Proof of Quality Inference (PoQI): An AI Consensus Protocol for Decentralized DNN Inference Frameworks

Dimitrios Papaioannou
School of Informatics,
Aristotle University of Thessaloniki
Thessaloniki, Greece
dpapaion@csd.auth.gr

Vasileios Mygdalis
Faculty of Business and Economics,
University of Antwerp
Antwerpen, Belgium
Vasileios.Mygdalis@uantwerpen.be

Ioannis Pitas
School of Informatics,
Aristotle University of Thessaloniki
Thessaloniki, Greece
pitas@csd.auth.gr

***Abstract*—In the realm of machine learning systems, achieving consensus among networking nodes is a fundamental yet challenging task. This paper presents Proof of Quality Inference (PoQI), a novel consensus protocol designed to integrate deep learning inference under the basic format of the Practical Byzantine Fault Tolerant (P-BFT) algorithm. PoQI is applied to Deep Neural Networks (DNNs) to infer the quality and authenticity of produced estimations by evaluating the trustworthiness of the DNN node's decisions. In this manner, PoQI enables DNN inference nodes to reach a consensus on a common DNN inference history in a fully decentralized fashion, rather than relying on a centralized inference decision-making process. Through PBFT adoption, our method ensures byzantine fault tolerance, permitting DNN nodes to reach an agreement on inference validity swiftly and efficiently. We demonstrate the efficacy of PoQI through theoretical analysis and empirical evaluations, highlighting its potential to forge trust among unreliable DNN nodes.**

***Index Terms*—Distributed Systems, Distributed Consensus Protocols, BFT-Consensus Protocols, Proof of Work, Decentralized DNN Inference**

I. Introduction

Over the years, Machine Learning has undergone significant evolution, reshaping our understanding of automation across various domains and tasks [1]. Traditional Deep Neural Networks (DNNs) have a centralized structure for data collection and training/testing, enabling the generation of highly accurate and robust DNN inference. However, this centralized approach poses several challenges. As the demand for data increases, concerns about privacy emerge, as sensitive data must be stored and maintained in a single location, raising confidentiality risks [2]. Additionally, a centralized infrastructure represents a single point of failure, rendering the entire system vulnerable to disruptions. In addressing these trade-offs, various distributed or decentralized DNN architectures emerged [3], [4]. In an edge-to-cloud computing enviroments, data typically reside locally on edge devices, while intensive computations are offloaded to the cloud for distributed computing. On the contrary, on-device DNN inference, can be performed by leveraging methods such as computational partitioning [5]–[7]. This entails the decomposition of complex DNN models across disparate computing devices. A notable drawback of such strategies is the necessity for all devices to be concurrently available.

Many distributed or decentralized deep neural network (DNN) methodologies operate under the assumption of reliable communication links among participating nodes, facilitated either through interconnected networks or by presuming the unwavering honesty of nodes irrespective of circumstances. In numerous cases, conventional aggregation techniques such as averaging [8] or simple majority voting [9] are employed to integrate individual DNN node outputs into a cohesive system-wide outcome. However, employing conventional aggregation methods like simple majority voting in decentralized systems with unreliable nodes presents significant challenges [10]. The presumption of consistent node honesty may be unfounded, opening avenues for malicious actors to infiltrate and manipulate transmitted data, compromising the system's integrity. Consequently, the resulting aggregation may be distorted by these faulty nodes, leading to inaccuracies. Moreover, since simple majority voting lacks built-in fault tolerance properties suitable for such environments [11], the aggregated results may not faithfully represent the network's true consensus. Ultimately, relying on majority voting in decentralized systems with unreliable nodes undermines system robustness and jeopardizes decision-making integrity.

Motivated by this, in this paper, we introduce a novel consensus protocol called Proof of Quality Inference (PoQI) to investigate the fusion of the P-BFT algorithm within the concept of decentralized DNN inference tasks.

