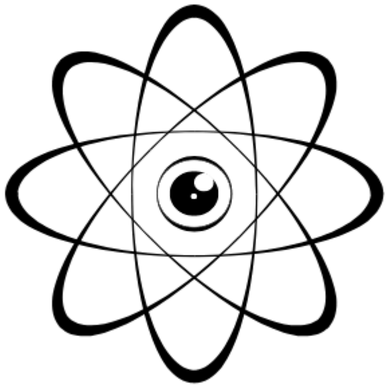
WOMANIUM | QUANTUM

QML for Conspicuity Detection in Production

What is quantum computing? (n.d.). Google Quantum AI. https://quantumai.google/static/site-assets/images/marketing/quantum-computing/hero-heading.jpg

‹ WOMANIUM | QUANTUM ›



Project QARS

‹ WOMANIUM | QUANTUM ›

QARS

Quantum Advanced & Relative Solutions

‹ WOMANIUM | QUANTUM ›

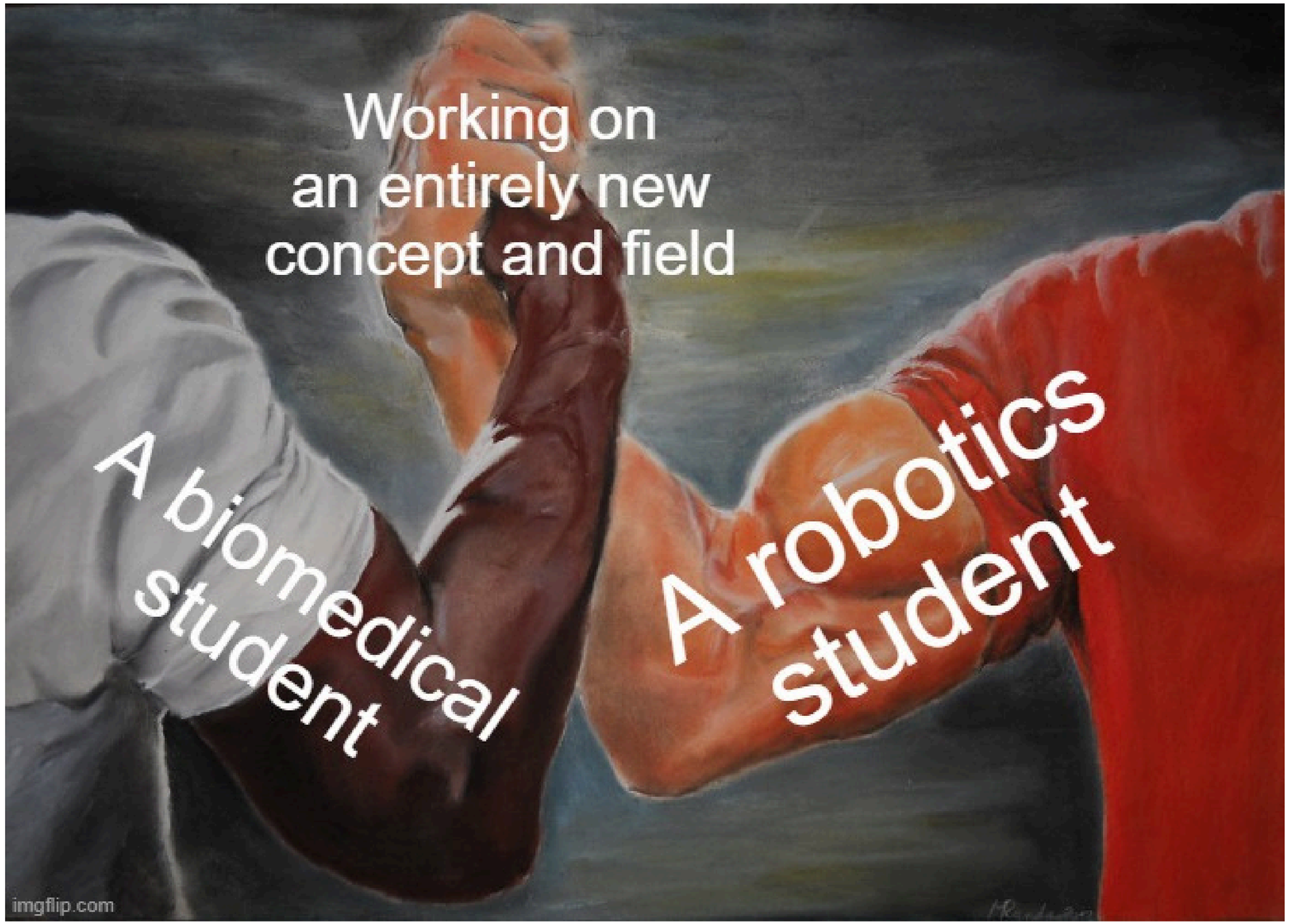
Who are we?

PARSA

BIOMEDICAL ENGINEERING
STUDENT

HANNA

ROBOTICS ENGINEERING
STUDENT


Working on
an entirely new
concept and field
A biomedical
student
A robotics
student
imgflip.com

‹ WOMANIUM | QUANTUM ›

QML Project Overview

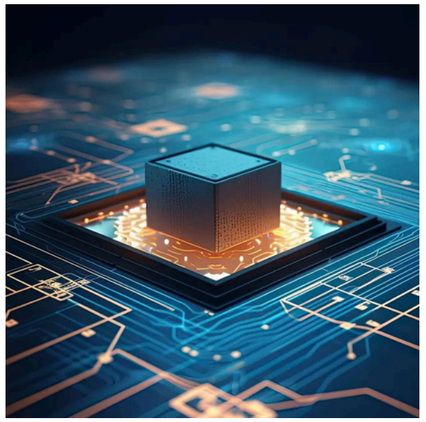


Image used under license from Shutterstock.com

Where to start?

