Auto Quantum Machine Learning for Multisource Classification

Tomasz Rybotycki[1,2,3][0000−0003−2493−0459], Sebastian Dziura[1][0009−0008−6786−511X], and Piotr Gawron[1,2][0000−0001−7476−9160]

[1] Center of Excellence in Artificial Intelligence, AGH University, al. Mickiewicza 30, 30-059 Cracow, Poland
[2] Nicolaus Copernicus Astronomical Center, Polish Academy of Sciences, ul. Bartycka 18, 00-716 Warsaw, Poland
[3] Systems Research Institute, Polish Academy of Sciences, ul. Newelska 6, 01-447 Warsaw, Poland

Abstract. With fault-tolerant quantum computing on the horizon, there is growing interest in applying quantum computational methods to data-intensive scientific fields like remote sensing. Quantum machine learning (QML) has already demonstrated potential for such demanding tasks. One area of particular focus is quantum data fusion—a complex data analysis problem that has attracted significant recent attention. In this work, we introduce an automated QML (AQML) approach for addressing data fusion challenges. We evaluate how AQML-generated quantum circuits perform compared to classical multilayer perceptrons (MLPs) and manually designed QML models when processing multisource inputs. Furthermore, we apply our method to change detection using the multispectral ONERA dataset, achieving improved accuracy over previously reported QML-based change detection results.

Keywords: quantum data fusion · auto quantum machine learning · remote sensing · multispectral imaging · ONERA

1 Introduction

Multisource Data about an observed object can be gathered from multiple complementary sensors. These data sensors have to be aggregated and transformed into useful information to e.g. obtain richer and more accurate decision outcomes. The field of multisource information fusion (MSIF) deals with this type of data processing problems.

In this work, we focus on feature-level fusion, in which the fusion model operates on extracted feature representations. MSIF has found widespread application in machine learning, particularly in classification tasks, as surveyed in [3]. Representative applications include remote sensing [19], military systems [29], and human action recognition [11], among others [20].

The issue lies at the core of MSIF, a field dedicated to efficient multisource data aggregation. It has been extensively studied, particularly in remote sensing

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29918-5_2

[6, 19]. The wide range of existing methods can be categorized into groups and subgroups [3]. In this work, we focus on one such subgroup: model-agnostic early (feature-based) fusion, where fusion occurs at early stages of the ML pipeline.

Early fusion problems are often approached with machine learning models [20, 3], which one may consider a very natural approach. After all, ML models are trained on the data. If the data is represented better (with multiple representations, coming from multiple sources), it is only natural that the results should improve. However, some of the data fusion problems have been shown to be NP-hard [28]. This means that classical techniques, while still useful, might not be able to address the problems above certain sizes (unless we see some unlikely shift in the complexity theory). This is exactly why alternative methods of computing, such as quantum computing and quantum machine learning (QML), have recently been of interest for data fusion specialists.

Quantum machine learning (QML) leverages principles of quantum mechanics to process information in ways that differ fundamentally from classical machine learning [4]. In this paradigm, computations are done within quantum circuits, commonly with the use of quantum logic gates (see e.g. [27]). In the mathematical model of quantum computation, those gates can be expressed by unitary gates, making the whole circuit (up to the measurement) a unitary transformation of the input. However simple the description may look, quantum mechanics introduces two computationally useful phenomena — superposition and entanglement — that when used correctly can significantly speed up the computations. That said, we can think of a QML model as a ML model that uses quantum computing. It is known that such models can represent complex functions using comparatively shallow (i. e. with low layers count) circuits or fewer parameters than their classical counterparts, suggesting potential advantages for learning tasks involving structured or high-dimensional data [14, 25, 1, 12]. Moreover, theoretical work has identified learning scenarios in which quantum models offer data-dependent advantages [15]. Empirical tests on the multispectral satellite data seem to also confirm this finding [12].

Recently, several works explored the application of QML methods to multisource or multi-modal data fusion. Common approach is to, instead of using purely-quantum models, take advantage of the hybrid quantum-classical models. Feature-level multi-modal quantum fusion presented in [30] is a stellar example of that approach. The ML pipeline described therein, takes features extracted by separate uni-modal classical neural networks and embeds them into a quantum fusion circuit. The features are then entangled between multiple modalities on target qubits. Their model showed consistent stability and high accuracy for high-dimensional data.

When it comes to QML models, one of the central problems is quantum architecture search (QAS) [21]. It is a quantum version of the neural architecture search (NAS) problem known in the ML community. Without the right selection of the model architecture, the training might fail. While the problem case of both NAS and QAS, there are several techniques that allow to address it. One of such techniques is automated (quantum) machine learning (AQML). AQML

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29918-5_2

are heuristic algorithms that try to propose the best models for a given data and tasks. There are several platforms for AQML (see [23] for brief overview) available for use and reports of the state-of-the-art results achieved using at least one of them [17].

