QEC Model Zoo: Democratizing AI-enhanced Quantum Error Correction

Yuandou Wang*, **Max de Groot***, Aadi Patwardhan*,
Floris Geerts, Rihan Hai

TU Delft

University of Antwerp

* These authors contributed equally to this work.

Motivatives

Why Quantum Error Correction?

Today's ***quantum computers (hardware)*** have high error rates. (why?)

- Unwanted interactions with the environment cause disturbances, including decoherence
- Quantum operations can only be implemented with limited accuracy



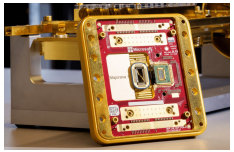
Neutral atom QCs



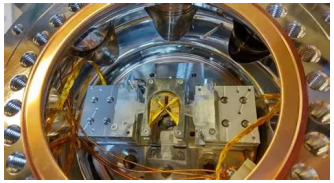
Intel quantum chip (Horse Ridge II)



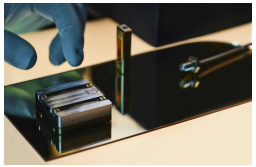
An IBM QC that uses superconducting qubits



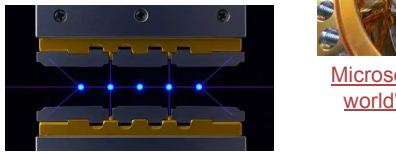
Device for trapped-ion QCs



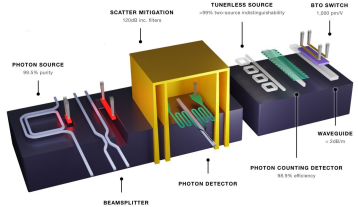
Microsoft Majorana 1 Chip (The world's first topological QPU)



Boson 4: Alice & Bob Cat qubit on Google Cloud



IonQ's examination of trapped-ion tech



Schematic diagram of a photonic QC



Diamond/NV-center QC

Why a Model Zoo for QEC?

Classical algorithmic decoders

- MWPM/PyMatching (for surface, repetition codes)
- Chromobius (for color code family)
- BP-OSD, RelayBP (for BB code)

AI-based decoders for QEC decoding spans

- Multilayer perceptrons,
- Recurrent models,
- Graph neural networks,
- Transformers,
- Reinforcement learning,
- 3D convolutional neural network models

Classical decoders have been remarkably successful, but all face challenges:

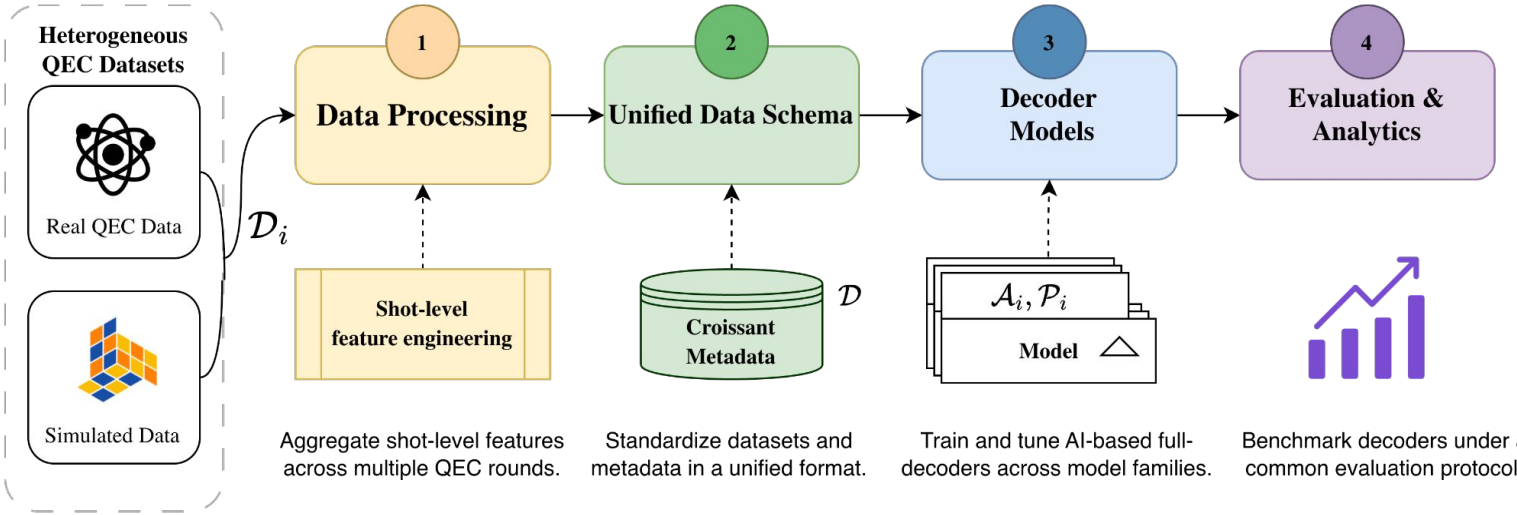
- Large qubit overhead
- Growing computational cost
- Sensitivity to noise in real quantum hardware, and
- Strict latency