Choosing the conspicuity detection project, we can work on familiar concepts while applying & developing new technologies

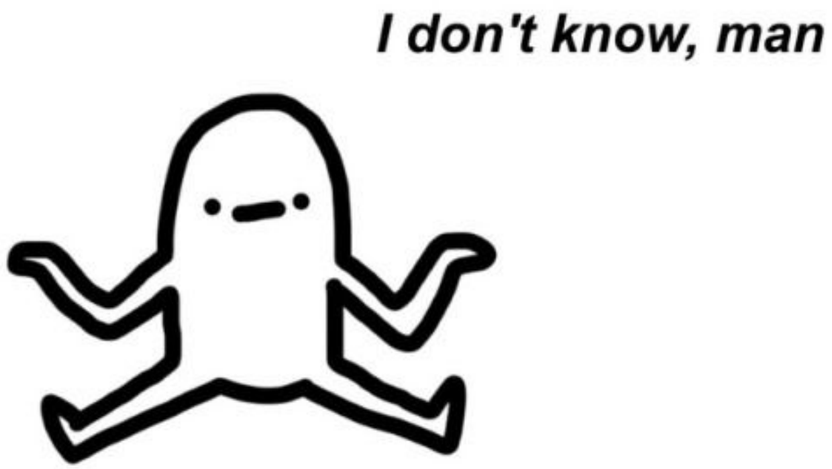
Image used under license from Shutterstock.com

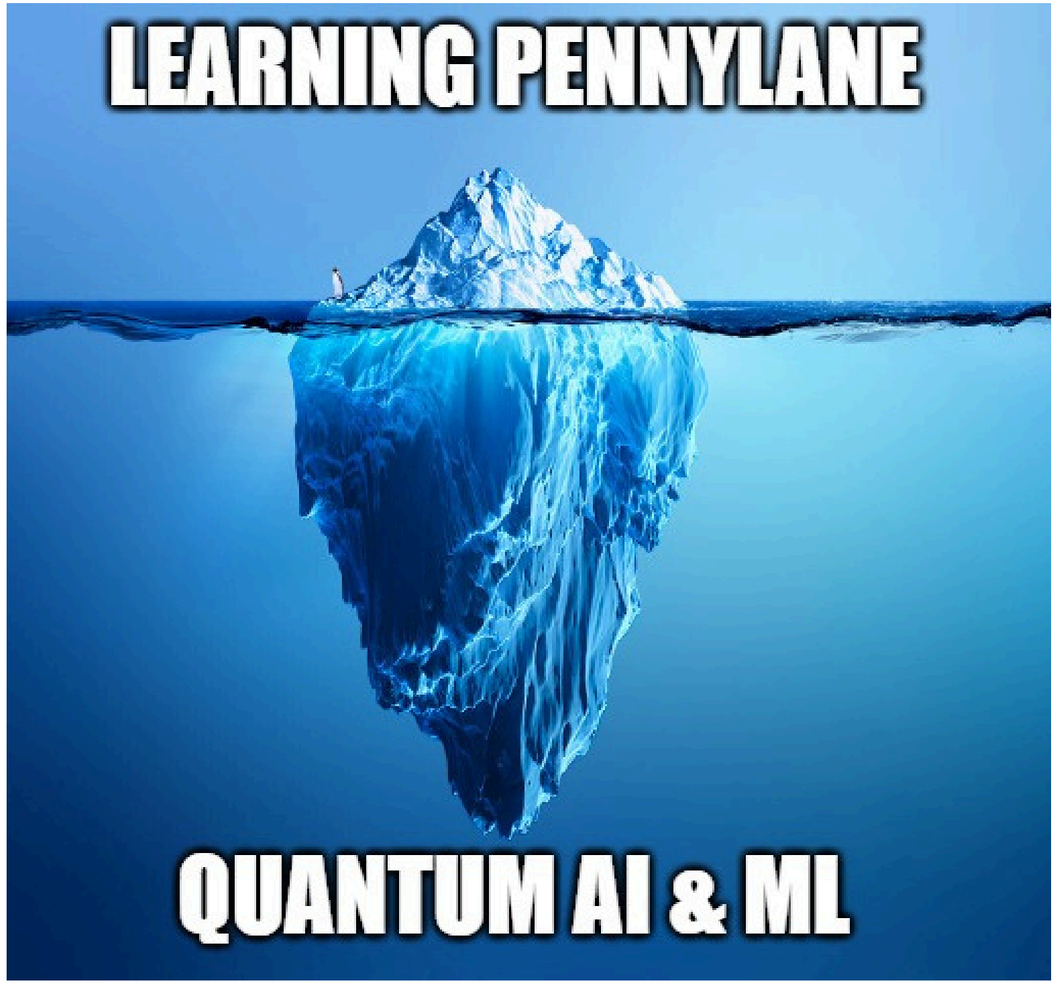
What is it?

In the Womanium Quantum + AI program, we learned efficiently detect conspicuity in industry lines could be of significant value.

Intoduction

“Pennylane”? What’s that?