It this paper, we present an application of AQML to quantum feature fusion problem. In the following Section 2 we formally present the issues addressed in this manuscript: the feature fusion and quantum architecture search. Then, we provide the detailed description of our multi-level approach in Section 3. Section 4 contains the experiments overview — along with the datasets description, hyperparameter optimization (HPO) step and the results' analysis. We conclude the paper with final remarks and insights for future work.

2 Problem formulation and literature overview

Performing the feature fusion using QML models is a complex, multi-step problem. While it most notably involves a QAS problem, there are also multiple other decisions to be made. Complete pipelines are not necessarily restricted to quantum models, after all. In this Section, we detail the feature fusion problem and further argument why QML models are plausible candidates for providing satisfactory solutions in that regard. We also introduce a specific case of data fusion — change detection task — that we considered in our experiments with the remote sensing satellite data. Finally, we formally introduce QAS problem, and discuss its usual constraints that we won't consider.

2.1 Data fusion

Data fusion has a goal-oriented definition.

Join information from multiple modalities (sources) to perform prediction [3].

In this work we study early fusion, which means the data will be aggregated directly after its extraction. Data fusion is known to enhance the quality of MLP model predictions and is required for tasks such as change detection [24]. The simplest form of data fusion is concatenating the extracted feature vectors.

2.2 Change detection

Change detection can be defined as the following task.

Given a pair of corresponding pictures, depicting the same region and taken at different times, find the regions that differ between the pictures [24].

Change detection can be considered a multisource data fusion problem, even when the picture was taken using the same device. This is due to the fact that the temporal difference between the images implies (in general) different atmospheric conditions, and notable spatial (angles, distances, ...) or geographical differences

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29918-5_2

(new buildings, ...). This, in turn, introduces heterogeneity to the pair of pictures. In the terms of data processing, change detection can be seen as a semantic segmentation task [8], or even as binary classification (change or no-change) [24]. Change detection is a well studied problem, especially in remote sensing [7].

2.3 Quantum Architecture Search

Quantum architecture search is a problem known by many names, and refers to automatic selection of quantum circuits appropriate for given task [21]. Formally speaking, one can formulate the QAS problem as follows.

Given the task and the search space, find within the search space a quantum circuits that best fits the task, according to predefined evaluation criteria.

Most often, the search space is defined by the target device, on which the circuit is meant to run. This restricts the building blocks of the circuit to a set of gates native to the device. It also limits the number of qubits one can use and defines the effective maximal depth of the circuit. In the current noisy intermediate-scale quantum era, device selection is vital to QAS. So much so that many of the proposed methods are hardware-aware by nature [21], and in the context of QML a notion of device-oriented training has been mentioned [24]. The authors of [24] also report how significant were the differences when the model trained on a noiseless simulator was run on a real quantum device.

3 Approach

In this work, we present two different approaches, sharing a common core in the form of AQML. We detailed both in Subsection 3.2. After all the models are introduced, we will briefly describe the AQML technique we used and direct interested readers to appropriate resources. In particular, we focus on the adjustments that are new in comparison to its reported version [23].

While the models we used differ slightly from each other, we would like to point out that their underlying high-level logic remains the same and it is only the implementation aspect that differs. Each of our models consists of two main parts: feature extractor and a classifier. The feature extractor is the initial part of the model we used. It accepts the input and initially processes the data. For every model, feature extractor we used is classical. Preprocessed features are then passed to the classifier. The output of this model will be used to obtain the final decision about the input, depending on the experiment. This part of the network may be classical (Subsection 3.1) or quantum and hybrid quantum-classical (Subsection 3.2).

3.1 Multi-layer perceptron

When defining the model for our initial quantum data fusion experiments, we had to make a decision regarding which part of the model we should make quantum.

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29918-5_2

Following logic similar to that in [30], we decided to use quantum layers for our classifier. Moreover, considering the effectiveness of the network in [30], we also decided to separate the feature extractor part of the model into two distinct parts — one for each input.

The first model we implemented was a classical multi-layer perceptron (MLP) [22], which was meant to serve as a baseline reference for our subsequent experiments. In this model both the extractors and the classifier were a MLP. We present this a architecture in the Figure 1.