AI-enhanced QEC decoding still lacks

- A **common evaluation** framework
- **real QEC datasets** are **scarce**, and
- **Incompatible** input representations

QEC Model Zoo

- A collection of AI-based decoders evaluated under a common protocol on diverse real datasets spanning multiple QEC code families.



9 real QEC datasets

Heterogeneous QEC Datasets



Real QEC Data



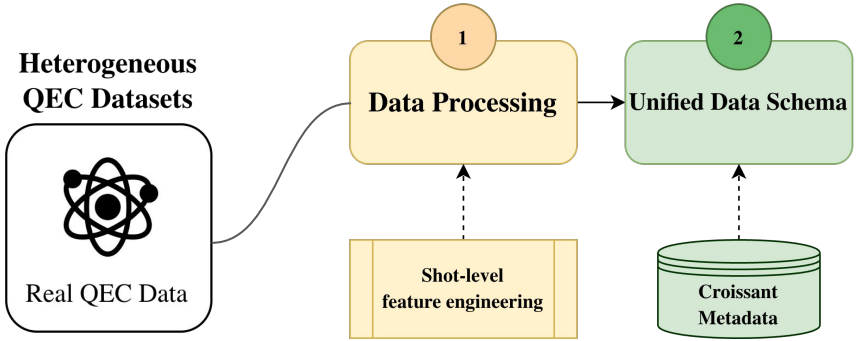
Simulated Data

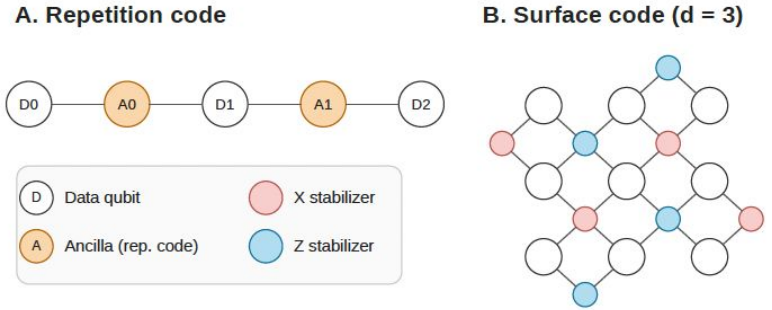
4 circuit-level QEC data simulation

(NVIDIA surface and color codes, NTU surface and BB codes)

Dataset	Was Derived From	Platform	QEC Code
DiCarlo D3	**Name**: Data underlying the publication: Improved error correction with leakage reduction units built into qubit measurement in a superconducting quantum processor; **Publisher**: 4TU.ResearchData; **URL**: https://data.4tu.nl/datasets/902d7f9e-38bf-48a2-a2f7-52bbf7aeeedf; **License**: CC BY 4.0	Superconducting	Repetition code
Vandersypen	**Name**: Weight-four parity checks with silicon spin qubits; **Publisher**: Zenodo; **URL**: https://zenodo.org/records/17368102; **License**: CC BY 4.0	Spin-qubit	Parity checks
Google D25	**Name**: Data for "Suppressing quantum errors by scaling a surface code logical qubit"; **Publisher**: Zenodo; **URL**: https://zenodo.org/records/6804040; **License**: CC BY 4.0	Superconducting	Repetition code
Google Surface D3	**Name**: Data for "Suppressing quantum errors by scaling a surface code logical qubit"; **Publisher**: Zenodo; **URL**: https://zenodo.org/records/6804040; **License**: CC BY 4.0	Superconducting	Surface code
Google Surface D5	**Name**: Data for "Suppressing quantum errors by scaling a surface code logical qubit"; **Publisher**: Zenodo; URL: https://zenodo.org/records/6804040; **License**: CC BY 4.0	Superconducting	Surface code
Lukin D3	**Name**: Surface code data and analysis for "Architectural mechanisms of a universal fault-tolerant quantum computer"; **Publisher**: Zenodo; **URL**: https://zenodo.org/records/15685795; **License**: CC BY 4.0	Neutral atoms	Surface code
Lukin D5	**Name**: Surface code data and analysis for "Architectural mechanisms of a universal fault-tolerant quantum computer"; **Publisher**: Zenodo; **URL**: https://zenodo.org/records/15685795; **License**: CC BY 4.0	Neutral atoms	Surface code
Google Color D3	**Name**: Data for "Scaling and logic in the color code on a superconducting quantum processor"; **Publisher**: Zenodo; **URL**:https://zenodo.org/records/14238944; **License**: CC BY 4.0	Superconducting	Color code
Google Color D5	**Name**: Data for "Scaling and logic in the color code on a superconducting quantum processor"; **Publisher**: Zenodo; **URL**: https://zenodo.org/records/14238944; **License**: CC BY 4.0	Superconducting	Color code