‹ WOMANIUM | QUANTUM ›

Variational Classifier

Our first QML model Variational Quantum Classifier

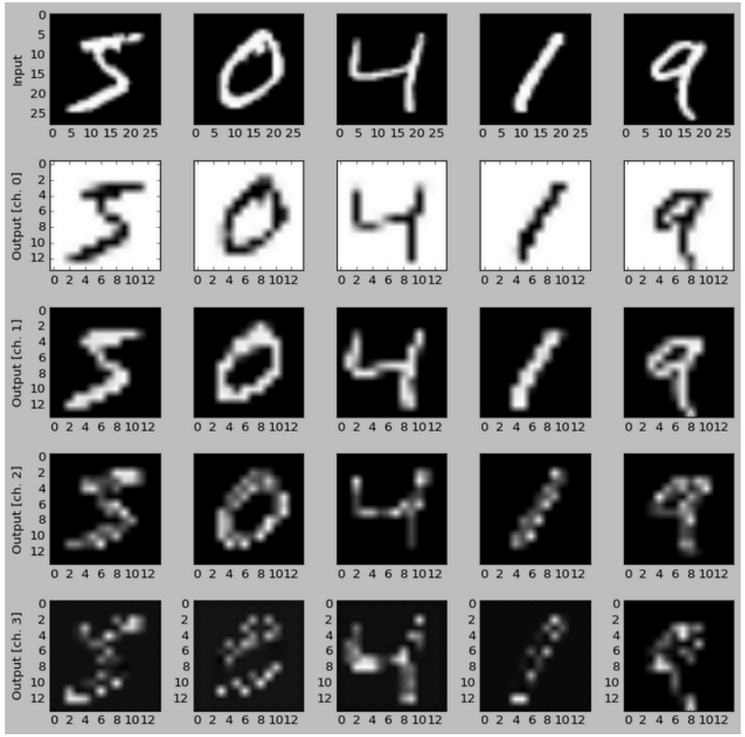
“After sweating over Pennylane, we suddenly fell into the rabbit hole of *Variational classifiers*, which is a fancy term for an algorithm that can distinguish 1 from 0 :P”

Parsa
Biomedical engineering Student

MNIST dataset

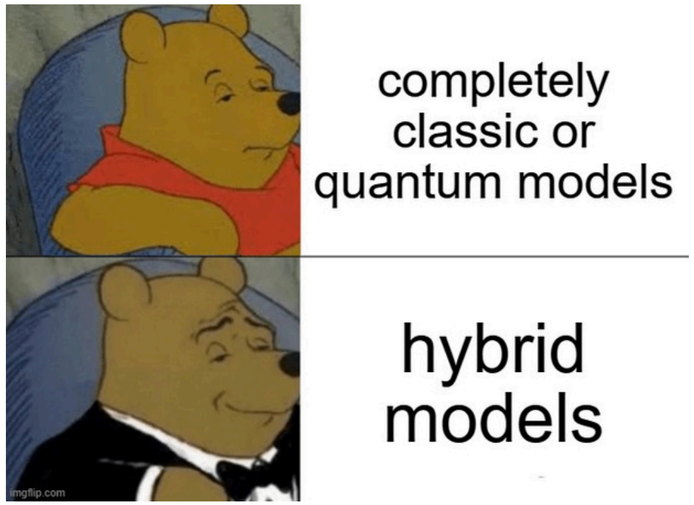
“Hello World!” of image classification… But quantum!

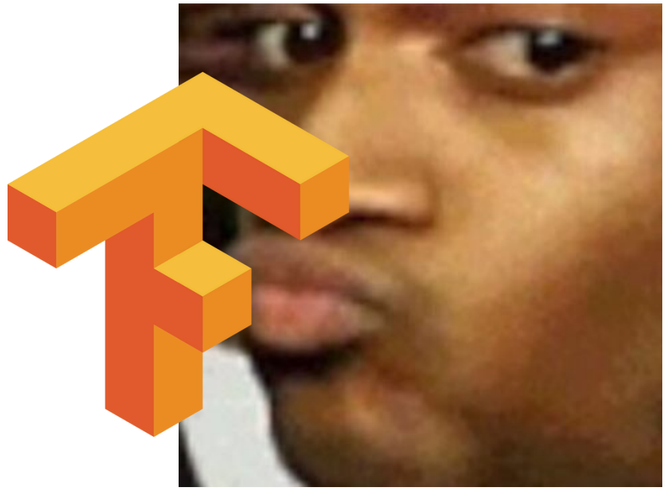
Ok now, how about we step up things a bit? From distinguishing 0 from 1, we jumped onto classifying numbers - drumroll please - the MNIST dataset!