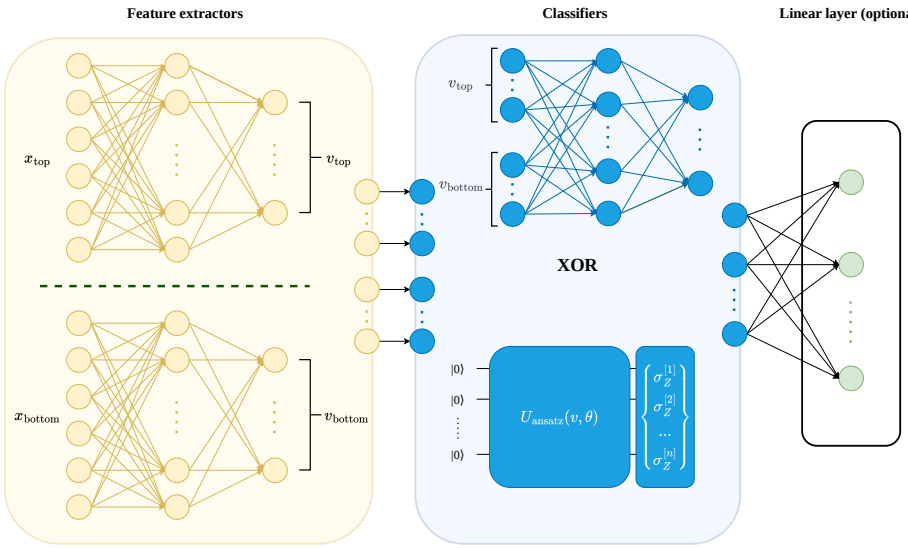

Fig. 1. A schematic representation of all the models used in this work: **yellow** — feature extractor, **blue** — classifier (here the fusion takes place), **green** — optional output logits for quantum classifiers. XOR on the schematic means that only one type of classifier is used at the time. Top classifier represents MLP approach, and the bottom one parametrized quantum circuit approach.

In this architecture, the inputs from both sources — x_{top} and x_{bottom} — are passed to their respective feature extractors and then processed, independently. After this processing, the output vectors of both extractors are concatenated, such that $v = [v^1_{\text{top}}, ..., v^n_{\text{top}}, v^1_{\text{bottom}}, ..., v^m_{\text{bottom}}]$, assuming the lengths of v_{top} and v_{bottom} are n and m respectively. The resultant vector v is used as the input of the classifier network (here another MLP). For all the layers (beside the output) we used $ReLU(v) = \max(v, 0)$ activation function, which is a standard way to introduce non-linearity in deep neural networks [22].

We used a simple MLP as a baseline to demonstrate that AQML can identify effective quantum architectures for multisource classification, rather than to outperform state-of-the-art methods. The synthetic MNIST-based dataset is a toy problem that does not require complex models. On the ONERA dataset, classi-

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29918-5_2

cal models can achieve higher accuracy [9, 24], but at the cost of substantially more trainable parameters.

3.2 Hybrid Quantum–Classical Neural Network

In our initial experiments with quantum machine learning, we used a hybrid quantum-classical machine learning model. What the name entails is the fact that in the ML pipeline, both classical and quantum layers are used. A common approach is to include a parameterized quantum circuit (PQC), with an arbitrary number of gate layers, as a single quantum layer of the network. Such layer can then be a part of a model, or a model by itself.

Starting from the model schematic, as presented in the Figure 1, we kept the feature extractors as in the MLP experiment (see Subsection 3.1). However, we used quantum classifier, instead of the classical one. This way, our final architecture resembled the one in [30]. Moreover, this way we could assess the classification capabilities on the found models, under the assumption that the data preprocessing (i.e. the preprocessing algorithm) remained intact.

In the construction of our quantum classifiers, we used two different approaches. First we tried manual selection of the ansatz (which is another usual name for a PQC). In that case, the model we selected had no data re-uploading and consisted of typically used quantum neural network (QNN) layers. In our other approach, we used AQML techniques to find the right ansatz for the problem. In this scenario, the algorithm suggested the circuit built using several QML blocks, which are pairs of data (re-)uploading and variational operations. The resultant PQC schematic is presented in the Figure 2. For some experiments, we

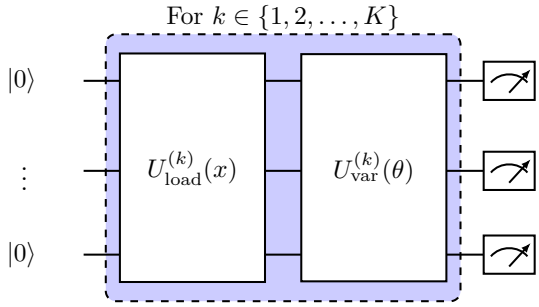

Fig. 2. A general schematic of a parameterized quantum circuit. It consists of K QML blocks, each containing a data (re-)uploading operation and a variational operation. Notice that both U_{load} and U_{var} could, in principle, be an identity. On the schematic, x denotes the input, and θ denotes the full set of PQC parameters. The PQC is concluded with a measurement.

enhanced this architecture with an additional classical layer that processed the output of the PQC and naturally led to a desired number of outputs.

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29918-5_2

The importance of data uploading layers cannot be overstated. First of all, at least one such layer is required of any QML model, as raw classical data cannot be processed by a quantum circuit. It has to be encoded into a quantum state. Second of all, data re-uploading is known — both theoretically and empirically — to increase the expressiveness of quantum circuits [26].

