

# 空气质量四分类任务：经典与量子机器学习模型设计与性能分析

跃迁元<sup>1</sup>

王俊亚，陈兴平，韩睿劼，陈华林

<sup>1</sup> 中山大学，广东省广州市 510006；

**摘要：**空气污染已成为全球性环境挑战，准确评估与预测空气质量对于城市可持续发展具有重要意义。本文围绕某空气质量预测竞赛任务，构建并比较了经典神经网络模型与量子机器学习模型的性能表现。我们首先实现了结构紧凑的多层感知机（MLP）与小型残差网络（ResNet），作为经典基线模型。在此基础上，提出了一种量子经典混合的多层感知机（QMLP）模型，采用参数化变量子电路对输入特征进行建模。实验结果表明，在参数量更少的情况下，量子混合模型具备良好的预测潜力，并展现出较高的模型压缩效率与泛化能力。研究结果为量子机器学习在环境数据分析中的应用提供了实践探索与技术参考。

**关键词：**机器学习；量子机器学习；量子经典混合；

## 1 概述

空气污染是全球性环境问题，对人类健康和生态系统产生了深远的影响。准确评估空气质量并预测污染等级对于制定环境保护政策和改善居民生活质量至关重要。本次比赛旨在利用量子机器学习技术，基于多维度的环境数据，预测空气质量等级，为城市环境管理提供决策支持。

我们在本地运行时的代码结构如下（代码仓库链接<https://github.com/YuZhuZhi/Problem2>）：

- **Data** 文件夹：存放数据集 `train_data.csv` 和 `test_data.csv`。
- **Utils.py**：实现数据读取、预处理等的工具类和函数。
- **ClassicalModel.py**：实现经典机器学习模型的网络框架。
- **QuantumModel.py**：实现量子机器学习模型的网络框架。
- **Trainer.py**：实现模型训练和评估。
- **Main.py**：主程序，负责调用各个模块，进行数据加载、模型训练和评估。

## 2 数据探索与预处理

本任务中提供了 `train_data.csv` 与 `test_data.csv` 两个数据集。两个文件具有相同的结构，从左到右依次为：温度（ $^{\circ}C$ ）、湿度（%）、 $PM_{2.5}$  浓度（ $\mu g/m^3$ ）、 $PM_{10}$  浓度（ $\mu g/m^3$ ）、 $NO_2$  浓度（ppb）、 $SO_2$  浓度（ppb）、CO 浓度（ppm）、到最近工业区的距离（km）、人口密度（人/ $km^2$ ）、空气质量。其中，前九列是特征，最后一列即空气质量是标签，共有四类：Good、Moderate、Poor 和 Hazardous，在之后我们会将其映射为 0 ~ 3。

为了读取文件，我们在 `Utils.py` 文件中实现了 `CSVReader` 类。该类的构造函数读入一个数据集文件路径，自动加载文件中所有数据，包括列名、数据维度、缺失值等。通过调用函数，也可以查看读入数据的统计信息，例如均值和方差等。这个类中有以下方法：

- `__init__(self, filePath: str)`：构造函数，传入数据集文件路径，自动读取数据集。
- `__loadData__`(self)：读取数据集文件，没有返回值。会被构造函数自动调用，用户不得手动调用。
- `getData(self)`：以二元组返回数据集的特征和标签。

- `checkBasicInfo(self)`: 输出检查数据集的基本信息, 包括维度、缺失值等。
- `showStatistics(self)`: 输出数据集的统计信息, 包括均值、方差等。
- `read(filePath: str)`: 静态方法, 传入数据集文件路径, 返回数据集特征和标签。用于快速读取数据集。

一般而言, 为了模型能够更好地学习特征, 我们会对数据进行预处理。由于数据量较小, 简单的预处理即可完成需求。我们在 `Utils.py` 文件中实现了 `Math` 静态类。这个类中只有一个静态函数:

