

Assignment 2: Density Matrix Reconstruction

Macha Hareesh
Enrollment No: 24114056
2nd Year, CSE

Objective

The primary goal of this assignment is to develop and train a computational model capable of reconstructing a density matrix ρ from measurement data while strictly enforcing physical constraints: Hermitian symmetry, Positive Semi-Definiteness (PSD), and Unit Trace.

1 Part 1: Model Selection & Training

Selected Track: Track 1 (Classical Shadows)

1.1 Architecture Choice

We utilized a **Transformer-based architecture**. The Transformer was chosen for its ability to capture global correlations between measurement snapshots (Classical Shadows) and the underlying quantum state.

- **Input:** A batch of simulated noisy density matrices (measurement estimates), represented as tensors of shape $(B, 2, N, N)$ where channels represent Real and Imaginary components.
- **Encoder:** A standard Transformer Encoder with multi-head self-attention. This processes the flattened density matrix as a sequence of tokens, allowing the model to learn dependencies between different basis elements.
- **Output Head:** A fully connected layer projects the simplified representation into a lower-triangular matrix form L .

2 Part 2: Enforcing Physical Constraints

To ensure the reconstructed density matrix $\hat{\rho}$ is physically valid, we employed the **Cholesky Decomposition** method.

2.1 Methodology

Instead of predicting ρ directly, the neural network outputs a complex lower-triangular matrix L . The density matrix is then constructed as:

$$\rho_{\text{unnormalized}} = LL^\dagger \quad (1)$$

By construction, LL^\dagger is always Hermitian and Positive Semi-Definite (PSD).

To enforce the Unit Trace constraint ($\text{Tr}(\rho) = 1$), we apply a normalization step:

$$\hat{\rho} = \frac{\rho_{\text{unnormalized}}}{\text{Tr}(\rho_{\text{unnormalized}})} \quad (2)$$

This rigorous mathematical structure guarantees that the output of the model is always a valid quantum state, regardless of the input noise.

3 Part 3: Submission Deliverables

3.1 Repository Structure

The project fits the required structure:

- `/src`: Contains `data.py`, `model.py`, `train.py`, and `evaluate.py`.
- `/outputs`: Stores the trained model weights (`model.pt`).
- `/docs`: Contains detailed Markdown documentation and this report.

3.2 AI Attribution and Usage Policy

Disclosure: This project was developed primarily by the author. **Gemini** (Google DeepMind) was used minimally for initial project setup and debugging assistance.

Usage Summary:

- **Planning:** Consulted for project structure best practices.
- **Setup:** Generation of initial directory hierarchy.
- **Debugging:** Assistance with resolving Python syntax errors (specifically complex number data types in PyTorch).

Verification: All AI-generated suggestions were manually reviewed. The core physical constraints (Cholesky logic) were implemented and verified by the author. Unit tests confirm the validity of the generated density matrices.

4 Part 4: Required Metrics

4.1 Performance Evaluation

The model was evaluated on a held-out test set of $N = 100$ random density matrices. The primary metrics for reconstruction quality are Quantum Fidelity $F(\rho, \hat{\rho})$ and Trace Distance $D(\rho, \hat{\rho})$.

Table 1: Mean Fidelity and Trace Distance on Test Set (100 Samples)

Metric	Value	Target Ideal
Mean Fidelity	0.99	1.00
Mean Trace Distance	0.005	0.00

4.2 Inference Latency

We measured the average time taken for the model to reconstruct a density matrix from the input measurement data.

- **Average Inference Latency:** **3.24 ms** per reconstruction.