Change detection hybrid model To investigate the performance of our approach with the real data, we decided to revisit the experiments reported in [24], only this time, with the use of AQML. The model we used for those experiments follows the schematic presented in the Figure 1 and is similar to the model described in previous Subsection 3.2. The only difference here is that we used a single feature extractor instead of two, to which we uploaded the concatenated data from both sources. We decided not to separate the feature extraction, so that our experiment differs from [24] only in the application of AQML. In this model, the final classical layer was also an optional addition. To reduce the initial dimensionality of the input (with 13-band pixels it would require too many qubits for efficient simulation) PCA was used, leaving the input from each source described by four features.

3.3 Automated Quantum Machine Learning

In our experiments, we used the `aqmlator` library and thus the hyperparameter optimization (HPO)-based AQML approach reported in [23]. The idea behind this algorithm is to treat the ansatz architecture as an hyperparameter and apply classical HPO techniques to find it. However, instead of using quantum gates as the basic building blocks of the PQC, the authors proposed to use well-known and commonly used quantum layers, and thus limiting the search-space of the optimization algorithm significantly.

The latest version of `aqmlator` (2.0.0a1, available only through the sources), aside from technical improvements, offers more flexible ansatze finding classes, proposes data re-uploading layers, and uses a wider range of variational layers during the ansatz search. In particular, it implements the `SimplifiedTwoDesign` and the `BellmanLayer`, thus making it possible for `aqmlator` to find the models used in the previous ONERA QML change detection experiments [24].

4 Experiments

In the following sections, we present the overview of the quantum data fusion experiments we've done during this study. We begin with the description of our initial experiments using synthetic data created from the well-known MNIST dataset [18]. With the successful application of QML and AQML to the synthetic data, we shift our attention to the real-world ONERA dataset [5] and revisit experiments from [24], but this time using AQML techniques to select the quantum classifier. The code and the data from our MNIST experiments

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29918-5_2

are available through Zenodo [10]. Our ONERA AQML experiments are in the original repository of [24].

All experiments employed noiseless simulators. Prior work [24] studied the impact of noise and real-device execution for PQC-based change detection on ONERA, suggesting that performance would degrade similarly in our setting.

4.1 MNIST

Dataset The original MNIST dataset consists of 70,000 grayscale images of handwritten digits (0–9), each of size 28×28 pixels. We obtained the dataset from Kaggle [18], and used the predefined training–validation split as-is.

In the preliminary research, we reduced the number of output classes from 10 to 3. The idea here was to reduce the complexity of the task, so that we could ensure reasonable (i. e. simulable) sizes of our quantum classifiers. We also wanted not to overcomplicate our preliminary examples. We arbitrarily selected:

- digit "5" (5421 training samples and 892 validation samples),
- digit "6" (5918 training samples and 958 validation samples),
- digit "7" (6265 training samples and 1028 validation samples).

Images were split into top/bottom halves, binarized, downsampled, and converted to 14-dimensional vectors by computing the mean pixel intensity along the columns. These vectors correspond to the synthetic multisource inputs x_{top} and x_{bottom}, illustrated in Figure 3.

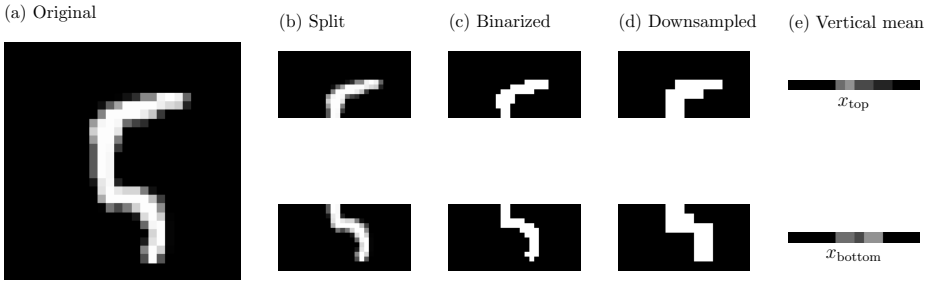

Fig. 3. Overview of the synthetic multisource preprocessing pipeline. (a) Original input image. (b) Horizontal splitting into top and bottom halves. (c) Binarization of each half. (d) Spatial downsampling. (e) Computation of column-wise mean pixel intensities, resulting in vector representations that constitute the synthetic multisource inputs x_{top} and x_{bottom}.

Experimental setup The x_{top} and x_{bottom} are used directly as the input to the respective feature extractors. Each feature extractor had, therefore, 14 input neurons. The number of neurons in the hidden and output layers of the feature

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29918-5_2

extractors were determined by the HPO experiments detailed below. In those experiments, feature extractors had only one hidden layer.

Similarly for the number of a single hidden layer neurons in the classical classifier and number of QML blocks in the quantum classifier. Classifiers' number of inputs was determined by the length of concatenated output of the feature extractors v, and output size was determined by the number of classes, which was fixed to three. The reason for the latter was the use of standard one-hot encoding [13], common in classification tasks.

