

Quantum-Assisted Agentic Learning: QAOA-Based Trust-Aware Coordination for Reliable Distributed Intelligence

Dr. Kishor Kumar Gajula¹, Associate Professor,

Smitha Karnewar², M.Tech(CSE) Student,

^{1&2}Department of Computer Science and Engineering,

Mother Theresa College of Engineering & Technology, Peddapalli, Telangana, India

drkishorkumarg@gmail.com, <https://orcid.org/0009-0003-8141-3332>

Abstract—Distributed intelligent systems rely on many autonomous agents that train a shared model on their own data, and a single faulty or malicious agent can corrupt that model. This paper proposes Quantum-Assisted Agentic Learning (QAAL), in which a coordinator agent perceives the agreement between agents' updates, keeps a trust memory of each agent, and plans which agents to aggregate by solving a quadratic unconstrained binary optimisation (QUBO) problem with a constraint-preserving quantum approximate optimisation algorithm (QAOA). QAAL was evaluated in simulation with 12 agents, non-IID data, random dropouts and three Byzantine attacks on two classification tasks, against FedAvg, coordinate-wise median, trimmed mean and Multi-Krum. With 4 of 12 agents faulty, QAAL reached 83.4% mean accuracy versus 55.2% for FedAvg and 72.5% for trimmed mean, and matched Multi-Krum (84.0%). The QAOA planner found the optimal agent subset in 98.7% of 1260 training rounds, and its trust memory separated honest from faulty agents with AUC up to 0.996. At the 12-agent scale tested, classical solvers are equally accurate and far faster, so no quantum speed-up is claimed.

Keywords—agentic AI, federated learning, Byzantine robustness, QAOA, QUBO, quantum optimisation, distributed intelligence.

I. INTRODUCTION

Intelligence is moving from central data centres to fleets of autonomous agents: edge devices in smart factories, vehicles, hospitals and power grids that learn from local data and cooperate to build a shared model. Federated learning (FL) [1], [7] is the standard way to do this without moving raw data, and agent-based AI systems increasingly add autonomy on top—perceiving their environment, keeping memory and planning actions [18], [19]. The weak point is reliability. In a large distributed system some agents will be faulty, compromised or simply unavailable, and a plain average of their updates can be destroyed by a single malicious participant—the classical Byzantine problem [8]—and attacks can be crafted to evade simple defences [6].

Robust rules—median and trimmed mean [3], Multi-Krum [2], Bulyan [4], trust bootstrapping [5]—limit bad updates but are memory-less filters that suffer when honest agents hold very different (non-IID) data [22]. Choosing the most mutually consistent subset of agents is naturally a combinatorial problem: with n agents there are $C(n, k)$ possible subsets, and the choice can be written as a quadratic unconstrained binary optimisation

(QUBO) problem [11]. QUBOs are the native input of the quantum approximate optimisation algorithm (QAOA) [9], [10], one of the most studied variational algorithms for near-term quantum hardware [14], [16], [17], [20].

This paper connects the two lines of work. We propose **Quantum-Assisted Agentic Learning (QAAL)**, in which a coordinator agent with perception, memory and planning selects trustworthy agents each round by solving a QUBO with a constraint-preserving QAOA. The contributions are:

- An agentic coordinator combining a trust graph, a reputation memory and QUBO planning.
- A QUBO for reliable agent selection, solved by constraint-preserving QAOA (Dicke state, XY mixer).
- An evaluation with 12 agents, non-IID data, dropouts and three attacks against four baselines.
- A QAOA depth study against random sampling and a greedy classical solver.

II. LITERATURE REVIEW

Table I summarises related work. Statistical rules [2]–[4] degrade under strong heterogeneity [22]; FLTrust [5] needs clean server data; quantum federated learning [15] trains quantum models rather than protecting aggregation; and QAOA has run on superconducting processors [20], with constraint-preserving mixers [10], [12] suited to fixed-size selection.