By requiring DNN nodes to reach a consensus solely on the final layer probability distributions, we significantly reduce the amount of information that needs to be exchanged between them. Consequently, DNN nodes operating in our system can achieve consensus and locally maintain a universally accepted DNN inference history without relying on any centralized DNN aggregation, thus making the system resilient to Byzantine failures. Our experiments on classification tasks demonstrate that PoQI is capable of achieving accurate results comparable to traditional centralized aggregation schemes.

II. Distributed Consensus Protocols

In distributed systems, consensus presents a state in which individual system nodes come to agreement on the same data values [12]. To reach a consensus, individual nodes operating within a network, are required to frequently exchange information between them to prevent any system abuse. That process is neither trivial nor simple, relying on several factors, such as network synchrony and systematic design, among others. It has been proved that its impossible to reach a consensus in an asynchronous network even if only one process fails to respond [13]. In distributed settings, several failures may occur during normal system operation. These include *crash failures*, where a node abruptly stop its execution, *fail-to-stop failures*, where a node fails to halt its execution as expected and *byzantine failures*, where a seemingly operational node transmits arbitrary data to neighboring nodes in an attempt to mislead and compromise the entire process.

A consensus protocol is characterized as Byzantine Fault Tolerant (BFT) [14], [15],if it can tolerate a specific number of f faulty nodes while it keeps normally functioning. A BFT consensus protocol must full fill the following requirements: *termination* (every non-faulty node decides an output), *agreeement* (every non-faulty node eventually decide on the same output y), *validity* (if a non-faulty node receive a value x then, outputs $y = x$) and finally *integrity* (every non-faulty node decision must have been proposed by some other non-faulty node). In order for the BFT requirements to be fulfilled, the total number of nodes N participating in the distributed network must satisfy the condition $N \geq 3f + 1$ [10]. The correctness of a BFT consensus protocol is closely tied to the accuracy of the *safety* and *liveness* properties [16]. Safety states that every node successfully executes the same sequence of requests, while liveness states that all requests should be served.

To ensure that a BFT-consensus protocol meets the aforementioned requirements, it typically adopts a classical State Machine Replication (SMR) architectural design [11]. In this setup, the system state is replicated across a set of N deterministic replicas (e.g., nodes) that are synchronized by a clock. An SMR BFT consensus protocol must guarantee three key properties: a) all nodes start from the same initial state, b) all nodes receive the same sequence of requests generated by other nodes (e.g., atomic broadcast), and c) all nodes receiving the same request produce the same execution result (e.g., uniform output). SMR approaches often limit the number of nodes involved to prevent scalability and latency issues. However, classical SMR approaches face challenges, when applied to DNN training applications due to the message communication complexity, which grows quadratically $O(n^2)$ with the node number n. As a result, employing classical SMR approaches in DNN training is impractical [7]. Instead, it may be worthwhile to explore the potential application of these approaches in scenarios involving pure DNN inference, where computation complexity is limited.

The Practical Byzantine Fault Tolerance (PBFT) [17], has long been regarded as a state-of-the-art consensus protocol for handling Byzantine systems. It can tolerate up to a $1/3$ fraction of Byzantine faults in a system, by employing a leader-based approach leading to a $O(|M|/n^2)$ message complexity. In brief, one of the participating nodes is designated as the leader and is responsible for coordinating the consensus process. If, for any reason, the leader is deemed faulty, the rest of the nodes initiate a process known as a *view change* to declare a new one until a consensus for a specific request is reached. To date, several architectures, such as [18]–[20], have been introduced to enhance the scalability and efficiency of the above protocol. However, none of the literature has explored the possibility of integrating it with deep learning to foster trust within decentralized DNN inference environments.