- `normalize(data: np.ndarray)`: 对数据进行归一化处理, 返回归一化后的数据。

这是为了保证不会修改 `CSVReader` 类中的数据。在 `Trainer.py` 文件中, `Trainer` 类读入数据后会自动调用 `Math.normalize` 函数对数据进行归一化处理。

### 3 经典神经网络模型

#### 3.1 算法原理

我们实现了两种经典神经网络结构作为基准模型: 残差网络和多层感知器。

##### 3.1.1 SmallMLP (小型多层感知机)

多层感知机是一种简单的神经网络结构, 由多个全连接层组成, 通常用于分类和回归任务。它的核心思想是通过逐层的线性变换和非线性激活函数来学习输入数据的复杂映射关系。

- 多层感知机 (MLP):

$$\begin{aligned}x_1 &= \text{ReLU}(\text{Linear}_1(x)) \\x_2 &= \text{ReLU}(\text{Linear}_2(x_1)) \\x_3 &= \text{Linear}_{\text{output}}(x_2) \\ \text{output} &= \text{Softmax}(x_3)\end{aligned}$$

##### 3.1.2 SmallResNet (小型残差网络)

残差网络<sup>[2]</sup>是一种深度学习架构, 主要用于解决深层神经网络中的梯度消失和梯度爆炸问题。它的核心思想是引入“残差学习”机制, 通过残差连接 (Skip Connection) 让网络中的信号可以绕过某些层直接传播。

- 残差块 (Residual Block):

$$\text{output} = \text{ReLU}(\text{Linear}_2(\text{ReLU}(\text{Linear}_1(x)))) + x$$

- 主网络:

$$\begin{aligned}x_1 &= \text{ReLU}(\text{Linear}_{\text{input}}(x)) \\x_2 &= \text{BasicBlock}(x_1) \\x_3 &= \text{Linear}_{\text{output}}(x_2) \\ \text{output} &= \text{Softmax}(x_3)\end{aligned}$$

#### 3.2 实现细节

我们实现的一个 MLP 网络结构如图3.1所示, 由于其规模较小, 称为 `smallMLP`。该网络包含一个输入层、两个隐藏层和一个输出层。输入层有 9 个节点, 分别对应 9 个特征; 第一个隐藏层有 8 个节点, 第二个隐藏层有 6 个节点; 输出层有 4 个节点, 分别对应四类标签。每一层的神经元之间是全连接的。在这个网络中, 共计使用 **162** 个参数。

另一方面, 我们还实现了一个小型的残差网络 (`smallResNet`) 如图3.2所示, 以期能取得更好的效果。该网络大体结构与 `smallMLP` 相同, 但对于残差连接结构, 使用了  $8 \rightarrow 4 \rightarrow 8$  的结构。`Linear1` 层的输出结果, 会直接与 `Res1A` 层的输出结果相加, 作为 `Res1B` 层的输入, 即图3.2中的红色虚线所示。在这个网络中, 共计使用 **192** 个参数。