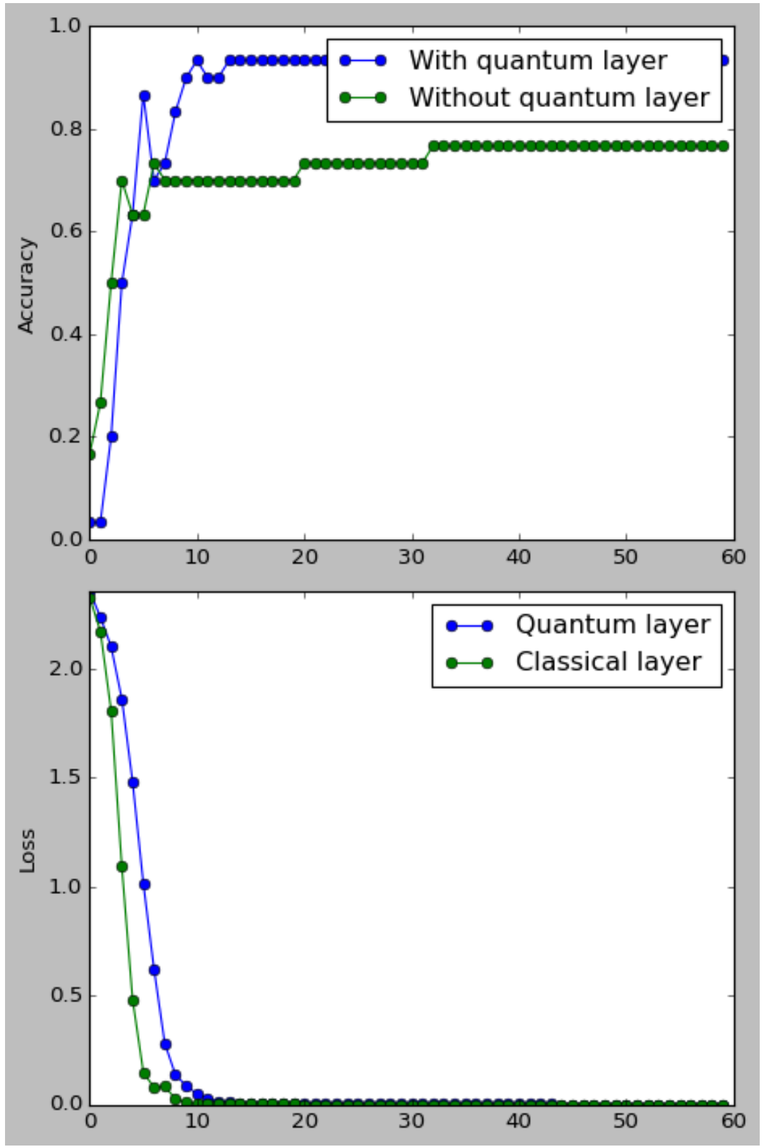
MNIST dataset

Results



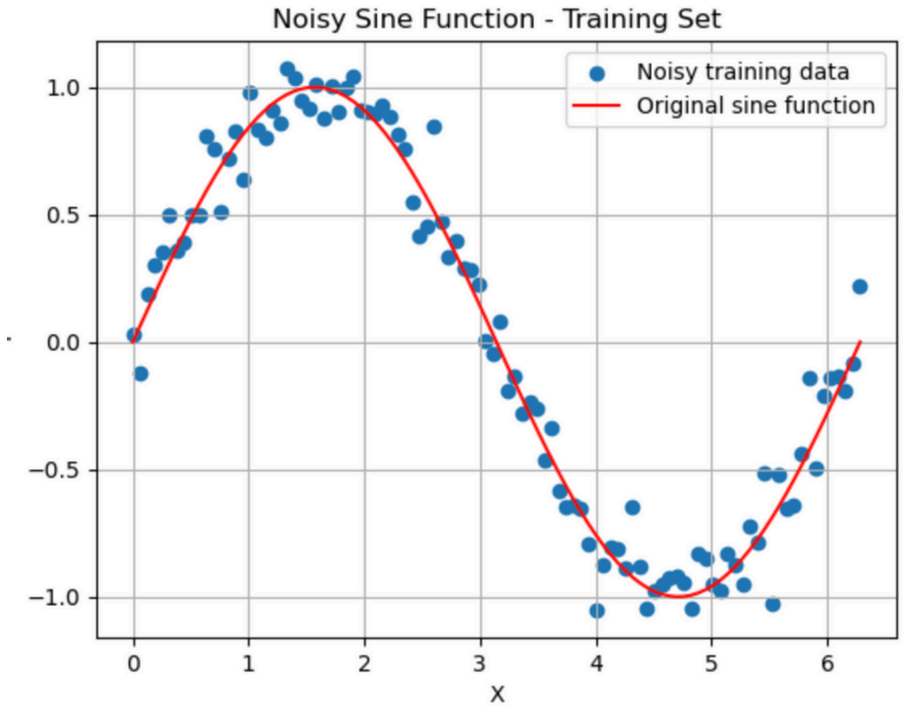


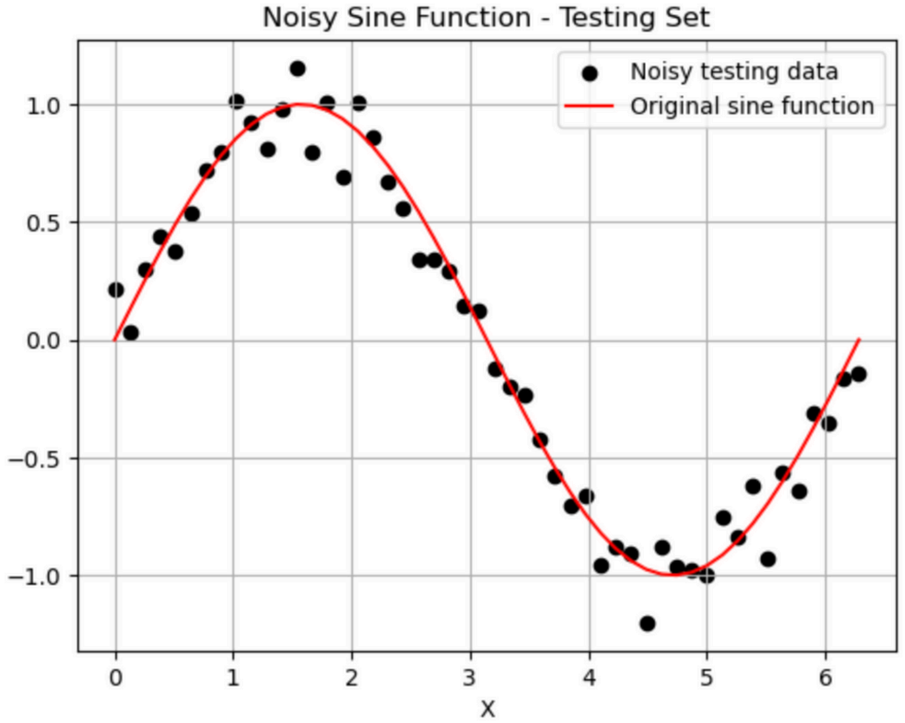
Tensorflow logo. Tensorflow. Retrieved from https://www.tensorflow.org/