III. PoQI: Proof of Quality Inference

Let $\mathcal{G} = \{\mathcal{A}, \mathcal{E}\}$ be a graph consisting of N collaborating AI agents described in a set $\mathcal{A} = \{\alpha_1, \alpha_2, ..., \alpha_N\}$, that are employed to perform a DNN inference task, e.g., data classification of the form $\hat{\mathbf{y}} = \mathbf{f}(\mathbf{x}; \boldsymbol{\theta})$ where $\mathbf{x}$ is a data sample and $\boldsymbol{\theta}$ are the DNN parameters. $\mathcal{E}$ is as a set of fixed communication links allowing them to communicate with each other. It is assumed that all nodes have obtained access to the same test sample $\mathbf{x}$, while their goal is to produce a single prediction $\hat{y}$ out of $\hat{y}_{ij}\ \forall\ i\epsilon[0, N]$ and $j\epsilon\mathcal{C}$, where $\mathcal{C}$ is a set of valid classes for the specific data classification task of the form $\mathcal{C} = \{c_1, \ldots, c_c\}$. Furthermore, it is assumed that each DNN node produces a softmax classification inference result so that $\hat{\mathbf{y}}_i = \mathbf{f}_i(\mathbf{x}_i; \boldsymbol{\theta}_i)$ and $\sum_{i=1}^{C} \hat{\mathbf{y}}_i = 1$. For the nodes to coordinate on a single inference prediction, we propose a novel consensus protocol to be formed as a single inference rule and provide coordination in

the individualized nodes decisions under a fully decentralized structure, thus eliminating the need of any kind of centralized coordination. The *Proof of Quality Inference (PoQI)* protocol can be thought of as a new hybrid consensus mechanism, where the core process is achieved by adapting the traditional BFT SMR approach, where $N \leq 2f + 1$ DNN nodes are needed to tolerate f *faulty* nodes who may fail during execution or who may behave maliciously by transmitting tampered data to the neighboring nodes. The normally operating nodes are considered to be the *honest* ones.

The PoQI protocol is designed to tackle Byzantine failures within the context of decentralized inference. In our systematic design, a classical Byzantine failure scenario is assumed, where malicious DNN nodes may attempt to disrupt the entire DNN Inference process by influencing honest DNN nodes with their incorrect inference predictions. Specifically, each DNN node processes the same data sample $\mathbf{x}$, and must collaborate with the other $N-1$ DNN nodes to achieve a consensus regarding both the sample label and the sequential order of DNN classifications. The adversaries within this model are capable of exhibiting arbitrary behavior with the intent of compromising the integrity of the overall DNN Inference process. These adversaries may collude with one another to maximize the extent of their influence and undermine the established protocol. In response to such adversarial behavior, the PoW Byzantine agreement model assigns decision-making authority to the most reliable DNN nodes, enabling them to manage situations where decisions are contested [21].

The system operates within synchronous assumptions, ensuring that the delivery schedule of DNN predictions through messages, is reliably maintained. Efforts are made to minimize communication delays within this framework. Each DNN node is responsible for broadcasting predictions, to all neighboring DNN node, thereby ensuring that every DNN node receives inference predictions in the same order. Consequently, each DNN node maintains a comprehensive record of its own DNN inference prediction history. Thus, PoQI is tasked with ensuring the following properties:

- **Validity:** If an individual honest DNN node i broadcasts a prediction $\hat{y}_i$, then every honest DNN node eventually receives $\hat{y}_i$.
- **Agreement:** If an individual honest DNN node i decides an inference $\hat{y}_i$, then every other honest DNN node must also produce the same inference decision $\hat{y}_i$.
- **Integrity:** A prediction $\hat{y}$ for sample $\mathbf{x}$ appears at most once in the delivery sequence of any honest DNN node.
- **Total Order:** The ordered sequence of predictions $\hat{y}_i$ and $\hat{y}_{i+1}$ for samples $\mathbf{x}_i$, $\mathbf{x}_{i+1}$ must be the same for all honest DNN nodes.

As previously stated, PoQI operates as a state machine replication protocol, consisting of three primary sub-operations: *view change*, *normal operation*, and *conflict decision agreement*. The view change operation orchestrates the primary election process, where a primary DNN node initiates the consensus process, by disseminating its DNN inference prediction regarding the given input sample $\mathbf{x}$ to all other DNN nodes. During normal operation, the core execution of the PoQI protocol takes place, wherein the proposed decision of the primary undergoes evaluation for universal acceptance or rejection. If universally accepted, the primary DNN node is responsible for conveying the network's final decision. Conversely, if the proposed primary decision faces universal rejection, a view-change process ensues. Additionally, in instances where the view change process fails to elect a universally accepted primary node, a conflict decision agreement mechanism is activated. This mechanism is employed to address scenarios arising from inherent characteristics of the DNN models themselves rather than from malicious behavior. In such conflict decision scenarios, the assurance of total ordering during the specific decision period is not guaranteed.