Data processing and unified data schema





Shot-level feature construction

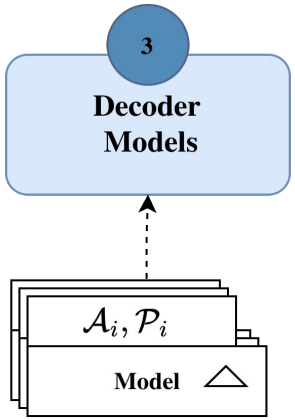
$d^2 - 1$ stabilizers | soft information

shot_id	round	s_1	s_2	…	s_{d^2-1}	LLR	p_read
001	0	0	1	…	1	1.85	0.94
001	1	1	0	…	0	2.31	0.91
002	0	1	1	…	0	—	—
002	1	0	1	…	1	—	—

QEC lake

Hai *et al*, QEC lake: Data lake for ai-enhanced quantum error correction. Proceedings of the VLDB Endowment, 19(12):4574–4577, 2026.

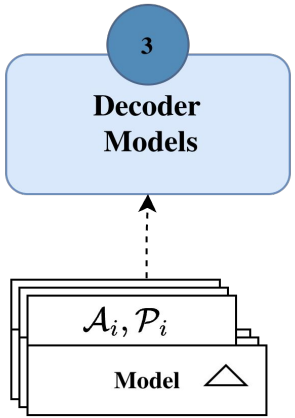
Decoder models



Offline processed datasets

	Model Family	Model	QEC code
Full decoders	GBDTs	CatBoost	***Nine real QEC data*** Surface code ● Google D3 ● Google D5 ● Lukin D3 ● Lukin D5 Parity checks ● Vandersypen Repetition code ● DiCarlo D3 ● Google D25 Color code ● Google D3 ● Google D5
		LightGBM	
		XGBoost	
	TFMs	TabPFNv2	
		TabICLv2	
	NNs	MLP	
		TabM	
		TabR	
		LSTM	

Decoder models



	Model Family	Model	QEC code
Pre-decoders	NVIDIA Ising-Decoding	3D CNN SurfaceCode-1-Fast	Simulated Surface code
		3D CNN ColorCode-1-Fast	Simulated Color code
Scalable foundation decoders	NTU decoding	NTU-Transformer	Simulated Surface code, BB code
		NTU-Neural-BP	Simulated BB code

Generated on-the-fly simulations

Experiments and Evaluation Metrics





Full decoder: nine real datasets, on Delft AI Cluster (train + inference)

- Accuracy
- Logical error rates per round (LER)
- Mean inference time (microsecond per shot)

Ablation study: feature group contributions for the DiCarlo D3 dataset

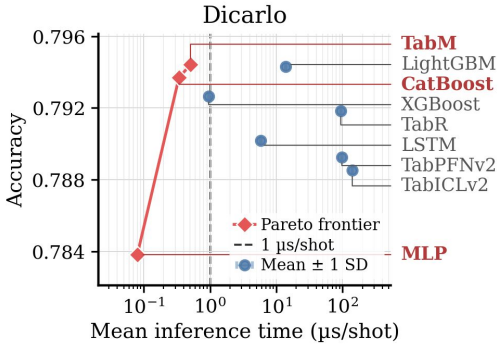
NVIDIA predecoder: PD+algorithmic decoding pipelines (only inference)