Sine Function

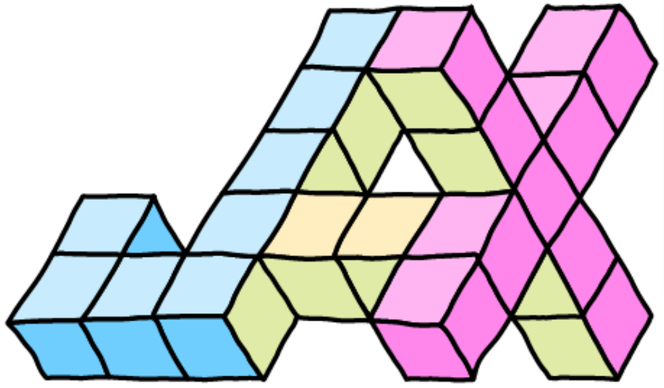
“Slightly challenging”? Nah bro, easy breeze





Sine Function

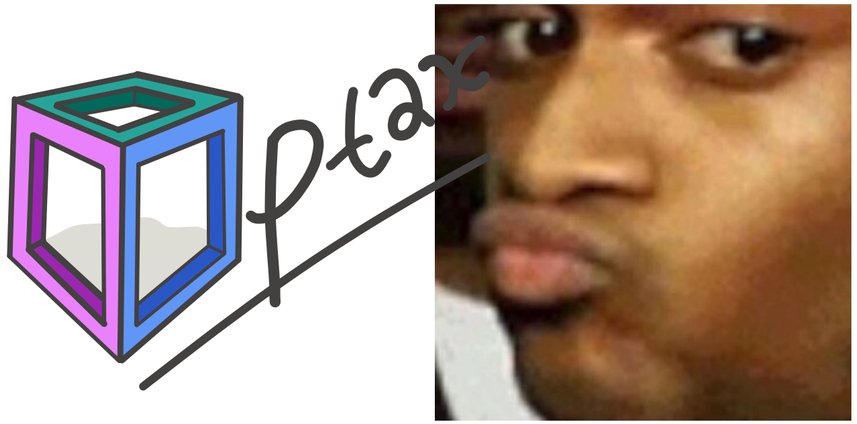
First meeting with JAX



Using JAX with PennyLane. (2021). Pennylane. Retrieved from https://pennylane.ai/qml/demos/tutorial_jax_transformations/.