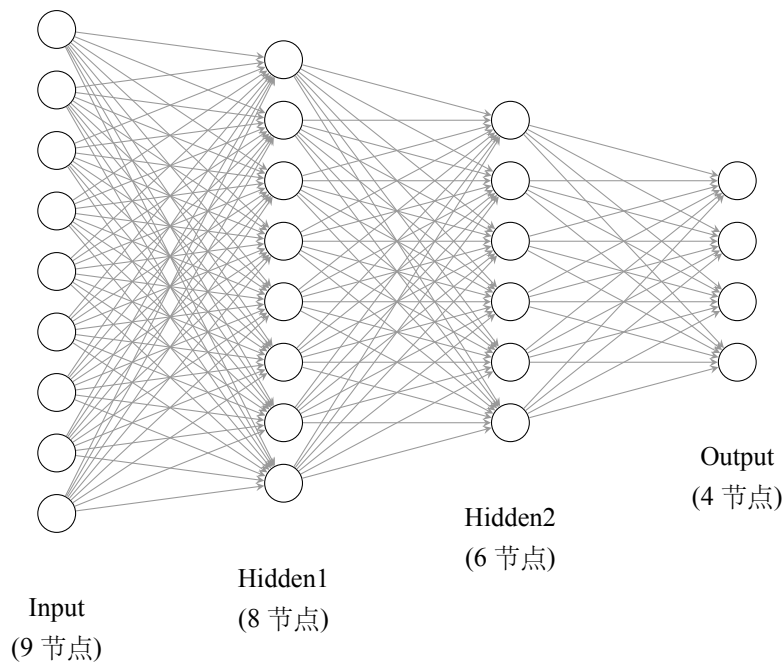


图 3.1 smallMLP 网络结构

3.3 性能评估

使用多层感知机与残差网络对数据集进行训练，得到如表3.1所示的实验结果。

表 3.1 经典模型的训练结果

| 迭代次数 | 经典模型        | 训练准确率  | 测试准确率  | 测试 F1 分数 |
|------|-------------|--------|--------|----------|
| 50   | SmallMLP    | 69.55% | 66.10% | 0.3993   |
|      | SmallResNet | 74.37% | 75.30% | 0.5564   |
| 250  | SmallMLP    | 86.98% | 85.60% | 0.6645   |
|      | SmallResNet | 87.02% | 85.50% | 0.6632   |
| 500  | SmallMLP    | 87.15% | 85.50% | 0.6635   |
|      | SmallResNet | 87.37% | 85.80% | 0.6662   |

根据实验结果可知，在固定参数大小下，经典的神经网络模型 ResNet 与 MLP 的实验精度基本一样，使用更复杂的 ResNet 模型对实验的结果精度并未产生较大提升。从模型理论上讲，由于更复杂的模型 (ResNet) 在提出的时候往往是解决增加模型参数却不能极大提高准确率的问题，而在该任务中所用经典参数较少，并未达到 MLP 的模型性能极限，因此使用 ResNet 并未提升较大精度。

由以上实验分析，我们可以提出如下假设：在固定参数下，存在经典模型并未超出其性能极限，即在固定参数下更改模型并不能提升预测精度。在此假设下，我们希望存在某种模型能够在原有经典模型的基础上，能进一步较大提升模型的预测精度，从而减少模型的参数量，进而提高模型的计算效率。

