# Framework for benchmarking quantum data encodings in variational algorithms

## Project Partners: Fraunhofer FOKUS Institute

Fraunhofer FOKUS is a leading research institute dedicated to advancing digital transformation across industry and public administration. Since 1988, it has combined technical, economic, and social perspectives to develop secure, interoperable, and resilient digital technologies. With expertise spanning from feasibility studies to prototype development, FOKUS supports the practical application of cutting-edge solutions. Its active role in standardization ensures that new technologies are both scientifically robust and ready for market adoption.

## Problem Description

Quantum computing has the potential to fundamentally transform machine learning, particularly through Quantum Machine Learning (QML)—an emerging field that integrates classical ML techniques with the computational advantages of quantum systems. A promising concept in QML is the use of Variational Quantum Algorithms (VQAs), which rely on a hybrid quantum-classical process where quantum circuits are optimized using classical algorithms.

One of the most critical components in these algorithms is the quantum encoding—the method used to embed classical data into quantum states. These encodings define how information is represented and processed within quantum circuits, and they have a significant impact on the performance, expressiveness, and generalization capabilities of VQAs.

Given their importance, systematically benchmarking quantum encodings is essential for understanding which encoding strategies are best suited for different types of data and tasks. However, this remains a challenging and largely open problem in the field.

This software project focuses on addressing that challenge by developing a dedicated benchmarking application for quantum encodings in VQAs. The goal is to provide a web application, which allows users to upload, evaluate, and compare different encodings

across a variety of datasets and algorithmic setups, laying the groundwork for more efficient and interpretable QML systems.

# Contact

Johannes Jung: johannes.jung@fokus.fraunhofer.de

# Approach

This project follows an iterative development process inspired by established artificial intelligence (AI) and software engineering methodologies. The aim is to develop a modular software platform for benchmarking quantum data encodings in variational quantum algorithms (VQAs). The process is divided into five main phases, which may be revisited as necessary during development.

**1. Problem Understanding and Scoping**

In the initial phase, the project goals are translated into a technical concept and system architecture. The functional requirements are defined, including the structure of the user interface, the evaluation logic, and the representation format for quantum encodings. The scope of the system is established based on these specifications.

**2. Component and System Design**

This phase focuses on the detailed design of the system's components. It includes the definition of the internal circuit format, the structure of frontend and backend modules, and the interface between them. The design emphasizes modularity, extensibility, and a clear separation of responsibilities.

**3. Implementation and Integration**

The implementation phase covers the development of the backend evaluation engine, the web-based frontend interface, and the interactive circuit editor. Once implemented, these components are integrated into a unified application. Communication between modules is handled through well-defined interfaces.

**4. Usability and Result Presentation**

After integration, the system is tested for functionality and usability. Benchmarking processes are validated, and the user interface is refined to support intuitive interaction and clear presentation of evaluation results. Visualizations are implemented to allow users to explore and compare performance metrics.

**5. Finalization and Delivery**

In the final phase, the complete system is prepared for delivery. Technical documentation, setup instructions, and user guidance are created. A final report summarizes the architecture, design decisions, and implementation details.

# Requirements

**1. Development of the Benchmarking Backend**

**1.1. Definition of Benchmarking Scope**
The system shall include a benchmarking setup composed of two to three classical datasets for quantum machine learning (suggested for binary classification), two to three variational ansätze, and two to three reference encoding strategies. All encodings and circuits will follow a shared internal format (e.g. structured gate list in JSON format). Each component must be documented with a clear rationale for its inclusion.

**Result:** A documented and implemented set of benchmark components, represented in the internal format and ready for evaluation use.

**1.2. Implementation of the Evaluation Logic**
The backend shall support parsing user-submitted encodings in the internal format, combining them with each predefined ansatz, and evaluating them across the selected datasets using a simulation framework such as PennyLane. In addition, the encoding shall be evaluated as a quantum kernel in a kernel-based classification setup.

**Result:** A backend module capable of evaluating encodings in multiple modes and returning structured performance data.

**1.3. Metric Calculation and Output Formatting**
The backend shall compute performance metrics including classification accuracy, loss,

and circuit depth. Results must be structured in a machine-readable format (e.g., JSON) and made accessible via an API or equivalent interface for frontend use.