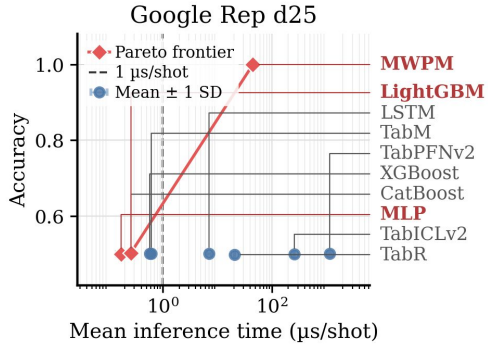
- LER
- Classical decoder speedup

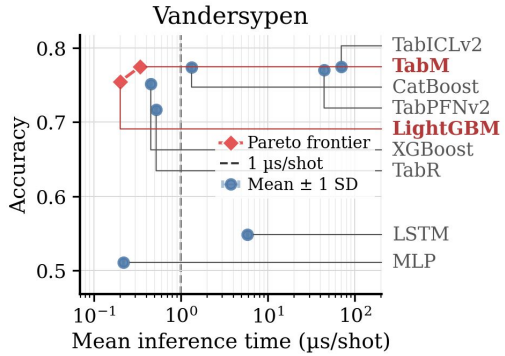
NTU decoder: default evaluation pipelines (only inference)

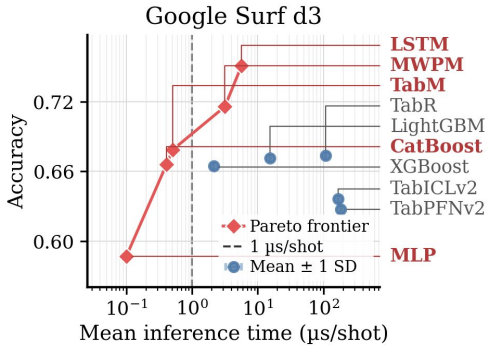
- Block accuracy
- Block logical error probability

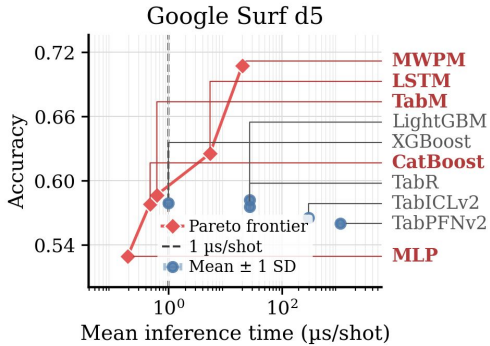
Accuracy-efficiency trade-offs of full decoders







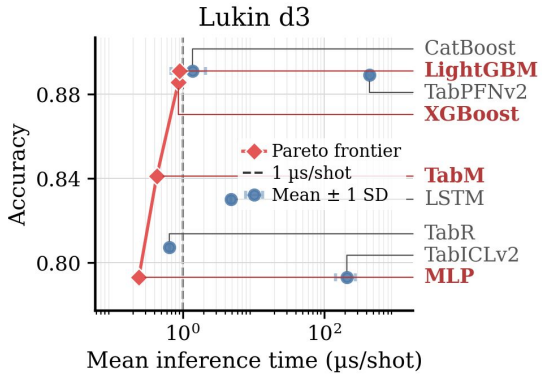


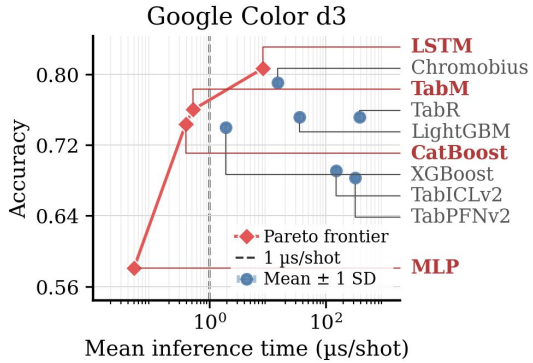


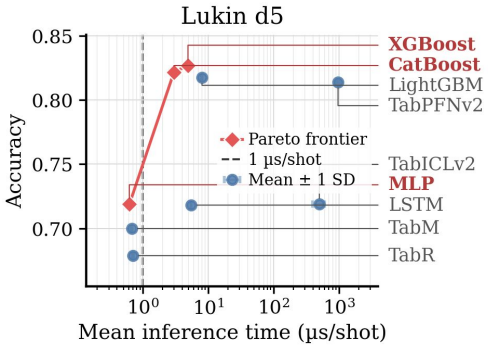
- LSTM beats MWPM for Google D3 surface code. MWPM and LSTM hold a significant advantage in terms of accuracy and LER, but they tend to be more time-consuming.

