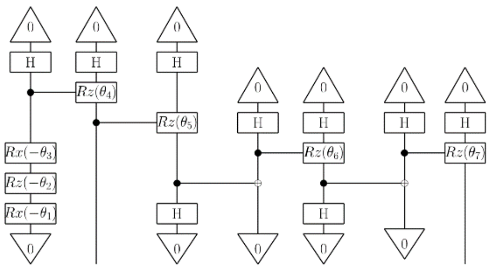
Project Outline by Henry Atkins for:

Transferring Classical Natural Language AI Embeddings to Quantum States

What is the most efficient method to encode word-meaning vectors on a quantum computer?

In their seminal paper ``Foundations for Near-Term Quantum Natural Language Processing", Coecke at al showed how to simultaneously represent meanings of words and grammatical structures of sentences in quantum circuits. This work has been followed up by others and extended from sentences to discourse structures by my supervisors Ian Kin Lo and Sadrzadeh, see https://arxiv.org/abs/2208.05393. This line of work uses quantum variational algorithms to translate grammatical types of words, such as nouns, adjectives and verbs, into parametrised Controlled Rotation gates, then machine learns the parameters. Below is a Quantum Circuit representing the discourse, i.e. two sentences that refer to each other, 'The cat broke the glass. It was fragile. ':



This methodology does an effective representation of the structures within and between sentences. It overcomes a conceptual shortcomings of the current (classical) deep neural network machine learning algorithms that do not directly model these structures. Nevertheless, it also is limited in its use of vacuous meaning vectors for words that have no informative data-related content.

My project aims to solve this problem by finding an optimum way of either:

- Finding an efficient way of learning the meaning vectors on a quantum computer.
- Finding an efficient means of encoding classically learnt vectors (from deep neural network algorithms such as BERT) into a circuit that may then a) further learn or b) follow this 'structural' encoding.

Learning the meaning vectors on a quantum computer may be very efficient due to the repeated use of the dot-product operation, a very efficient operation in quantum computers. So we may offer a fast approach to learning word vectors. However there are potential issues:

- Noise may reduce the time available to learn, meaning they cannot reach optimum values,
- Number of available qubits may not support the full vector space.

This means that we must consider the possibility of starting not from a quantum-ly learnt vector, but from a quantum encoded classically learnt vector. This brings with it a host of problems, but many of which may be reduced under certain assumptions:

- Lower vocabulary sizes mean exponentially smaller Q' circuits,
- Normalised vectors may be encoded via rotations,

- Vectors may be pre-processed via classical means, maximising Q’ circuit efficiency (for example for v word vectors from a whole vocabulary of V, all vectors could be realigned via Gram-Schmidt – a quantum efficient calculation – to represent meanings relative to each other. This means we reduce our vector length by V – v.)

A personal goal is to produce a python library that implements these encoding patterns, such that experiments may be done to test the theories. This would include 1-2 methods developed by me, and 3-4 methods from previous papers.

2022 November	Finish Literature Review and finalise plans.
December	**How previous researchers solved this problem** -Learn Lambeq & Pennylane - Recreate 2022Aug paper as base to work from. -Review papers in other fields (Q’ chemistry, Q’ finance) for encoding data and collate a list of: Conditions -> Consequences for data encoding efficiencies.
2023 January	**Recreating Previous Solutions and Literature Review** -Formally write up existing encoding methods on Overleaf as future reference -Begin to develop my own algorithm for encoding -Write a short summary of Classical NLP, identify quantum advantage (intro)
February	**Methods & Implementation** -Implement encoding efficient methods in a python library (to be reused). -Begin preliminary experiments on researched methods of encoding -Progress Report (due 3-Feb)
March	**My Contribution** -Run experiments (find optimal encoding) and benchmark against classical tests (measure number calls to some oracle, not time).
April	**My Contribution** -Additional research opportunities, left blank to fill at later date (spare time) -Idea here: See if a variational learner can learn classical pre-processing step-- -Draft of conclusions
May	**Outcomes of Research** -Finalise Literature Review and transform it to a draft Introduction -Document and publish python library (GitHub) and use this to draft methods. (Parallel) Literature Survey
June	**Write Thesis** -Finalise Introduction, Methods and Conclusion -Draft Discussions, considering dead ends and missed opportunities -Draft Abstract Submit Draft 2 months early for revisions and changes (Parallel) Case Study 1
July	**Finalise Thesis** -Finalise Discussions & Abstract section
August	(Parallel) Case Study 2 Submit Thesis