Sine Function

Relationship with JIT and Optax

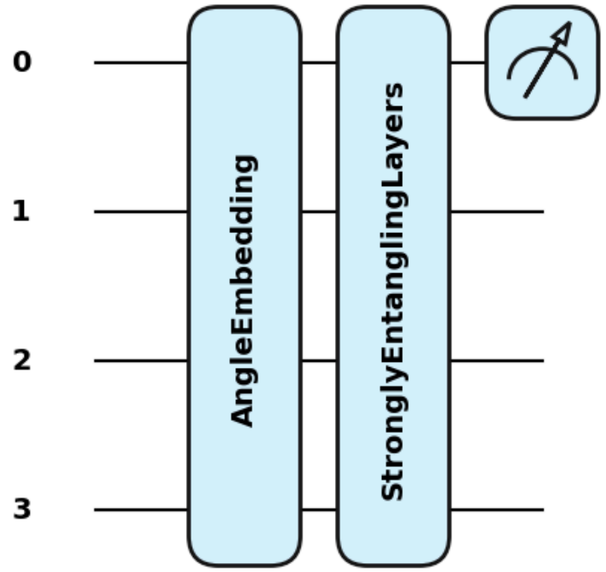


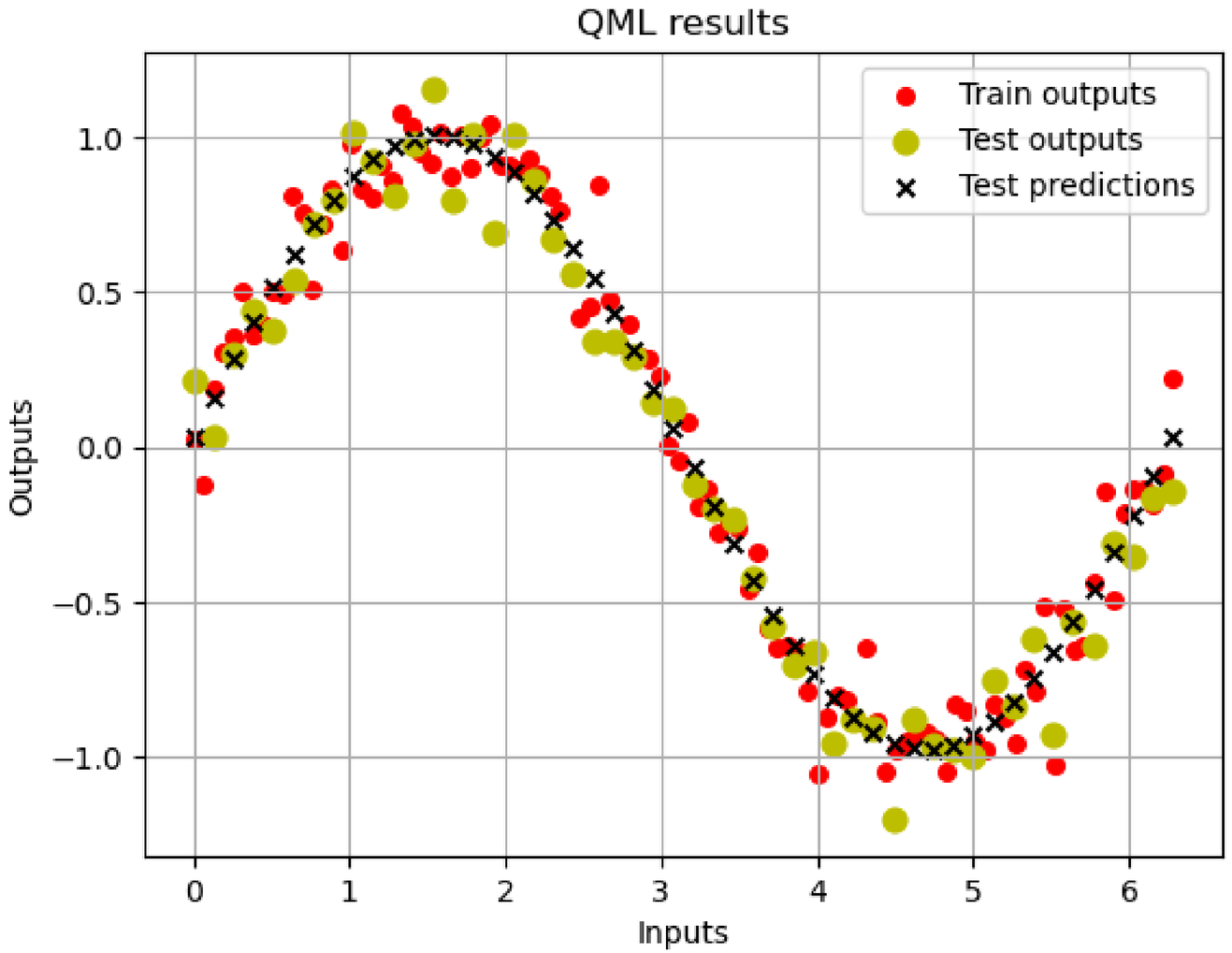
Optax logo. (2021). Optax. Retrieved from https://optax.readthedocs.io/en/latest/#.

```
opt = optax.adam(learning_rate=0.25)  # our optimiser

@jax.jit  # our update sequence which will also be JIT function to run faster
def update(params, opt_state, data, targets):
    loss_val, grads = jax.value_and_grad(loss_func)(params, data, targets)  # 
    updates, opt_state = opt.update(grads, opt_state)  # the gradients from th
    params = optax.apply_updates(params, updates)  # and finally we apply the 
    return params, opt_state, loss_val
```

Sine Function

Results





TIG Aluminium 5083


TASK 5, BOSS LEVEL
US
imgflip.com

AND WE GOT THE SOLUTION

TIG Aluminium 5083

Solution

```python
def load_images_from_folder(folder, img_size=(64, 64), limit=None):
    images = []
    labels = []
    total_files = sum([len(files) for r, d, files in os.walk(folder) if any(file.endswith(".png") for file in files)])
    processed_files = 0

    for subdir, _, files in os.walk(folder):
        for file in files:
            if file.endswith(".png"):
                img_path = os.path.join(subdir, file)
                img = load_img(img_path, target_size=img_size, color_mode='grayscale')
                img_array = img_to_array(img)
                images.append(img_array)
                labels.append(subdir)
                processed_files += 1

                # Print progress
                print(f"Processed {processed_files}/{total_files} files ({(processed_files / total_files) * 100:.2f}%)")

                # Check if the limit has been reached
                if limit and processed_files >= limit:
                    print(f"Reached limit of {limit} images.")
                    break
        if limit and processed_files >= limit:
            break

    print("Loading images completed.")
    return np.array(images), np.array(labels)
```

This function loads images from a specified directory, resizes them, converts them to grayscale, and stores them in a list along with their corresponding labels.

```python
@qml.qnode(dev)
def circuit(weights, biases):
    for i, layer_weights in enumerate(weights):
        qml.MERA(range(n_wires), n_block_wires, block, n_params_block, layer_weights)
        for j in range(n_wires):
            qml.RX(biases[i][j], wires=j)  # Add bias as an RX rotation
    return qml.expval(qml.PauliZ(0))

result = circuit(weights, biases)
print("Result from the circuit:", result)
```

```
Result from the circuit: -0.5811243139846571
```

Set up the quantum device with the specified number of qubits (wires).

The circuit applies a MERA architecture and RX rotations based on the biases and returns the expected value of the Pauli-Z measurement.