TABLE I. SUMMARY OF RELATED WORK

Ref	Author, year	Method	Key finding	Limitation
[1]	McMahan et al., 2017	FedAvg	Communication-efficient distributed training	Not robust to a single bad agent
[2]	Blanchard et al., 2017	Krum / Multi-Krum	Provable Byzantine tolerance	Memory-less; hurt by non-IID data
[3]	Yin et al., 2018	Median, trimmed mean	Optimal statistical rates	Weak when many agents are faulty
[4]	El Mhamdi et al., 2018	Bulyan	Defends against hidden high-dimensional attacks	Needs $n \geq 4f + 3$ agents
[5]	Cao et al., 2021	FLTrust	Trust scores from a server dataset	Requires clean root data
[22]	Karimireddy et al., 2022	Bucketing	Robust rules fail on heterogeneous data; bucketing helps	Randomised; no agent memory
[15]	Chen & Yoo, 2021	Federated quantum ML	Trains VQCs across clients	No Byzantine robustness
[20]	Harrigan et al., 2021	QAOA on hardware	QAOA runs on a 23-qubit processor	Quality drops on dense problems

III. PROBLEM STATEMENT

n agents hold private datasets $\mathcal{D}_i$ and, in round t , send updates Δw_i from the global model w_t ; up to f are Byzantine and honest agents may be unavailable. The goal is an aggregation rule that (i) keeps the global model accurate when f is large, (ii) does not need clean server data, (iii) uses information from past rounds, and (iv) is expressed in a form that quantum optimisers can solve. Existing rules meet (i)–(ii) only partly under non-IID data and ignore (iii)–(iv).

IV. PROPOSED METHODOLOGY

Fig. 1 shows the workflow: agents train locally; the coordinator perceives, remembers, plans with QAOA and aggregates.

A. Learner Agents

Each available agent i starts from w_t , runs E epochs of mini-batch SGD with learning rate η on its own data, and returns

$$\Delta w_i = w_i^{(E)} - w_t, \quad w_i^{(e+1)} = w_i^{(e)} - \eta \nabla \ell(w_i^{(e)}; \mathcal{D}_i) \quad (1)$$

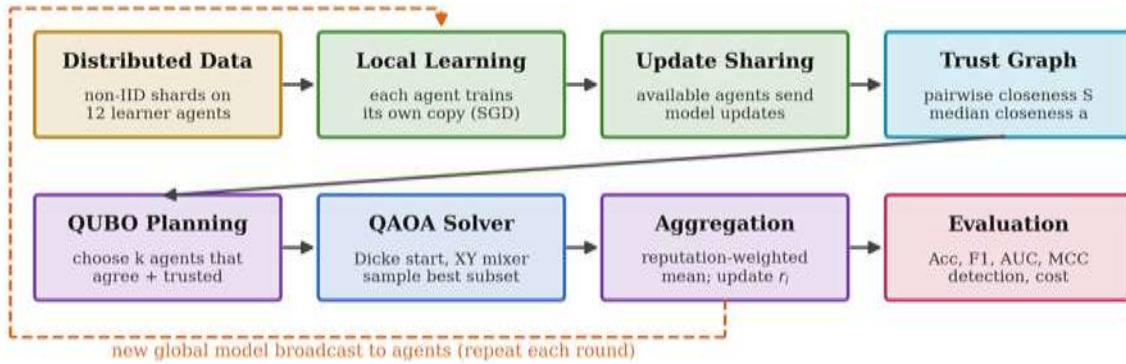


Fig. 1. Workflow of Quantum-Assisted Agentic Learning (QAAL).

B. Coordinator Perception: Trust Graph

The coordinator measures how close each pair of updates is, and how close each update is to the coordinate-wise median m :

$$S_{ij} = \exp(-\frac{\|\Delta w_i - \Delta w_j\|^2}{\sigma^2}), \quad a_i = \exp(-\frac{\|\Delta w_i - m\|^2}{\sigma_m^2}) \quad (2)$$

where σ^2 and σ_m^2 are the median squared distances, so the scores adapt to the scale of each round.

C. Memory and Planning: QUBO Formulation

The coordinator keeps a reputation r_i for every agent, a moving average of its median closeness: $r_i \leftarrow 0.8 r_i + 0.2 a_i$. Basing reputation on closeness rather than on past selection avoids a feedback loop in which once-selected agents keep being selected. Selection of k agents is then the QUBO

$$C(x) = -\sum_{i<j} S_{ij}x_ix_j - \sum_i (\mu a_i + \rho r_i)x_i, \sum_i x_i = k, x \in \{0,1\}^n \quad (3)$$

with $\mu = \rho = 2$ and $k = 7$. The first term rewards choosing agents that agree with each other; the second rewards agents close to the median and with a good history.