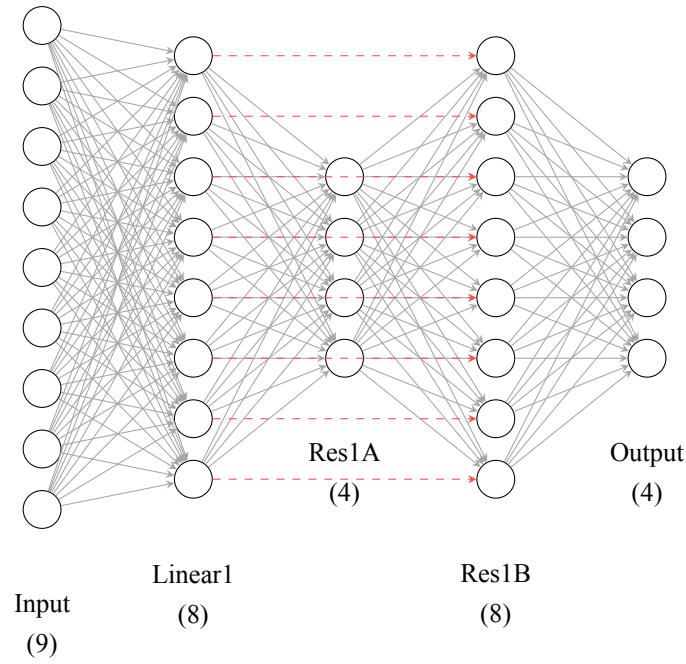


图 3.2 SmallResNet 网络结构（含残差连接）

## 4 量子机器学习模型

### 4.1 算法原理

#### Algorithm 1 QMLP 训练流程

- 1: 初始化参数  $\{\theta_l\}_{l=1}^L$  和  $\{f_l\}_{l=1}^L$
- 2: **for** 每个训练 epoch **do**
- 3:   编码输入:  $|\phi_0\rangle = S(\mathbf{x})|0\rangle^{\otimes N}$
- 4:   量子态演化:  $|\phi_L\rangle = \prod_{l=1}^L U_l(\theta_l)|\phi_0\rangle$
- 5:   参数化纠缠:  $|\phi_L\rangle = \bigotimes_{m=1}^{N/2} CRX(\phi_m)|\phi_L\rangle$
- 6:   重传:  $|\phi_L\rangle = R_l(\mathbf{x})|\phi_L\rangle$
- 7:   测量期望值:  $\langle Z \rangle_i = \langle \phi_L | Z_i | \phi_L \rangle$
- 8:   计算损失:  $\mathcal{L} = -\sum_c y_c \log p_c$
- 9:   通过参数偏移规则更新参数
- 10: **end for**

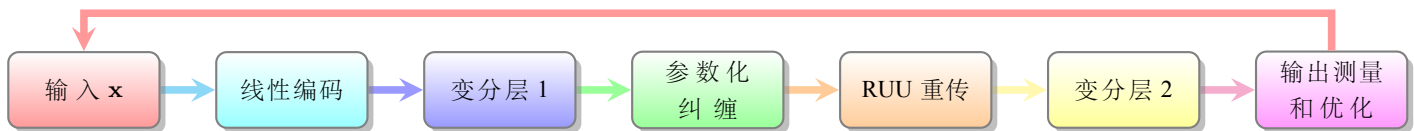


图 4.1 QMLP 算法流程图

在本次报告中，我们实现的量子机器学习模型为 QMLP（Quantum Multi-Layer Perceptron），流程如算法1和图4.1所示。

QMLP 是基于变分量子算法的量子神经网络模型，主要由以下几个部分组成：

#### 4.1.1 量子数据编码

QMLP 采用线性旋转编码将经典数据映射到量子态：

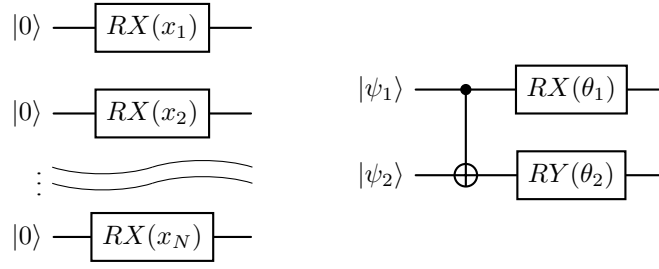


图 4.2 左：量子数据编码电路；右：参数化纠缠层电路

$$S(\mathbf{x}) = \bigotimes_{k=1}^N RX(x_k) \quad (4.1)$$

这个电路都构造非常简单，如图4.2左图所示。由此我们将经典数据映射到量子态，同时通过单比特旋转门独立编码每个输入特征，达到抑制噪声传播的目的。

#### 4.1.2 变分量子电路

在对输入进行量子编码后，便可以对其应用式4.2对应的变分量子电路进行参数化么正变换（类似于经典神经网络中的权重矩阵乘法）。

$$|\phi_L\rangle = \prod_{l=1}^L U_l(\theta_l) |\phi_0\rangle \quad (4.2)$$

$U_l(\theta_l)$  表示由一组可训练参数  $\theta_l$  控制的么正变换， $|\phi_0\rangle$  表示编码后的输入态。

然后相邻量子比特之间应用 CRX 门（式4.3）创建纠缠，增强局部关联性。电路如图4.2右图所示。

$$|\phi_L\rangle = \bigotimes_{m=1}^{N/2} CRX(\phi_m) |\phi_L\rangle \quad (4.3)$$