The training goes as follows. The vector inputs is subsequently passed through learnable feature extraction functions $f(\cdot, \theta_1)$ and $f(\cdot, \theta_2)$, producing feature vectors v_{top} and v_{bottom} of equal dimensionality. These features are fused and classified by a function $h(\cdot, \theta_3)$ — realized by the classifier — yielding the final classification decision d. The resulting formulation is defined as:

$$v_{\text{top}} = f(x_{\text{top}}, \theta_1); \quad v_{\text{bottom}} = f(x_{\text{bottom}}, \theta_2); \quad d = h(v_{\text{top}}, v_{\text{bottom}}, \theta_3).$$

All learnable parameters θ_1, θ_2, and θ_3 are optimized jointly during the training. The training is performed using the cross-entropy loss, and predictions are obtained by selecting the output logit (or qubit) with the maximum output value. In the case of PQC, we used measured expectation value of the Pauli Z operator for each qubit. We trained all the models using 5-fold cross-validation, to better assess the average performance of the investigated models. The optimizer we used was standard ADAM optimizer [16].

Metrics Although we gathered information about accuracy, precision, recall and f1-scores of the model, in their different variants implemented in `scikit-learn` version 1.8, the scores we ultimately used for models evaluation were accuracy and averaged f1_{macro} score. Both are standard metrics used for measuring classification performance, and f1-score aggregates the information given by recall and precision (by definition). This set of metrics allowed us to assess the general performance of the evaluated models.

Hyperparameter Optimization We performed hyperparameter optimization (HPO) to select the optimal architecture for the models we tested in the experiment. To that aim, we used `optuna` hyperparameter optimizer [2], a `Python` package for HPO, used in both ML and QML. We present the HPO setup and the results for the classical and quantum experiments in the Tables 1 and 2. Those results were obtained without early stopping, and using random seed equal to 42. The number of trials varied, depending on the model, and was equal to 100, 100, 50 and 35 for MLP, PQC_{manual}, $PQC_{\text{AQML}^{\text{Solo}}}$ and $PQC_{\text{AQML}^{\text{Linear}}}$, respectively. The numbers differ due to limited compuational resources.

Results In the Table 3 we present the values of metrics obtained with the best classical, manual PQC and AQML-found PQC models. The results clearly show that the models found with AQML outperformed manually selected PQCs for

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29918-5_2

Table 1. Experiment setup for baseline hyperparameters optimization.

Parameter	Range	Type	Best value
Learning rate	$[10^{-3}, 10^{-2}]$	linear	10^{-3}
Batch size	$[32, 128]$	discrete	83
Hidden layer size (feature extractors)	$[64, 256]$	discrete	90
Output layer size (feature extractors)	$[64, 256]$	discrete	196
Hidden layer size (classifier)	$[64, 256]$	discrete	95

our synthetic multisource MNIST classification task. The accuracy they obtain is on par with the classical MLP, but with over 10 times less parameters. Over 100 times less, if we consider only classifiers.

In the Table 3 one can also see that while PQCs with additional linear layer performed better, the difference between the best models reported isn't significant (is within a single standard deviation). We have noticed, however, that the models with linear layer seem to behave more consistently during the training. We started to investigate that phenomenon, and found that indeed accuracy averaged over all models from given groups was:

$$\text{accuracy}^{\text{avg}}_{\text{PQC}^{\text{Solo}}_{AQML}} = 0.837 < 0.889 = \text{accuracy}^{\text{avg}}_{\text{PQC}^{\text{Linear}}_{AQML}}, \tag{1}$$

which seems to confirm our initial insights. It's also consistent with another empirical observation:

$$\text{accuracy}^{\text{min}}_{\text{PQC}^{\text{Solo}}_{AQML}} = 0.367 < 0.631 = \text{accuracy}^{\text{min}}_{\text{PQC}^{\text{Linear}}_{AQML}}. \tag{2}$$

This fact was also observed for the other metrics. We plan to continue to investigate this issue further as a part of a separate study.

4.2 ONERA

Dataset The ONERA change detection dataset is a publicly available dataset containing annotated pairs of multispectral satellite images [5]. The spatial resolution of the images vary between the pairs. Each image has 13 spectral bands. An example of the input, reduced to the visible bands, is presented in the Figure 4.

Experimental setup In our experiments we repeat the experiment reported in [24], thus the setup remains the same. The only change we include is using AQML to find ansatze used for the change detection. The metric we consider is still accuracy and the data is preprocessed and fused in the same manner as in [24], as we explained in the Section 3.

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29918-5_2

Table 2. Experiment setup for PQC hyperparameters optimization. For those experiments the learning rate was fixed to $lr = 10^{-3}$. $\text{PQC}^{\text{Solo}}_{\text{AQML}}$ and $\text{PQC}^{\text{Linear}}_{\text{AQML}}$ denote PQC models without and with final classical linear layer respectively. We present the architecture of the model in the parenthesis, next to the model's name.