D. Quantum Solver: Constraint-Preserving QAOA

Instead of adding a penalty for the constraint, QAAL starts in the Dicke state $|D_k^n\rangle$, the equal superposition of all subsets of size k [13], and alternates the cost phase with a ring XY mixer that preserves the number of selected agents [10], [12]:

$$|\psi(\gamma, \beta)\rangle = \prod_{l=1}^p e^{-i\beta_l H_{XY}} e^{-i\gamma_l C} |D_k^n\rangle, H_{XY} = \frac{1}{2} \sum_i (X_i X_{i+1} + Y_i Y_{i+1}) \quad (4)$$

The angles are tuned by COBYLA to minimise $\langle \psi | C | \psi \rangle$, warm-started from the previous round. The circuit is then sampled 1024 times, and the sampled subset $\mathbf{x}^*$ with the lowest cost is chosen. The global model is updated with the mean of the selected updates:

$$w_{t+1} = w_t + \frac{1}{k} \sum_{i \in \mathbf{x}^*} \Delta w_i \quad (5)$$

E. Algorithm

Algorithm 1: QAAL coordination

```

1 Input: n agents, rounds T, subset size k, QAOA depth p
2 initialise global model w, reputations r_i = 0.5
3 for t = 1..T:
4   for each available agent i: dw_i <- LocalSGD(w, D_i)    # act
5   S, a <- TrustGraph({dw_i})                             # perceive
6   build QUBO C(x) from S, a, r; constraint sum x = k      # plan
7   (gamma, beta) <- COBYLA( <psi|C|psi> ), warm start
8   x* <- lowest-cost subset among 1024 QAOA samples
9   w <- w + mean_{i in x*} dw_i                             # aggregate
10  r_i <- 0.8 r_i + 0.2 a_i                                 # remember
11 Output: global model w, reputations r