CRX 门的矩阵表达为式4.4。

$$CRX(\theta) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \cos \theta/2 & -i \sin \theta/2 \\ 0 & 0 & -i \sin \theta/2 & \cos \theta/2 \end{pmatrix} \quad (4.4)$$

#### 4.1.3 重上传单元

接着对每一位输入进行重传（式4.5），重复注入编码层  $S(\mathbf{x})$  在变分电路中生成输入的高阶项引入非线性（类似于经典电路的激活函数），增强量子神经网络的表达能力。

$$|\phi_L\rangle = R_l(\mathbf{x}) |\phi_L\rangle \quad (4.5)$$

$R_l$  表示所用的重传单元如  $RX, RY$  门

#### 4.1.4 测量与优化

通过 Pauli-Z 基测量（式4.6）将处理后的量子态转换为经典可读信号，然后便可应用经典方法如 Adam 优化算法等对模型参数进行优化。

$$\langle Z_i \rangle = \langle \phi_L | Z_i | \phi_L \rangle \in [-1, 1] \quad (4.6)$$

### 4.2 实现细节

出于效率的考量，我们采用了量子经典混合的架构，即首先由经典神经网络将 9 维的输入降维到 4 维，之后将这些数据输入到 QMLP 中，最后的测量结果就是输出。整个架构如图4.3之子图 1 所示，而其中的“Quantum Circuit”部分具体如图4.3之子图 2 所示。在子图 2 中， $Rot(\theta_i \sim \theta_{i+2})$  表示了一套需要三个量子参数的  $RX, RY, RZ$  门组合；而四个相连受控位表示一套 CRX 门组合，需要四个量子参数，分别如图4.3之子图 3,4 所示。显而易见，在这个架构中，经典参数数量为  $9 \times 4 + 4 = 40$ ，而量子参数数量为  $8 \times 3 + 2 \times 4 = 32$  个，总共 72 个参数。

4.3 性能评估

使用如图4.3的结构对数据集进行训练，得到如表4.1所示的实验结果。

表 4.1 量子经典混合模型的训练结果

| 迭代次数 | 训练准确率  | 测试准确率  | 测试 F1 分数 |
|------|--------|--------|----------|
| 50   | 75.98% | 76.60% | 0.6004   |
| 250  | 92.95% | 90.80% | 0.8880   |
| 500  | 93.33% | 90.90% | 0.8820   |

对比表3.1，我们可以显著发现量子经典混合模型有着更高的收敛速度，收敛之后的准确率与 F1 分数也更高。

5 性能对比与分析

5.1 性能对比

表 5.1 经典机器学习与量子机器学习的性能对比

| 模型          | 参数量 | 训练准确率  | 测试准确率  | 测试 F1 分数 |
|-------------|-----|--------|--------|----------|
| 经典小型 ResNet | 192 | 87.37% | 85.80% | 0.6662   |
| 量子经典混合 MLP  | 72  | 93.33% | 91.00% | 0.8820   |