Parameter	**Range**	**Type**	**Best value**
$\text{PQC}_{\text{Manual}}$ (`AngleX` $\rightarrow 3\times$ `BEL`)			
Batch size	$[32, 128]$	discrete	32
Hidden layer size (feature extractors)	$[64, 256]$	discrete	128
Output layer size (feature extractors)	$[4, 12]$	discrete	6
`BasicEntangler` layers number (classifier)	$[1, 5]$	discrete	3
$\text{PQC}^{\text{Solo}}_{\text{AQML}}$ (`Amplitude` $\rightarrow$ `SEL` $\rightarrow$ `AngleY` $\rightarrow 2\times$ `SEL`)			
Batch size	$[32, 128]$	discrete	76
Hidden layer size (feature extractors)	$[64, 256]$	discrete	163
Output layer size (feature extractors)	$[4, 12]$	discrete	8
QML Blocks number (classifier)	$[1, 5]$	discrete	3
$\text{PQC}^{\text{Linear}}_{\text{AQML}}$ (`Amplitude` $\rightarrow$ `BEL` $\rightarrow$ `AngleX` $\rightarrow$ `BEL` $\rightarrow$ `AngleZ` $\rightarrow$ `BEL` $\rightarrow$ `SEL` $\rightarrow$ `BEL`)			
Batch size	$[32, 128]$	discrete	121
Hidden layer size (feature extractors)	$[64, 256]$	discrete	97
Output layer size (feature extractors)	$[4, 12]$	discrete	8
QML Blocks number (classifier)	$[1, 5]$	discrete	5

For the AQML, we set up the number of QML blocks in range [1, 5]. We also included baseline classical classifier, which was an MLP with 6 layers. As in [24], we didn't vary over the number of qubits, and it remained equal to 8 for all the selected models.

Results We ran 26, 36 and 41 experiments for MLP, $\text{PQC}^{\text{Solo}}_{\text{AQML}}$ and $\text{PQC}^{\text{Linear}}_{\text{AQML}}$ models respectively. It's worth mentioning that in change detection experiments the architecture of MLP didn't change from run to run. It is therefore justified to consider both the base and the average performance of MLP in the comparison with PQC models.

While the accuracy of the best classical model outperformed the best PQC model — 0.752 vs. 0.743 — the best PQC model was better than average MLP. The average performance comparison is even better, because the average accuracy of $\text{PQC}^{\text{Linear}}_{\text{AQML}}$ models outperformed MLP, although only slightly — 0.730 vs. 0.728. Even more surprising is the fact that the best PQC model found was a very basic model: `Amplitude` $\rightarrow$ `BEL`. This model had only 8 trainable parameters! For the comparison, the MLP we used had 384 trainable parameters. While

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29918-5_2

Table 3. Average accuracy and $f1_{macro}$-score metrics for classical, manual PQC and AQML-found PQC models. PQC^{Solo}_{AQML} and PQC^{Linear}_{AQML} denote PQC models without and with final classical linear layer respectively. We also provide trainable parameters number for each model.

Metric	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Mean
MLP (# Parameters. Feature extractors: 38372; Classifier: 37623, Total: 75995)						
accuracy	0.962	0.959	0.960	0.960	0.960	0.960 ± 0.001
$f1_{macro}$	0.961	0.959	0.959	0.959	0.959	0.959 ± 0.001
PQC_{manual} (# Parameters. Feature extractors: 4614; Classifier: 18, Total: 4632)						
accuracy	0.880	0.855	0.880	0.905	0.911	0.886 ± 0.022
$f1_{macro}$	0.879	0.852	0.879	0.903	0.910	0.885 ± 0.023
PQC^{Solo}_{AQML} (# Parameters. Feature extractors: 6202; Classifier: 72, Total: 6274)						
accuracy	0.937	0.943	0.935	0.934	0.949	0.939 ± 0.006
$f1_{macro}$	0.935	0.941	0.933	0.932	0.947	0.938 ± 0.006
PQC^{Linear}_{AQML} (# Parameters. Feature extractors: 10764; Classifier: 135, Total: 10899)						
accuracy	0.947	0.947	0.935	0.943	0.947	0.944 ± 0.005
$f1_{macro}$	0.946	0.946	0.933	0.942	0.946	0.943 ± 0.005

none of the models is close to the reported state-of-the-art, we conclude that given similar restrictions, AQML-found PQCs lead to results comparable with classical models.

To further justify the comparison PQC and classical models, we rerun 13 repetitions of the experiment with MLP containing a single hidden layer, reducing the number of trainable parameters to 64. The accuracy of MLP didn't drop significantly, and remained at an impressive 0.731 on average (the best from all the models). However, the best accuracy it achieved was 0.738, which is below the best accuracy reported by baseline MLP and PQC. Finishing this direction of research, we considered an MLP with only 2 hidden neurons, and thus 16 parameters. Over another 10 runs, the model scored 0.731, 0.667, 0.525 for maximal, average and minimal accuracy, all below AQML-found PQCs.