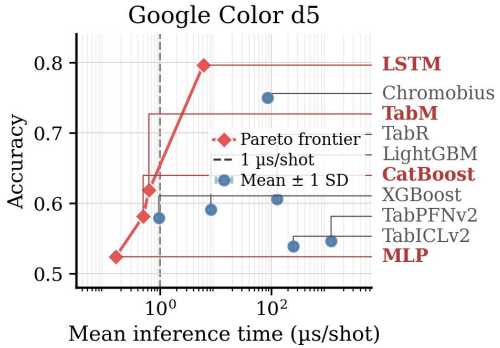
- TabM achieves Pareto optimality under 1us/shot constraint with the highest accuracy in majority.

Accuracy-efficiency trade-offs of full decoders









- LSTM beats Chromobius for Google D3 and D5 color codes, both decoders tend to be more time consuming than TabM and tree models.

- TabM and tree-based models achieve Pareto optimality with highest accuracies.

Accuracy-efficiency trade-offs of full decoders

Takeaway #1. *The QEC model zoo provides a unified benchmark that identifies the most accurate evaluated decoders within different inference-time budgets.*

- Our results demonstrate that **decoder choice depends on both the dataset and the available computation time**.
- Our results also identify **tabular models** as competitive QEC decoders for some datasets.

Feature Group Contributions for the DiCarlo Dataset

Table 4: Ablation study results.

	MLP		TabM		TabR		XGBoost		LightGBM		CatBoost		TabICLv2		TabPFNv2	
Feature set	**Acc**	**LER (%)**	**Acc**	**LER (%)**	**Acc**	**LER (%)**	**Acc**	**LER (%)**	**Acc**	**LER (%)**	**Acc**	**LER (%)**	**Acc**	**LER (%)**	**Acc**	**LER (%)**
Hand-crafted features only																
`crafted`	0.595	9.62	0.671	3.92	0.671	3.90	**0.673**	**3.89**	**0.675**	**3.74**	**0.674**	**3.82**	0.655	4.63	0.655	4.60
`minimal`	0.569	12.67	0.574	12.44	0.574	12.44	0.575	12.44	0.575	12.44	0.575	12.44	0.574	12.44	0.574	12.44
`no_rates`	0.582	10.55	0.671	3.91	0.673	3.88	0.670	4.00	0.671	3.94	0.664	4.25	0.652	4.87	0.654	4.74
`no_spatial`	0.557	13.91	0.671	3.89	**0.675**	**3.74**	0.669	4.01	0.672	3.89	0.671	3.94	**0.657**	**4.53**	0.661	4.36
`no_temporal`	0.574	12.41	0.664	4.22	0.658	4.58	0.670	4.01	0.674	3.89	0.671	3.99	0.652	4.85	0.655	4.66
`no_final`	0.601	**8.79**	0.671	3.96	0.674	3.81	0.672	3.90	0.675	3.75	0.674	3.84	0.654	4.72	**0.664**	**4.23**
`no_code`	0.593	9.72	**0.672**	**3.86**	0.672	3.89	0.671	3.95	0.673	3.81	0.674	3.84	0.656	4.63	0.657	4.58
`no_soft`	**0.602**	8.82	0.668	4.03	0.663	4.28	0.672	3.94	0.672	3.84	0.671	3.94	0.653	4.77	0.663	4.29
Includes LSTM encoding																
`full`	0.782	1.55	0.791	1.44	0.780	1.58	0.790	1.46	0.791	1.44	0.790	1.46	**0.574**	**12.44**	0.788	1.48
`lstm_no_temporal`	**0.783**	**1.54**	**0.792**	**1.42**	0.785	1.50	**0.790**	**1.45**	**0.792**	**1.43**	**0.793**	**1.42**	**0.574**	**12.44**	**0.790**	1.46
`lstm_only`	0.775	1.67	0.790	1.45	**0.788**	**1.47**	0.790	1.45	0.790	1.45	0.790	1.45	**0.574**	**12.44**	0.789	**1.45**