```

V. SYSTEM ARCHITECTURE

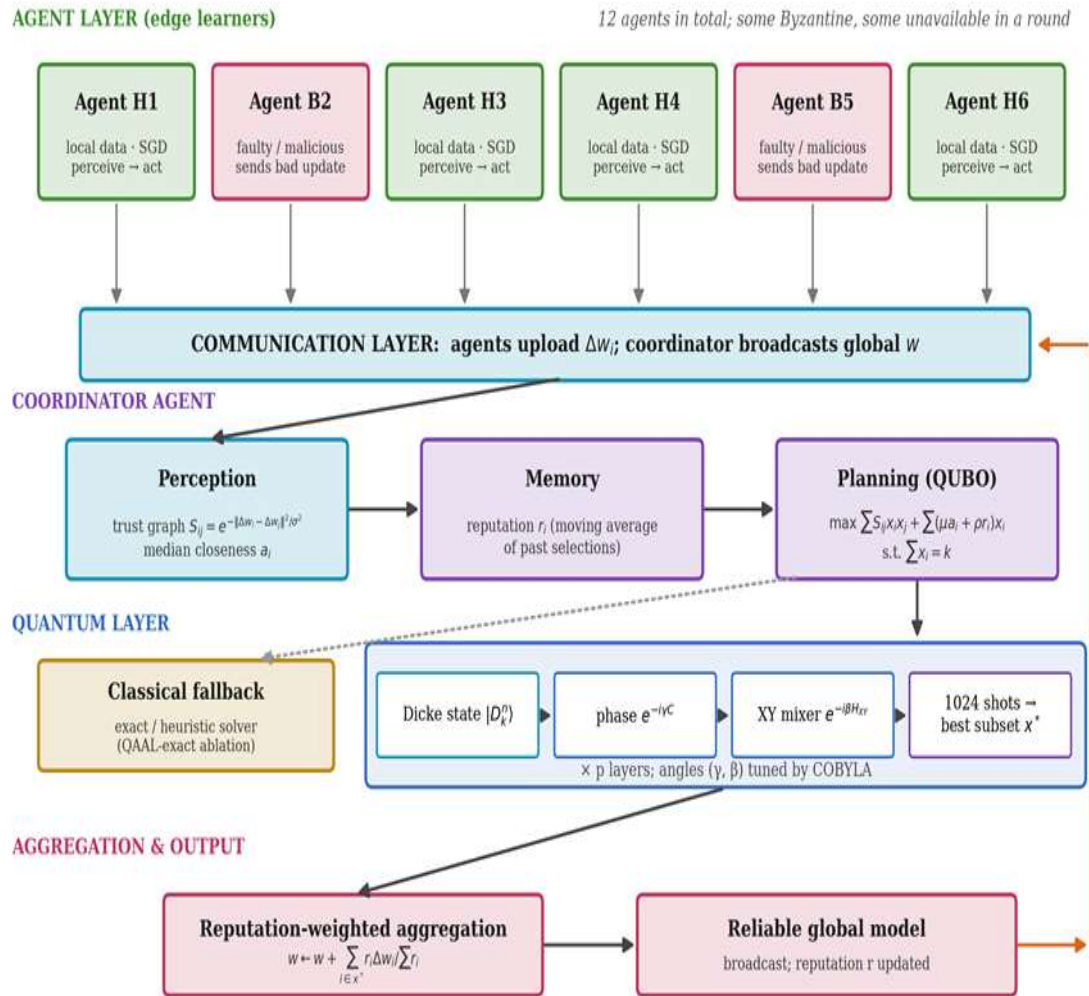


Fig. 2. System architecture of QAAL.

Fig. 2 shows five layers: the **agent layer** of learners (some faulty); the **communication layer** carrying updates and the global model; the **coordinator agent** with perception, memory and planning; the **quantum layer** running QAOA, with a classical exact solver as fallback; and the **aggregation layer** producing the reliable global model.

VI. IMPLEMENTATION

The system was implemented in Python. Learner agents use PyTorch, and QAOA runs on an exact statevector simulator written in NumPy (up to 12 qubits, one per agent). Tasks: *Digits* (1797 8×8 images, 10 classes) [21] and *IoT-Sensor* (synthetic, 6000 samples, 40 features, 5 classes, standing in for sensor telemetry). 75% of each was split over 12 agents by a Dirichlet(0.5) non-IID partition [23]; 25% formed the test set. Attacks were applied by 2 or 4 agents: *sign-flip* (update multiplied by -3), *Gaussian* (random update of three times the honest norm) and *label-flip* (label y replaced by $C-1-y$). Each honest agent was also unavailable with probability 0.1 per round. Each scenario used three seeds (Table II).

TABLE II. IMPLEMENTATION PARAMETERS

Parameter	Setting
Environment	Python 3.11, PyTorch, NumPy/SciPy; 2-core CPU; exact statevector QAOA simulation (no hardware noise)
Agents / model	12 agents, 30 rounds, dropout 0.1; MLP (input=32=classes), ReLU, cross-entropy
Local optimiser	SGD, learning rate 0.1, batch size 32, 2 local epochs
QUBO	$k = 7$ agents, $\mu = \rho = 2$, reputation decay 0.8
QAOA	Depth $p = 3$, Dicke start + XY ring mixer, COBYLA (≤ 120 evaluations), 1024 shots
Baselines	FedAvg, median, trimmed mean ($\beta = 4$), Multi-Krum ($f = 4$, $m = 7$)

VII. RESULTS AND PERFORMANCE EVALUATION

Table III reports test accuracy with 4 of 12 agents faulty, together with the attack-free case. FedAvg collapses under sign-flipping (12.8% on Digits), and median and trimmed mean lose 20–25 points. QAAL keeps 92.3% and 74.8% and is within about one point of Multi-Krum in every attacked scenario. Averaged over the six attacked scenarios with four faulty agents, QAAL improves on FedAvg by 28.2 points and on trimmed mean by 10.9 points, and trails Multi-Krum by 0.6 points. With two faulty agents the averages are 84.4% (QAAL) and 84.6% (Multi-Krum). Without attacks, all selective methods lose accuracy to FedAvg because they discard some honest non-IID updates.

