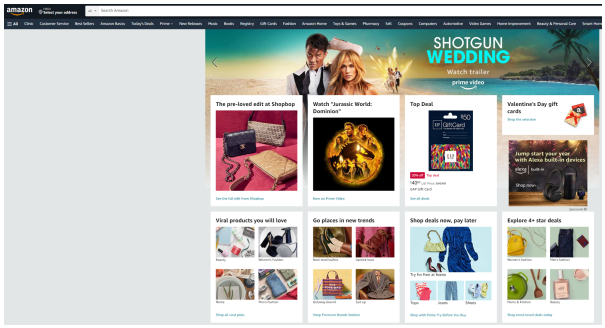
The hyper-realistic ~~ridiculous~~ background:

Mr. Feynman's spaceship crashed on an unknown planet. Many things exploded during the accident, including his suitcase. Oh no—his favorite t-shirts were in the suitcase!

Luckily Mr. Feynman's spacecraft is a cargo craft, which carries in the cargo bay a whole mountain of items including xbox, guitar, dry aged A5 Wagyu ribeye, California King sized bed, and of course, t-shirts. Sounds more like an exotic holiday then cast away yeah?

It's not to say Mr. Feynman doesn't still have a problem. Look at the size of the cargo bay.

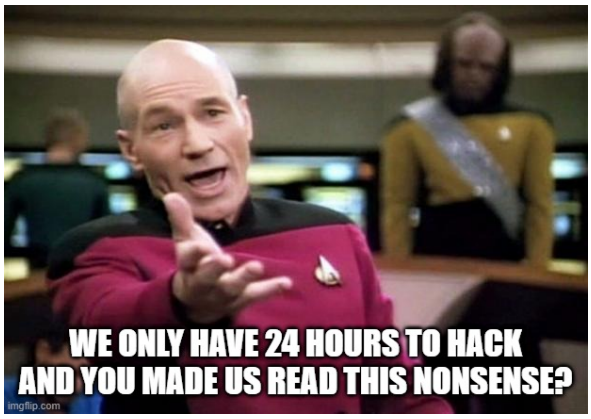
When the cargo bay AI system was working, retrieving an item was super easy—all he needed to do was go to the AI's web page:



But now the whole AI system is gone with the crash, Mr. Feynman has to look up everything manually. It's the third day, and t-shirts are still missing.

Luckily, he was also carrying an IonQ quantum computer, and it survived the crash. Mr. Fineman came up with a brilliant idea: having you design an image classification algorithm to help him

sort out useful items from the cargo bay. Yes, it's going to be a quantum image classifier, since the IonQ quantum computer is all that is left.



Ok here's what you need to do to help Mr. Feynman find t-shirts.

Part 1. Data loading (30%)

The first step is to encode images captured by camera into quantum circuits. This way the quantum computer can "see" the item.

You will be given an image dataset **(Fashion-MNIST)**. Your task is to make a data loading scheme that encodes the images into a quantum state as lossless as possible. Because Mr.Feynman doesn't have much classical computing resources, the encoded image should be

interpretable by simple measurements at the end of the circuit.

Preprocessing of the data is allowed, but the scoring is adjusted so that higher preference is given to solutions that can directly store as much of the state as possible in the wave-function (i.e. solutions which don't use much classical processing).

Submission and grading:

You will submit a python program (.py file) that explicitly contains the definition of three functions:

1. `encoder(image)` which creates/output a qiskit/cirq circuit
2. `decoder(histogram)` which returns an image.
3. `run_part1(image)`, which output a circuit (from `encoder()`) and an image(reconstructed using `decoder()`).

The `encoder(image)->qiskit` function takes an image as input and returns a qiskit circuit (**with no more than 16 qubits**) as the output.

The decoder(histogram)->image takes a histogram representation of the measurement of the cirq/qiskit circuit and returns an image.
The run_part1(image) explicitly does the following:

a. Use encoder(image) to convert image into a circuit
b. Use simulator(circuit) to simulate a circuit and get a histogram
c. Use decoder(histogram) to convert the histogram into the regenerated image
d. Output the image

Your score will be the average fidelity of image reconstruction (one minus the image_mse(image,regenerated image), the image_mse code is provided in the example_submission) times a noise overhead factor **(0.999^(number of 2-qubit gates in your circuit))**

Part 2. Classification (70%)

Now you will design a quantum circuit to classify the encoded image.

Submission and grading:
You will submit a `.pickle` file containing a single quantum classifier circuit, written with Qiskit or Cirq, and a .py file that defines `run_part2(image)`.

Your `.pickle` circuit will define a quantum classifier. Given an image, appending the classifier to your `encoder(image)` from part1 give you a histogram that can be processed into a label.

Your `run_part2(image)` should explicitly does the following:

a. Use `encoder(image)` to convert the image into a quantum circuit
b. Append the circuit with the classifier circuit loaded from the .pickle file
c. Simulate the circuit (encoded_image+classifier_circuit) and get a histogram
d. Run the provided histogram_to_label(histogram) to convert the histogram from c into a label
e. Output the label, and the circuit (encoded_image+classifier_circuit)

Your score will be the average accuracy (match between the actual label and predicted label) times a noise overhead factor **(0.999^(number_2q_gates)).**

Important Final Note: The goal of this problem is to explore quantum computing approaches to machine learning problems. To this end, the final score will take into account the degree of quantumness in the program. Preference will be given to creative solutions with a minimal amount of classical processing in the final solution.