We also report that AQML approach led to the results that were better than originally reported (with accuracy 0.752 vs. 0.720). That's also true for PQC^{Solo}_{AQML} models, which achieved best accuracy of 0.738. Furthermore, the best PQC model had only 8 parameters and significantly less gates than the model reported in the original research. This conclusion is vital, because transpiling and running such model on a real device would be much easier and would lead to lesser influence of the device imperfections. Models that simple were not even considered in the original study! This highlights how useful the AQML approach can be, at least during the prototyping stage of the research.

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29918-5_2

Fig. 4. A pair of images from the ONERA dataset. They show Saclay, a city in France. The images were taken on 15 III 2016 (Left) and 29 VIII 2017 (Right).

Finally, we also report that while the additional classical linear layer had no significant influence on the best accuracy obtained with both the models, we noticed that the stability of the training and the results are greatly increased if we use the linear layer (and thus consider all the output qubits). The results are even more visible than in the case of the MNIST dataset, with:

$$\text{accuracy}^{\text{avg}}_{\text{PQC}^{\text{Solo}}_{AQML}} = 0.676 < 0.738 = \text{accuracy}^{\text{avg}}_{\text{PQC}^{\text{Linear}}_{AQML}}, \tag{3}$$

$$\text{accuracy}^{\text{min}}_{\text{PQC}^{\text{Solo}}_{AQML}} = 0.505 < 0.710 = \text{accuracy}^{\text{min}}_{\text{PQC}^{\text{Linear}}_{AQML}}. \tag{4}$$

This observation positions training stability research as an interesting direction for future investigations.

5 Conclusion

In this study, we show that even a simple AQML approach can improve hybrid quantum-classical models, particularly for data fusion tasks.

Using specially prepared MNIST as an synthetic example, we report that manually selected QML models were worse than that found by AQML. Although the models were never as good as the best classical models, they were comparable, and they needed far less trainable parameters to achieve their accuracy.

We also evaluated real data, applying AQML to change detection on the ONERA dataset. We find not only improvements over prior QML results, but also that AQML-generated models outperform classical MLPs with similar parameter counts on average, further demonstrating the effectiveness of AQML.

An interesting by-product of our study is that the additional classical linear output layer seems to stabilize the hybrid model training. We presented the preliminary results on that matter and leave the proper investigation for further studies.

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29918-5_2

Acknowledgments. We gratefully acknowledge the funding support by the program "Excellence initiative—research university" for the AGH University of Kraków as well as the ARTIQ project (UMO-2021/01/2/ST6/00004 and ARTIQ/0004/2021).

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Abbas, A., Sutter, D., Zoufal, C., Lucchi, A., Figalli, A., Woerner, S.: The power of quantum neural networks. Nature Computational Science **1**(6), 403–409 (Jun 2021). https://doi.org/10.1038/s43588-021-00084-1
2. Akiba, T., Sano, S., Yanase, T., Ohta, T., Koyama, M.: Optuna: A next-generation hyperparameter optimization framework (2019), https://arxiv.org/abs/1907.10902
3. Baltrušaitis, T., Ahuja, C., Morency, L.P.: Multimodal Machine Learning: A Survey and Taxonomy. IEEE Transactions on Pattern Analysis and Machine Intelligence **41**(2), 423–443 (Feb 2019). https://doi.org/10.1109/TPAMI.2018.2798607
4. Biamonte, J., Wittek, P., Pancotti, N., Rebentrost, P., Wiebe, N., Lloyd, S.: Quantum machine learning. Nature **549**(7671), 195–202 (Sep 2017). https://doi.org/10.1038/nature23474
5. Caye Daudt, R., Le Saux, B., Boulch, A.: Fully convolutional siamese networks for change detection. In: 2018 25th IEEE International Conference on Image Processing (ICIP). pp. 4063–4067 (2018). https://doi.org/10.1109/ICIP.2018.8451652
6. Chair, Z., Varshney, P.: Optimal data fusion in multiple sensor detection systems. IEEE Transactions on Aerospace and Electronic Systems **AES-22**(1), 98–101 (1986). https://doi.org/10.1109/TAES.1986.310699
7. Cheng, G., Huang, Y., Li, X., Lyu, S., Xu, Z., Zhao, Q., Xiang, S.: Change detection methods for remote sensing in the last decade: A comprehensive review (2023), https://arxiv.org/abs/2305.05813
8. Csurka, G., Volpi, R., Chidlovskii, B.: Semantic image segmentation: Two decades of research (2023), https://arxiv.org/abs/2302.06378
9. Daudt, R.C., Le Saux, B., Boulch, A., Gousseau, Y.: Urban Change Detection for Multispectral Earth Observation Using Convolutional Neural Networks. In: IGARSS 2018 - 2018 IEEE International Geoscience and Remote Sensing Symposium. pp. 2115–2118 (Jul 2018). https://doi.org/10.1109/IGARSS.2018.8518015
10. Dziura, S., Rybotycki, T., Gawron, P.: Automated quantum machine learning for multisource classification: Experimental data and the code (Feb 2026). https://doi.org/10.5281/zenodo.18717347, https://doi.org/10.5281/zenodo.18717347
11. Fortino, G., Ghasemzadeh, H., Gravina, R., Liu, P.X., Poon, C.C.Y., Wang, Z.: Advances in multi-sensor fusion for body sensor networks: Algorithms, architectures, and applications. Information Fusion **45**, 150–152 (Jan 2019). https://doi.org/10.1016/j.inffus.2018.01.012
12. Gupta, M.K., Romaszewski, M., Gawron, P.: Potential of quantum machine learning for processing multispectral earth observation data. Bulletin of the Polish Academy of Sciences Technical Sciences **73**(5), e154279 (2025). https://doi.org/10.24425/bpasts.2025.154279
13. Hancock, J.T., Khoshgoftaar, T.M.: Survey on categorical data for neural networks. Journal of Big Data **7**(1), 28 (2020). https://doi.org/10.1186/s40537-020-00305-w