TABLE III. TEST ACCURACY (%) WITH 4 OF 12 AGENTS FAULTY, MEAN $\pm$ STD OVER 3 SEEDS

Model	Digits: clean	sign-flip	Gaussian	label-flip	IoT: clean	sign-flip	Gaussian	label-flip
FedAvg [1]	95.7 $\pm$ 0.4	12.8 $\pm$ 4.1	92.3 $\pm$ 1.5	74.1 $\pm$ 6.8	81.3 $\pm$ 0.7	20.3 $\pm$ 0.2	73.1 $\pm$ 4.5	58.4 $\pm$ 15.2
Median [3]	93.0 $\pm$ 0.9	67.2 $\pm$ 9.3	89.3 $\pm$ 2.3	84.1 $\pm$ 1.9	76.6 $\pm$ 0.2	53.5 $\pm$ 5.2	71.9 $\pm$ 1.2	61.7 $\pm$ 4.9
Trimmed mean [3]	93.2 $\pm$ 0.8	69.8 $\pm$ 5.7	90.0 $\pm$ 2.0	86.0 $\pm$ 3.1	77.6 $\pm$ 0.2	52.8 $\pm$ 5.5	72.6 $\pm$ 1.7	63.7 $\pm$ 3.3
Multi-Krum [2]	92.4 $\pm$ 1.2	92.8 $\pm$ 1.4	94.0 $\pm$ 0.6	93.3 $\pm$ 0.8	73.0 $\pm$ 1.4	75.0 $\pm$ 1.2	76.2 $\pm$ 1.2	72.8 $\pm$ 2.1
QAAL w/o memory	92.2 $\pm$ 1.6	92.8 $\pm$ 1.4	94.0 $\pm$ 0.6	92.5 $\pm$ 0.1	72.8 $\pm$ 1.5	74.8 $\pm$ 1.2	76.3 $\pm$ 1.4	71.9 $\pm$ 3.4
QAAL, exact solver	92.0 $\pm$ 1.4	92.6 $\pm$ 1.2	94.0 $\pm$ 0.6	92.6 $\pm$ 0.2	72.8 $\pm$ 1.6	74.8 $\pm$ 1.2	76.5 $\pm$ 1.1	69.5 $\pm$ 3.3
QAAL (proposed)	91.9 $\pm$ 1.4	92.3 $\pm$ 1.8	94.0 $\pm$ 0.5	92.6 $\pm$ 0.2	72.8 $\pm$ 1.6	74.8 $\pm$ 1.2	76.5 $\pm$ 1.1	70.3 $\pm$ 4.3

Table IV gives the full metric set averaged over all attacked scenarios; QAAL and Multi-Krum lead on every metric.

TABLE IV. MACRO PRECISION, RECALL, SPECIFICITY, F1, AUC AND MCC (MEAN OVER ALL ATTACKED SCENARIOS)

Model	Accuracy	Precision	Recall	Specificity	F1	AUC	MCC
FedAvg [1]	0.639	0.607	0.640	0.932	0.605	0.844	0.581
Median [3]	0.767	0.778	0.769	0.953	0.752	0.947	0.729
Trimmed mean [3]	0.777	0.787	0.780	0.954	0.763	0.950	0.741
Multi-Krum [2]	0.843	0.856	0.846	0.966	0.842	0.972	0.815
QAAL w/o memory	0.841	0.854	0.844	0.965	0.840	0.971	0.813

Model	Accuracy	Precision	Recall	Specificity	F1	AUC	MCC
QAAL, exact solver	0.839	0.852	0.841	0.965	0.837	0.971	0.810
QAAL (proposed)	0.839	0.853	0.842	0.965	0.837	0.971	0.811

Table V measures how well each selective method filters faulty agents with four attackers, and the aggregation cost per round. Multi-Krum excluded 88.2% of faulty updates and QAAL 85.1%. QAAL's simulated QAOA took about 520 ms per round, against under 3 ms for classical exact enumeration and under 1 ms for Multi-Krum.

TABLE V. FAULTY-AGENT FILTERING (4 FAULTY) AND AGGREGATION COST

Method	Faulty excluded (recall)	Exclusion precision	Honest kept	Time / round (ms)
Multi-Krum [2]	0.882	0.849	0.912	0.77
QAAL w/o memory	0.857	0.825	0.898	523
QAAL, exact solver	0.847	0.815	0.893	2.89
QAAL (proposed)	0.851	0.819	0.895	520

Fig. 3 collects the charts. Panels (a)–(b) compare accuracy; (c) shows convergence on Digits under sign-flipping, where QAAL and Multi-Krum rise together while FedAvg never learns; (d) shows QAAL's average gain over each baseline; (e) shows that the final reputation scores separate honest from faulty agents (AUC 0.996 for sign-flip, 0.995 for Gaussian, 0.943 for label-flip); and (f) shows QAOA quality versus depth. Fig. 4 gives the agent-level confusion matrices of QAAL.