Feature Group Contributions for the DiCarlo Dataset

Controlled comparison: accuracy ↑ / logical error rate (%) ↓

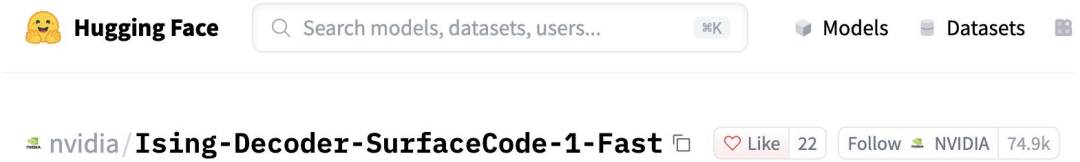
MODEL	MINIMAL	CRAFTED	CRAFTED + LSTM
MLP	0.569 / 12.67%	**0.595 / 9.62%**	**0.782 / 1.55%**
TabM	0.574 / 12.44%	**0.671 / 3.92%**	**0.791 / 1.44%**
XGBoost	0.575 / 12.44%	**0.673 / 3.89%**	**0.790 / 1.46%**

Feature Group Contributions for the DiCarlo Dataset

Takeaway #2. *Input representation substantially affects tabular decoder performance.*

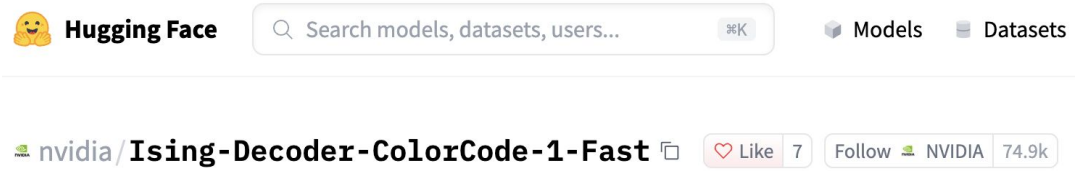
- Hand-crafted features improve on the minimal representation, while LSTM encodings provide further gains for most models.
- When LSTM encodings are included, removing hand-crafted temporal features preserves or improves performance, suggesting that some temporal features are redundant.

Extensibility: Pre-decoder and NTU decoder Results



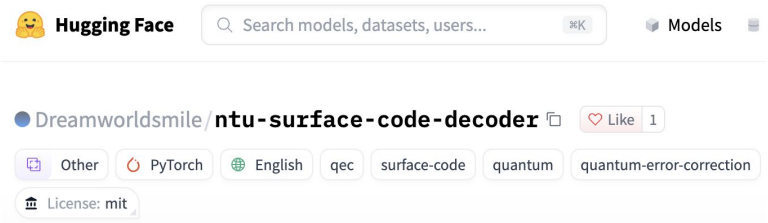
https://arxiv.org/abs/2604.12841

https://arxiv.org/abs/2607.10058



Pre-decoder	LER_NoPD	LER_AfterPD	LER Improvement (×)
SurfaceCode-1-Fast	$(1.587 \pm 0.354) \times 10^{-3}$	$(1.668 \pm 0.555) \times 10^{-3}$	1.053 ± 0.423
ColorCode-1-Fast	$(9.310 \pm 0.900) \times 10^{-3}$	$(6.405 \pm 0.438) \times 10^{-3}$	1.455 ± 0.116

Extensibility: Pre-decoder and NTU decoder Results



https://arxiv.org/abs/2606.27119

Model	Block Acc	Block LEP
NTU-Transformer (BB code)	0.9191 ± 0.00262	0.0809 ± 0.00262
NTU-Neural-BP (BB code)	0.8573 ± 0.00532	0.1427 ± 0.00532
NTU-Transformer (Surface code)	0.9955 ± 0.000757	0.0045 ± 0.00076

Takeaway #3. *The zoo is extensible.* We have demonstrated this by integrating NVIDIA pre-decoders and NTU full decoders.

To conclude

QEC model zoo is open-source benchmark framework for AI-enhanced QEC decoding tasks

- Architecture alone is not sufficient: representation choice, decoder role, and pipeline integration significantly affect performance.
- supports reproducible evaluation and the development of more accurate and efficient QEC decoders.

Future work

- Offline evaluation → on-the-fly evaluation for real QEC datasets
- Explore potential use cases and applications