametrized quantum circuit classifier.

Option 1: Implement using Qiskit (the implementation itself is easier, however the runtimes are longer and therefore you can only run smaller experiments).

Option 2: Implement using Cirq (the implementation itself is harder, however the runtimes are faster and therefore you can run larger experiments).

Option 3: Implement using PennyLane.

- Train and test the parametrized quantum circuit classifier on some of the following datasets:
  - Artificial datasets such as the datasets in the original paper, the bars-and-stripes datasets, the half-moon datasets or others that test properties of the classifiers.
  - Real-world datasets.
  - Your choice of dataset (explain why you thought this was a good dataset for this classifier).

The report should contain:

1. Description of the parametrized quantum circuits and how to use them as classifiers.
2. Explanation of how to train the parametrized quantum circuits.
3. Analysis of why this could be a good or interesting learning model.
4. Description of your implementation and overview of the obtained results from the experiments.
5. Discussion of the experiments and applicability of the learning model.

For a 10 and a paper: derive a learning setting in which you can show that the parametrized quantum circuit classifier has a provable advantage over standard (deep) neural networks with a fixed number of parameters.

Project 3. (*QAOA for problem(s) of your choice*).
In this project you will learn about the "quantum approximate optimization algorithm" (QAOA) and use it on a combinatorial optimization problem of your choice.

Remark. *There will be a short lecture on this topic in one of the next lectures.*

Literature:

- `https://arxiv.org/abs/1411.4028`.
- `https://pennylane.ai/qml/app/tutorial_qaoa_maxcut.html`

Objectives and tasks:

- Learn and understand the QAOA algorithm.
  - What are combinatorial optimization problems?
  - How does QAOA find approximate solutions for these combinatorial optimization problems?
- Implement QAOA in Qiskit, Cirq or PennyLane.
- Use your implementation to solve a combinatorial optimization problem of your choice.
  - E.g., MaxCut, Portfolio optimization or Portfolio diversification.
- Compare and experiment with different optimizers using noiseless simulation.
- **(Extra)** Compare different optimizers under *noisy* simulation.

The report should contain:

1. Introduction to combinatorial optimization problems.
2. Explanation of the QAOA algorithm.
3. Basic complexity analysis (including a comparison between approximation algorithms and heuristic algorithms).
4. Description of your implementation and overview of the obtained results from the experiments.

5. Discussion of the experiments and applicability of the algorithm (if large quantum computers would be available).

For a 10 and a paper: find a way to prove that QAOA with $p = 3$ or higher cannot achieve an approximation ratio better than the Goemans-Williamnson algorithm for MaxCut (0.878..). For more information on this see section 5.4.5 of `https://arxiv.org/abs/1805.03265`.

Project 4. (*Quantum complexity-theory & BQP-completeness of LA problems).* In this project you will study and report on up to three papers that discuss what kinds of linear algebraic computational problems capture the full power of quantum computing (i.e., are BQP-complete).

Literature:

- `https://arxiv.org/abs/0811.3171`
- `https://arxiv.org/abs/quant-ph/0606179`.
- `https://arxiv.org/abs/quant-ph/0606229`.

Objectives and tasks:

- Learn and understand (quantum) complexity theory and BQP.
- Learn (both formally and informally) what BQP-completeness of a problem is.
- Get introduced to the BQP-complete problems discussed in the above literature.
- Study the proofs of BQP-completeness of at least two of the above problems.

The report should contain:

1. Introduction to complexity theory and BQP (including clean statements of definitions and a discussion of some of the main results/conjectures around BQP).
2. Discussion and definition of BQP-completeness of problems.
3. Statements and discussions of the BQP-complete problems discussed in the above literature.
4. Formulation and discussions the proofs of BQP-completeness of at least two of the above problems.

Project 5. (*Classical simulability of quantum circuits*).
In this project you will study and report on two papers, or rather, learn about two interesting concepts in quantum computing. Namely, you will learn about computation with *Clifford*-gates and computation with *magic states*.

Literature:

- `https://arxiv.org/abs/0811.0898`.
- `https://arxiv.org/abs/quant-ph/0403025`.

Objectives and tasks:

- Learn what it means for a family of quantum circuits to be classically efficiently simulatable (and compare this with what it means to solve a `BQP` decision problem).
- Learn how to classically efficiently simulate quantum circuits with gates that are restricted to the Clifford-gates.
- Learn how adding a resource (i.e., a type of pure quantum states called magic states) allows quantum circuits with gates that are restricted to the Clifford-gates to become universal.
- Study how this is important for fault tolerance in quantum computers.

The report should contain:

1. Introduction to (efficient) classical simulation of quantum circuits.
2. Discussion on Clifford-gates and how to classically efficiently simulate them.
3. Explanation of magic states and the T-gadget trick.
4. Proof of universality of Clifford-gates with magic states.
5. Discussion of the implications on implementations of quantum computers.

Project 6. (*Quantum computational supremacy).*
Lately there has been a substantial buzz generated around Google's claim of having achieved "Quantum supremacy" using their 53 qubit quantum chip. In this project you will understand what quantum supremacy entails, and develop your opinion on whether Google's claim is justified.

Literature:

- `https://arxiv.org/abs/1809.07442`.

- https://www.nature.com/articles/s41586-019-1666-5.

Objectives and tasks:

- Learn what the objective and definition of quantum computational supremacy is
- Learn about the most promising candidates for problems/tasks which may yield a quantum supremacy result
- Learn the basic complexity theoretic ideas behind the proofs of hardness of sampling problems

The report should contain:

1. Clear and rigorous definition of quantum supremacy (as rigorous as possible)
2. Decription of the problems which may yield a supremacy result, and a discussion on what open questions still remain (ideal, average, noisy cases)
3. Discussion of the basic complexity-theoretic ideas behind these problems
4. Argue whether Google achieved quantum supremacy.

Project 7. (*Your favourite algorithm*).
For this project you pick your favourite quantum algorithm (that we have not yet studied) from the "quantum algorithm zoo" and you analyse it, explain it and discuss its applications.

Literature:

- http://quantumalgorithmzoo.org/

Objectives and tasks:

- Learn a new quantum algorithm!

The report should contain:

1. Introduction to the problem.
2. Explanation and proof of correctness of the quantum algorithm
3. Discussion of the speedup over its classical counterparts.
4. Discussion of the applications of the quantum algorithm.

Project 8. (*Your own idea for a project*).
If you have an interesting project in mind, or you know of a topic in quantum computing that you are interested in and would like to study in more detail, then you can pitch this to us for your project. Besides the constraints that the workload of the project should be equivalent to the others and that the topic should be sufficiently related to quantum computing, we are interested in hearing your ideas and interests!