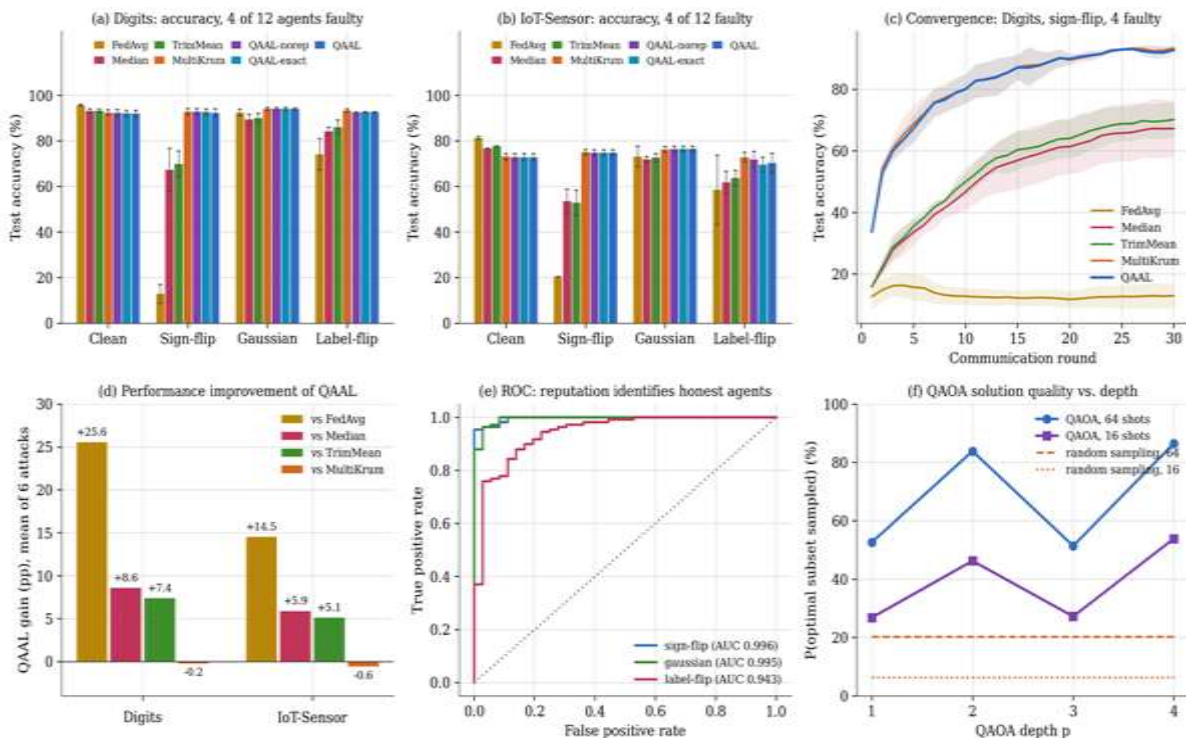


Fig. 3. (a)–(b) Accuracy comparison; (c) convergence; (d) performance improvement; (e) ROC of reputation; (f) QAOA quality vs. depth.

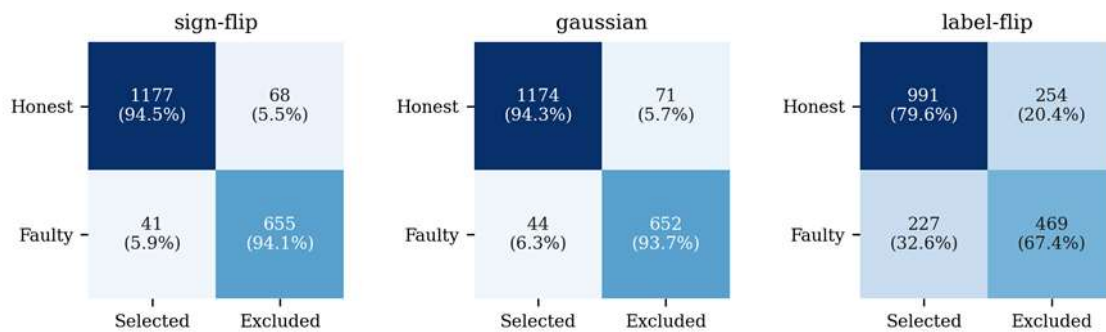


Fig. 4. Agent-level confusion matrices of QAAL with 4 faulty agents (all rounds, both datasets).