**Result:** A backend component that calculates and formats evaluation results, ready for integration with the frontend.

## 2. Development of the Interactive Frontend Interface

### 2.1. Encoding Upload and Visualization

The frontend shall allow users to upload quantum encodings using the internal format and provide a visual representation of the circuit using a wire-and-gate diagram.

**Result:** A functional upload interface with circuit visualization that connects seamlessly to the backend evaluation process.

### 2.2. Display of Benchmark Components

The interface shall provide descriptive information and visual previews of the selected datasets, ansätze, and baseline encodings to help users understand the evaluation context.

**Result:** A well-structured overview section within the frontend, containing explanations and visual representations of all benchmark components.

### 2.3. Results Visualization and Comparison

Benchmark results shall be presented through an interactive dashboard using tables, charts, and comparison plots. The interface must allow users to assess the performance of their encodings relative to predefined baselines.

**Result:** A dynamic, user-friendly dashboard for displaying benchmarking outcomes and supporting comparative analysis.

### 2.4. Visual Quantum Circuit Builder

The system shall offer a drag-and-drop circuit editor where users can construct quantum circuits by placing gates, setting parameters, and exporting the result in the internal format.

**Result:** A visual editor integrated into the interface that produces valid, exportable circuit definitions for use in the benchmark.

### 3. System Integration, Testing, and Documentation

#### 3.1. Integration of Backend and Frontend Components

All modules must be connected into a complete, usable system. The frontend shall handle circuit uploads, initiate evaluations, and receive and display the results, all within a cohesive workflow. The system should be deployable via containerization.

**Result:** A fully integrated application with working backend–frontend interaction and deployable setup.

#### 3.2. Testing and Optimization

System functionality, data flow, and interface responsiveness must be tested to ensure correct behavior and stable operation. Identified issues should be addressed, and the system optimized where appropriate.

**Result:** A validated and stable system accompanied by a summary of testing activities and improvements made.

#### 3.3. Final Documentation and Delivery

The final delivery shall include complete technical documentation covering architecture, component roles, internal formats, deployment steps, and user instructions. A final report shall summarize the development process, benchmarking setup, and key outcomes.

**Result:** A complete documentation package including the final report, codebase, usage instructions, and optional demo material.

## Resources and Links:

**QML learning material:**

Tutorials Pennylane:

https://pennylane.ai/codebook/learning-paths

https://pennylane.ai/qml

https://pennylane.ai/qml/demos/tutorial_variational_classifier

https://medium.com/@ashrafboussahi/quantum-variational-classifier-11b74bd1bd17

https://pennylane.ai/qml/demonstrations

https://pennylane.ai/qml/glossary/quantum_embedding

Quantum Kernel Method:

https://pennylane.ai/qml/demos/tutorial_kernels_module

https://pennylane.ai/qml/demos/tutorial_kernel_based_training

Qiskit (Alternative):

https://learning.quantum.ibm.com/

Videos/Playlists:

https://www.youtube.com/watch?v=YBHzT5V1SzU&list=PL_hJxz_HrXxsQNJHWp10up8x-hwd5uwr0

https://www.youtube.com/watch?v=uCm027_jvZ0&list=PLzgi0kRtN5sO8dkomgshjSGDabnjtjBiA&index=14

Book:
https://www.academia.edu/download/99319563/Supervised_Learning_with_Quantum_Computers_Maria_Schuld_Francesco_Petruccione_z_lib.org_.pdf

**Quantum Computing general learning material (for interested students):**

Course/Videos:

https://www.youtube.com/watch?v=X2q1PuI2RFI&list=PL1826E60FD05B44E4&index=1

https://www.youtube.com/watch?v=L-vjihvQnd0&list=PLo0Vs5tDeRLRlPcJ83SN91M-asGuaa1AD

Book:

https://repositories.nust.edu.pk/xmlui/bitstream/handle/123456789/17008/992.pdf?sequence=1&isAllowed=y

**QML Datasets:**

https://pennylane.ai/datasets/collection/qml-benchmarks

**Quantum Circuit builder examples:**

https://algassert.com/quirk

https://quantum.ibm.com/composer/files/new