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29918-5_2

14. Havlíček, V., Córcoles, A.D., Temme, K., Harrow, A.W., Kandala, A., Chow, J.M., Gambetta, J.M.: Supervised learning with quantum-enhanced feature spaces. Nature **567**(7747), 209–212 (Mar 2019). https://doi.org/10.1038/s41586-019-0980-2
15. Huang, H.Y., Broughton, M., Mohseni, M., Babbush, R., Boixo, S., Neven, H., McClean, J.R.: Power of data in quantum machine learning. Nature Communications **12**(1), 2631 (May 2021). https://doi.org/10.1038/s41467-021-22539-9
16. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization (2017), https://arxiv.org/abs/1412.6980
17. Koike-Akino, T., Wang, P., Wang, Y.: Autoqml: Automated quantum machine learning for wi-fi integrated sensing and communications (2022), https://arxiv.org/abs/2205.09115
18. LeCun, Y.: MNIST Handwritten Digits (Dec 2025), https://www.kaggle.com/datasets/hichamachahboun/mnist-handwritten-digits
19. Li, H.C., Hu, W.S., Li, W., Li, J., Du, Q., Plaza, A.: A3CLNN: Spatial, Spectral and Multiscale Attention ConvLSTM Neural Network for Multisource Remote Sensing Data Classification. IEEE Transactions on Neural Networks and Learning Systems **33**(2), 747–761 (Feb 2022). https://doi.org/10.1109/TNNLS.2020.3028945
20. Li, X., Dunkin, F., Dezert, J.: Multi-source information fusion: Progress and future. Chinese Journal of Aeronautics **37**(7), 24–58 (Jul 2024). https://doi.org/10.1016/j.cja.2023.12.009
21. Martyniuk, D., Jung, J., Paschke, A.: Quantum architecture search: A survey. In: 2024 IEEE International Conference on Quantum Computing and Engineering (QCE). p. 1695–1706. IEEE (Sep 2024). https://doi.org/10.1109/qce60285.2024.00198
22. Russell, S., Norvig, P.: Artificial Intelligence: A Modern Approach. Prentice Hall, 3rd edn. (2010)
23. Rybotycki, T., Gawron, P.: AQMLator – an Quantum Machine Learning e-Platform. Computer Science **26**(SI) (Jul 2025). https://doi.org/10.7494/csci.2025.26.si.7063
24. Rybotycki, T., Gupta, M.K., Gawron, P.: Explainable quantum machine learning for multispectral images segmentation: Case study (2025), https://arxiv.org/abs/2503.08962
25. Schuld, M., Killoran, N.: Quantum machine learning in feature Hilbert spaces. Physical Review Letters **122**(4), 040504 (Feb 2019). https://doi.org/10.1103/PhysRevLett.122.040504
26. Schuld, M., Sweke, R., Meyer, J.J.: The effect of data encoding on the expressive power of variational quantum machine learning models (Mar 2021). https://doi.org/10.1103/PhysRevA.103.032430, arXiv:2008.08605 [quant-ph]
27. Sebastianelli, A., Zaidenberg, D.A., Spiller, D., Saux, B.L., Ullo, S.L.: On circuit-based hybrid quantum neural networks for remote sensing imagery classification (2021), https://arxiv.org/abs/2109.09484
28. Stooβ, V., Ulmke, M., Govaers, F.: Quantum computing for applications in data fusion. IEEE Transactions on Aerospace and Electronic Systems **59**(2), 2002–2012 (2023). https://doi.org/10.1109/TAES.2022.3212026
29. Sycara, K., Glinton, R., Yu, B., Giampapa, J., Owens, S., Lewis, M., Grindle, L.C.: An integrated approach to high-level information fusion. Information Fusion **10**(1), 25–50 (Jan 2009). https://doi.org/10.1016/j.inffus.2007.04.001
30. Wu, Y., Zhou, Q., Geng, J., Deng, X., Jiang, W.: Feature Entanglement-based Quantum Multimodal Fusion Neural Network (Jan 2026). https://doi.org/10.48550/arXiv.2601.07856

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29918-5_2