Fig. 3(f) evaluates the quantum solver on 40 QUBO instances recorded during training (mean 11.2 agents, about 462 feasible subsets). At depth $p = 2$, QAOA raised the probability of the optimal subset 15.4-fold over uniform sampling, so 64 shots contain the optimum 83.8% of the time against 20.2% for random sampling; at $p = 4$ the gain is 22.3-fold, and approximation ratios are ≥ 0.993 at every depth. During the full training runs ($p = 3$ with warm starts) QAOA returned the exact optimum in 98.7% of 1260 rounds, which is why QAAL and QAAL-exact perform alike. Quality is not monotonic in p because COBYLA sometimes stops in a poorer local optimum at $p = 3$. A classical greedy solver also found the optimum in every instance, so at this size the quantum solver offers no advantage.

Discussion. QAAL turns robust aggregation into an explicit agentic planning problem that quantum hardware can solve, and in simulation it performs on par with the strongest classical rule while clearly outperforming averaging, median and trimmed mean. Label-flipping remains the hardest case: faulty agents that train honestly on wrong labels produce plausible, mutually similar updates, and QAAL excluded only 67.4% of them. The reputation memory did not change accuracy measurably (QAAL vs. QAAL w/o memory), although it gave an accurate trust ranking. Limitations are the small scale (12 agents, fully simulable), noise-free QAOA simulation, two tasks, and the absence of adaptive attacks designed against QAAL. Target applications are IoT fleets, connected vehicles and multi-hospital learning.

VIII. CONCLUSION AND FUTURE WORK

This paper proposed Quantum-Assisted Agentic Learning, in which a coordinator agent with a trust graph, a reputation memory and QUBO planning selects reliable agents each round using a constraint-preserving QAOA. In experiments with 12 agents, non-IID data, dropouts and three attacks, QAAL kept 83.4% average accuracy with a third of agents faulty, 28.2 points above FedAvg and 10.9 above trimmed mean, and matched Multi-Krum. QAOA found the optimal subset in 98.7% of rounds. Classical solvers were equally accurate and much faster at this scale, so the value of QAAL lies in a hardware-ready formulation rather than a present-day speed-up. Future work will run QAAL on quantum hardware with noise, scale to hundreds of agents where exact enumeration is impossible, add adaptive attacks, and combine QUBO planning with learned agent policies.