❮ WOMANIUM | QUANTUM ❯

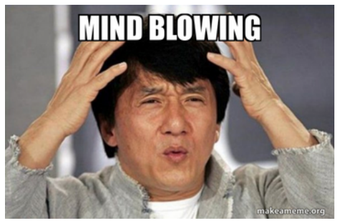
```
File /opt/anaconda3/lib/python3.11/site-packages/autograd/differential_operators.py:5
      3 from functools import partial
      4 from collections import OrderedDict
----> 5 from inspect import getargspec

/opt/anaconda3/lib/python3.12/site-packages/pennylane/_grad.py:158: UserWarning: Attempted to differentiate a function with no trainable para
meters. If this is unintended, please add trainable parameters via the 'requires_grad' attribute or 'argnum' keyword.
  warnings.warn(

ImportError: cannot import name 'getargspec' from 'inspect' (/opt/anaconda3/lib/python3.11/inspect.py)
```

```
Cell In[19], line 2
      1 # Test the model
----> 2 test_preds = [qml_model(x, trained_weights) for x
      4 # Plot the training loss
      5 plt.plot(range(len(losses)), losses, label='Train

NameError: name 'X test flattened' is not defined
```

```
NameError                                 Traceback (most
Cell In[17], line 2
      1 # Initialize optimizer
----> 2 opt = AdamOptimizer(0.01)
      3 weights = init_weights(n_layers=2, n_wires=4)
      5 # Training loop
```

```
ValueError                                Traceback (most
Cell In[34], line 34
     32     print(test_df.columns)
     33 else:
---> 34     raise ValueError("Data not loaded correctly."
     36 # Handle missing values
     37 train_df.fillna(method='ffill', inplace=True)

ValueError: Data not loaded correctly.
```

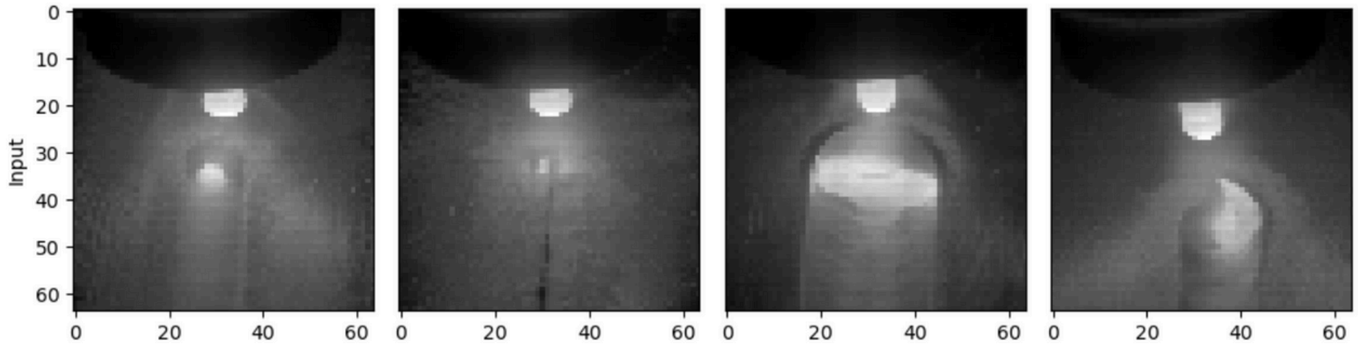
```
ecent call last)
workflow/execution.py:523, in execute(tapes, device, gradient_fn, i
ache, cachesize, max_diff, override_shots, expand_fn, max_expansion

erfaces'

The above exception was the direct cause of the following exception:

QuantumFunctionError                      Traceback (most recent call last)
Cell In[126], line 20
     18 x = np.random.random(4)
     19 weights_jnp = [jnp.array(w) for w in weights]
---> 20 result = circuit(weights_jnp, x)
     21 print("Result from the circuit:", result)

File /opt/anaconda3/lib/python3.11/site-packages/pennylane/workflow/qnode.py:1098, in __call__(self, *args, **kwargs)

File /opt/anaconda3/lib/python3.11/site-packages/pennylane/workflow/qnode.py:1052, in _execution_component(self, args, kwargs,
s)

File /opt/anaconda3/lib/python3.11/site-packages/pennylane/workflow/execution.py:525, in execute(tapes, device, gradient_fn, i
form_program, config, grad_on_execution, gradient_kwargs, cache, cachesize, max_diff, override_shots, expand_fn, max_expansion
transform, device_vjp)

QuantumFunctionError: jax not found. Please install the latest vers                    he 'jax' interface.
```

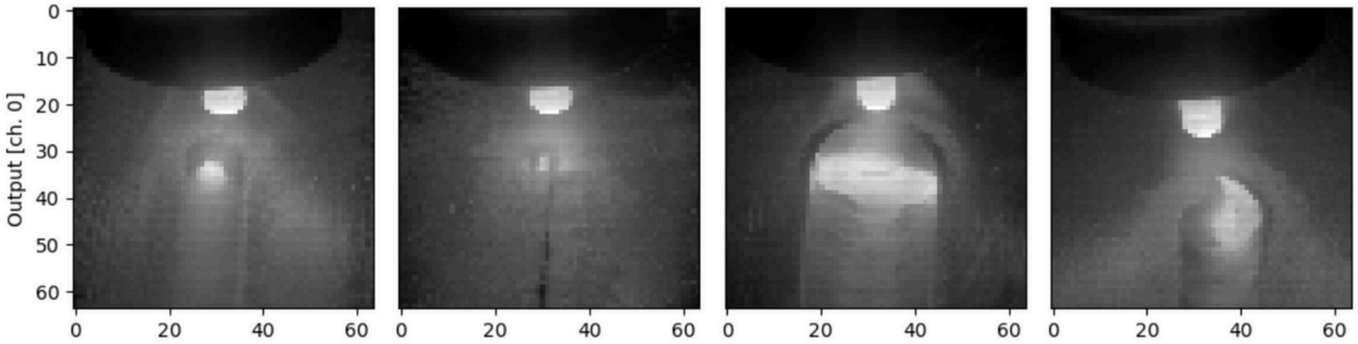


CODE IS WORKING!