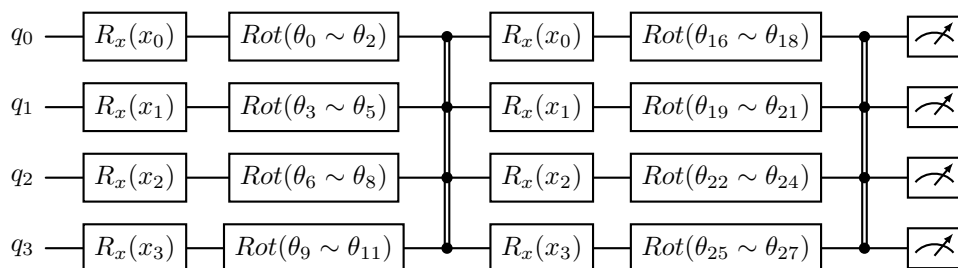
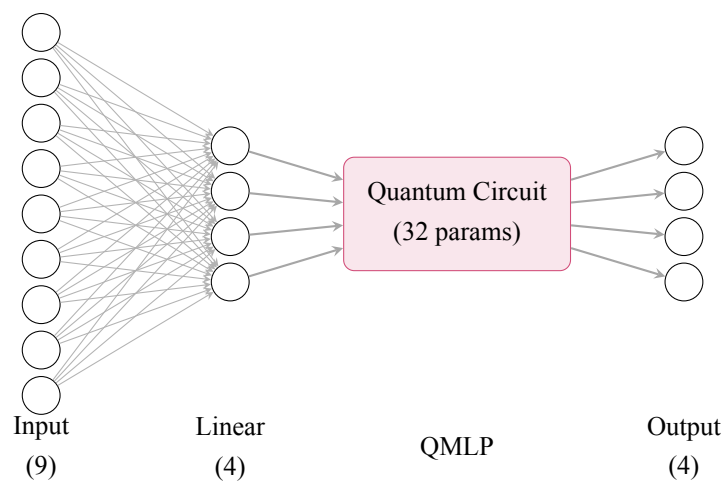
通过上表我们得知，混合了经典 MLP 与量子 MLP 的模型，在使用参数量远远小于经典小型 ResNet 的情况下，取得了更好的训练准确率、测试准确率和 F1 分数。我们可以认为，量子 MLP 在该数据集上取得了比经典 MLP 更好的性能。

5.2 差异分析

表 5.2 量子与经典神经网络核心组件对比

| 功能    | 经典神经网络组件  | QMLP 组件   |
|-------|---|---|
| 输入编码  | 数据标准化层<br>$\mathbf{x} \in \mathbb{R}^n \rightarrow \text{Norm}(\mathbf{x})$ | 单比特旋转门编码<br>$S(\mathbf{x}) = \bigotimes_k RX(x_k)$  |
| 特征变换  | 全连接层<br>$W\mathbf{x} + \mathbf{b}$  | 变分量子电路<br>$U(\theta) \phi\rangle$   |
| 非线性引入 | 激活函数<br>$\text{ReLU}(x) = \max(0, x)$                                       | 数据重上传单元 (RUU)<br>$R_l(\mathbf{x}) = RX(f_l(\mathbf{x}))$  |
| 参数优化  | 梯度下降<br>$\theta \leftarrow \theta - \eta \nabla_{\theta} \mathcal{L}$       | 参数偏移规则<br>$\nabla_{\theta} \langle Z \rangle \approx \frac{\langle Z \rangle_{\theta+\delta} - \langle Z \rangle_{\theta-\delta}}{2}$ |
| 输出处理  | Softmax 分类层<br>$p_c = \frac{e^{y_c}}{\sum e^{y_i}}$                         | Pauli-Z 测量<br>$\langle Z \rangle_i = \langle \psi   Z_i   \psi \rangle$   |

6 总结



$$\text{---} \boxed{Rot((\theta_i \sim \theta_{i+2}))} \text{---} = \text{---} \boxed{RX(\theta_i)} \boxed{RY(\theta_{i+1})} \boxed{RZ(\theta_{i+2})} \text{---}$$

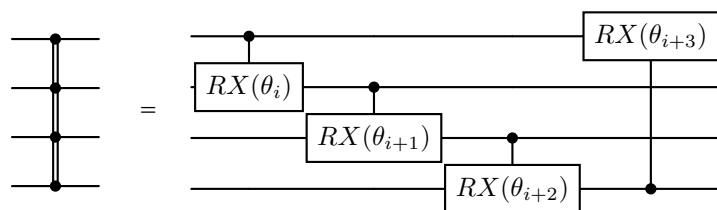


图 4.3 自上而下：子图 1：QuantumMLP 网络结构图：线性映射 + 参数化量子电路 + 测量输出；

子图 2：参数化量子电路的具体实现；

子图 3： $Rot(\theta_i \sim \theta_{i+2})$  门的具体实现；

子图 4： $CRX$  门组合的具体实现。