REFERENCES

- [1]. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, “Communication-efficient learning of deep networks from decentralized data,” in *Proc. AISTATS*, PMLR 54, 2017, pp. 1273–1282.
- [2]. P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer, “Machine learning with adversaries: Byzantine tolerant gradient descent,” in *Adv. Neural Inf. Process. Syst. (NeurIPS)*, vol. 30, 2017.
- [3]. K. K. Gajula, “Quantum-Enhanced Cryptographic Protocols for Ultra-Secure Data Transmission in Next-Generation Networks,” *Journal of Dalian University of Technology*, vol. 31, no. 12, 2024.
- [4]. D. Yin, Y. Chen, K. Ramchandran, and P. Bartlett, “Byzantine-robust distributed learning: Towards optimal statistical rates,” in *Proc. ICML*, PMLR 80, 2018, pp. 5650–5659.
- [5]. E. M. El Mhamdi, R. Guerraoui, and S. Rouault, “The hidden vulnerability of distributed learning in Byzantium,” in *Proc. ICML*, PMLR 80, 2018, pp. 3521–3530.
- [6]. X. Cao, M. Fang, J. Liu, and N. Z. Gong, “FLTrust: Byzantine-robust federated learning via trust bootstrapping,” in *Proc. NDSS*, 2021, doi: 10.14722/ndss.2021.24434.
- [7]. K. K. Gajula, “Hierarchical quantum learning: A modular two-level variational framework for trainable and scalable intelligent systems,” *International Journal for Research in Science & Advanced Technologies*, vol. 26, no. 2, 2026, doi: 10.65726/ijrsat.2026.V26.I02.10.
- [8]. M. Fang, X. Cao, J. Jia, and N. Gong, “Local model poisoning attacks to Byzantine-robust federated learning,” in *Proc. USENIX Security*, 2020, pp. 1605–1622.
- [9]. P. Kairouz *et al.*, “Advances and open problems in federated learning,” *Found. Trends Mach. Learn.*, vol. 14, no. 1–2, pp. 1–210, 2021, doi: 10.1561/22000000083.
- [10]. L. Lamport, R. Shostak, and M. Pease, “The Byzantine generals problem,” *ACM Trans. Program. Lang. Syst.*, vol. 4, no. 3, pp. 382–401, 1982, doi: 10.1145/357172.357176.
- [11]. E. Farhi, J. Goldstone, and S. Gutmann, “A quantum approximate optimization algorithm,” arXiv:1411.4028, 2014.
- [12]. S. Hadfield *et al.*, “From the quantum approximate optimization algorithm to a quantum alternating operator ansatz,” *Algorithms*, vol. 12, no. 2, Art. 34, 2019, doi: 10.3390/a12020034.
- [13]. A. Lucas, “Ising formulations of many NP problems,” *Front. Phys.*, vol. 2, Art. 5, 2014, doi: 10.3389/fphy.2014.00005.
- [14]. Z. Wang, N. C. Rubin, J. M. Dominy, and E. G. Rieffel, “XY mixers: Analytical and numerical results for the quantum alternating operator ansatz,” *Phys. Rev. A*, vol. 101, Art. 012320, 2020, doi: 10.1103/PhysRevA.101.012320.
- [15]. A. Bärttschi and S. Eidenbenz, “Deterministic preparation of Dicke states,” in *Fundamentals of Computation Theory (FCT)*, LNCS 11651, 2019, pp. 126–139, doi: 10.1007/978-3-030-25027-0_9.
- [16]. L. Zhou, S.-T. Wang, S. Choi, H. Pichler, and M. D. Lukin, “Quantum approximate optimization algorithm: Performance, mechanism, and implementation on near-term devices,” *Phys. Rev. X*, vol. 10, Art. 021067, 2020, doi: 10.1103/PhysRevX.10.021067.
- [17]. K. K. Gajula, “Advancing Post-Quantum Cryptography: Novel Algorithmic Models and Security Implementations,” *Goya Journal*, vol. 18, no. 12, pp. 594–605, 2025.
- [18]. S. Y.-C. Chen and S. Yoo, “Federated quantum machine learning,” *Entropy*, vol. 23, no. 4, Art. 460, 2021, doi: 10.3390/e23040460.
- [19]. . Preskill, “Quantum computing in the NISQ era and beyond,” *Quantum*, vol. 2, p. 79, 2018, doi: 10.22331/q-2018-08-06-79.
- [20]. M. Cerezo *et al.*, “Variational quantum algorithms,” *Nat. Rev. Phys.*, vol. 3, pp. 625–644, 2021, doi: 10.1038/s42254-021-00348-9.
- [21]. M. Wooldridge and N. R. Jennings, “Intelligent agents: Theory and practice,” *Knowl. Eng. Rev.*, vol. 10, no. 2, pp. 115–152, 1995, doi: 10.1017/S0269888900008122.
- [22]. L. Wang *et al.*, “A survey on large language model based autonomous agents,” *Front. Comput. Sci.*, vol. 18, Art. 186345, 2024, doi: 10.1007/s11704-024-40231-1.
- [23]. M. P. Harrigan *et al.*, “Quantum approximate optimization of non-planar graph problems on a planar superconducting processor,” *Nat. Phys.*, vol. 17, pp. 332–336, 2021, doi: 10.1038/s41567-020-01105-y.



- [24]. E. Alpaydin and C. Kaynak, "Optical recognition of handwritten digits," UCI Machine Learning Repository, 1998, doi: 10.24432/C50P49.
- [25]. S. P. Karimireddy, L. He, and M. Jaggi, "Byzantine-robust learning on heterogeneous datasets via bucketing," in *Proc. ICLR*, 2022.
- [26]. T.-M. H. Hsu, H. Qi, and M. Brown, "Measuring the effects of non-identical data distribution for federated visual classification," arXiv:1909.06335, 2019.