TIG Aluminium 5083

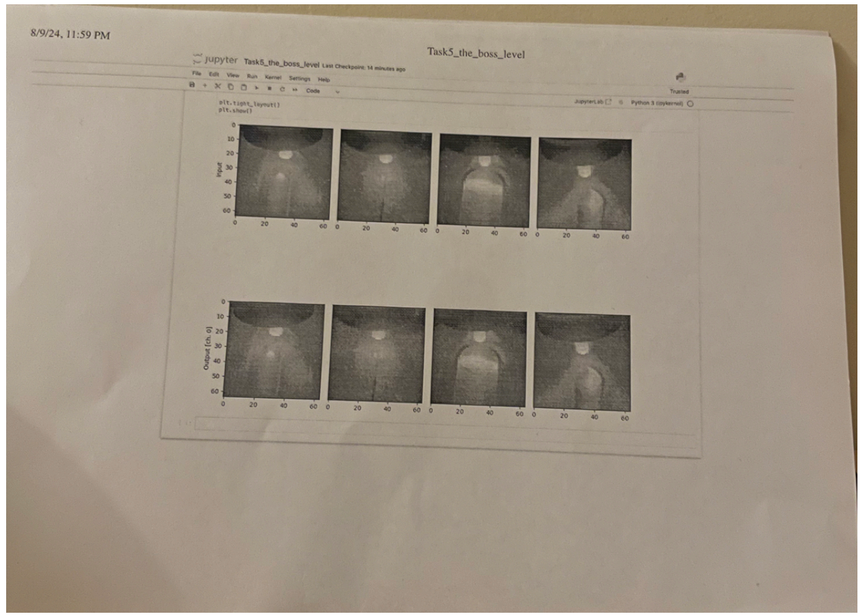
Results





‹ WOMANIUM | QUANTUM ›

TIG Aluminium 5083



“I’ll print this output and put it on my wall. I want to remind myself everyday that all my sleepless nights pay off.”

Hanna
Robotics Engineering Student

TIG Aluminium 5083 analyze

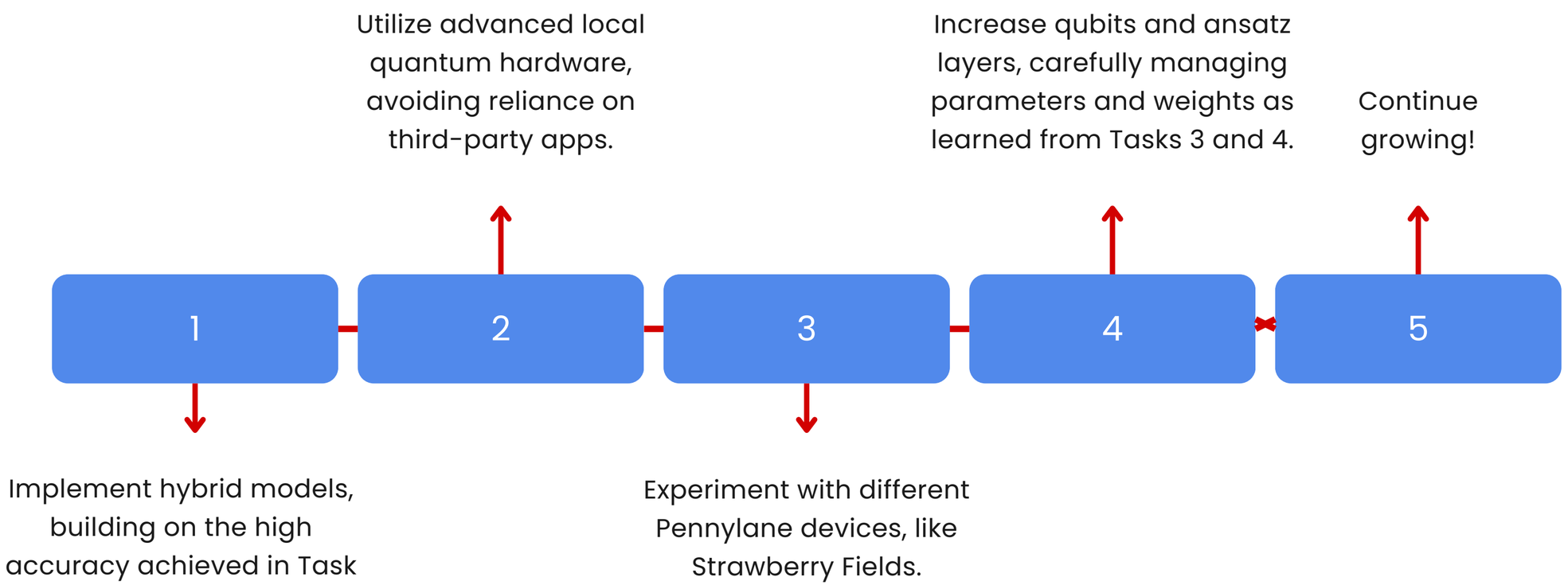
Limitations:

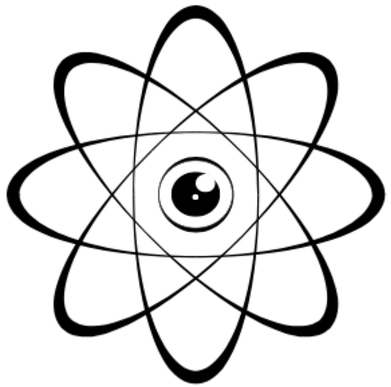
- Slow computation due to hardware, software, and internet limitations.
- Inefficient ansatz or embedding technique.
- Large dataset size causing processing delays.

Suggestions:

- Use local quantum hardware and advanced libraries.
- Explore better encoding methods or ansatzes.
- Apply data reduction techniques.
- Test different metrics and loss functions.
- Consider hybrid quantum-classical models.

Future Steps





‹ WOMANIUM | QUANTUM ›

Thank you!