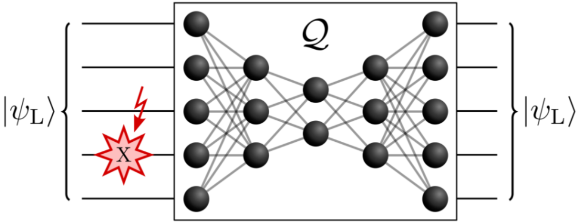
UNIVERSITY
OF OSLO

ML for measurement-free quantum error correction



Jørgen Vestly, Department of Physics

29th January 2026

About the project

Motivation: to automate QEC codes applied tos real NISQ hardware. Compare with classical, non-adapting codes. QEC remains one of the biggest challanges for scalable quantum systems. These systems aren’t merely quantum computers, but also quantum networks- and sensing; any quantum information carrier.

The project will consider a multitude of ML techniques such as CNN, RNN, Reinforcement learning, and maybe Transformers and GNN (*).

Background and literature

Measurement-free quantum error correction is interesting since there is no need to collapse the wavefunction (measure the state) to infer if an error has occured (stabilizer codes).

Meta-analysis on ML for QEC: https://arxiv.org/pdf/2412.20380
Paper on measurement-free QEC (LaRose): ht-
tps://journals.aps.org/prxquantum/abstract/10.1103/PRXQuantum.5.010333s2

The project pipeline - 1

Data: will likely used a generated supervised dataset, coming from simulations of quantum systems (random noise), with *recovery operations* as labels, based on *syndromes*. Some kind of data augmentation. Also, Morten has mentioned some sensor datasets.

The goal for the models (or agents) is to infer the right error-correcting strategy at each temporal step. Supervised models does this by classification, and RL does it by choosing an action and being rewarded (dynamic). The use of GNNs (*Graph Neural Networks*) can assist specifically in surface codes.

The project pipeline - 2

Training and testing will mostly be on simulated data. Although, it would be nice to include real NISQ hardware like IBM or LumiQ, to do final testing when models are refined. Note that the methods are classical, and the codebase is therefore simply Python, although if real NISQ is used, then some Qiskit/PennyLane might be included.

The project should focus on strengths and weaknesses on the pipeline (data, methods, training, results), and the ML methods themselves. Other papers might be uses as testing benchmarks.

Possible method pathways

(i) CNN+RNN: Use CNN as a feature extraction to feed to an RNN, introducing temporal QEC.

(ii*) GNN+Transformers: Use GNN as feature extractor (more complex), and use attention via Transformers. Unsure about this path as for now.

(iii) Unsupervised learning using reinforcement learning only, to find a dynamic decision process, maybe with a temporal layer (RNN)?

*Also thinking about only testing with no time, if there is not enough time to add time.