DNN nodes operate through a sequence of actions known as *views* $v \epsilon \mathcal{V}$ where $\mathcal{V} = \{v_1, \ldots, v_k\}$. Given that, PoQI protocol operates in consensus rounds, each defined as one execution of the normal consensus process, regardless if it is successful or not. Views describe the consensus rounds that are required, in order for the DNN network to reach a consensus about the label of a given sample $\mathbf{x}$. It is defined as an index of the form $v \epsilon \mathcal{V}$, containing a sequence of testing pairs whose DNN inference predictions have been scheduled in the time interval t. At each view, one DNN node is operating as *primary* node while the rest $N-1$ nodes are operating as *validators*. In the remainder of this work and for simplicity reasons, each view refers to a single inference prediction of the form $(\mathbf{x}, y)$. Our goal is that every honest DNN node in N maintains an identical DNN inference history set defined as $\hat{\mathcal{Y}} = \{\hat{\mathbf{y}}_{ij},\ \forall i \in \mathcal{V} \text{ and } j \in \mathcal{C}\}$.

1) View Change: **Leader Election.** At any given time, all DNN nodes must be synchronized and begin from the same view. At the beginning, a primary DNN node is selected from the DNN node set $\mathcal{A}$ to begin the consensus process for the first round. From this moment onwards, the rest $N-1$ DNN nodes work as validators. The primary DNN node is elected in circular order, from the first a_1 to the last one a_N, ensuring that every node has a chance to be elected as long as they strictly adhere to the consensus rules. Let $a_p \epsilon \mathcal{A}$ be the primary DNN

node, the election formula is defined as:

$$a_p = v \bmod N, \tag{1}$$

where $v\epsilon\mathcal{V}$ represents the current view we are currently working on.

View Change. Once the primary DNN node of the current view is detected as faulty, view change is performed in order to replace the primary node. Specifically, in the v^{th} view, let the primary node to promote an inference for the i^{th} sample of the form:

$$\hat{y}_p = \text{argmax}\left(\mathbf{f_p}\left(\mathbf{x_i}; \boldsymbol{\theta_p}\right)\right). \tag{2}$$

The primary node communicates its inference prediction $\hat{y}_p$ to the validators by constructing and broadcasting a pre-prepared message of the form $pre-prepare < a_p, \hat{y}_p, v_p, r_p >$ where a_p is the primary node id, $\hat{y}_p \in \mathcal{C}$ is its inference output for the current sample $\mathbf{x}_i$, v_p is the view index and $r_p\epsilon\mathcal{R}$ is the rewards he has collected so far. Set $\mathcal{R}$ is defined as $\mathcal{R} = \{r_1, r_2, \ldots, r_N\}$ and is dynamically updated by the system according to each primary node behavior.

Let $a_j\epsilon\mathcal{A}$ represent a random validator DNN node that has just received the primary's message. He calculates its prediction value as:

$$\hat{y}_j = \text{argmax}\left(\mathbf{f_j}\left(\mathbf{x_i}; \boldsymbol{\theta_j}\right)\right),\ j \neq p. \tag{3}$$

If its inference output differs from the one produced by the primary (e.g., $\hat{y}_j \neq \hat{y}_p$) or if $v_j \neq v_p$ then, from now onwards, the j^{th} DNN node recognizes the primary as a faulty one. In such case, it must immediately multi-cast a view-change message to the rest of the validators of the form $view-change < a_j, v_j + 1, vo_j, r_j >$. The parameter vo_j is equal to 1 if $\hat{y}_j \neq \hat{y}_p$, otherwise $vo_j = 0$. Once the validators receive a view change message, they append it to a local log. If $\frac{\sum_{i=1}^{N-1} vo_i}{N} \geq 0.5$ then the primary node is globally recognized as a faulty one since it has lost majority. To this end, consensus has failed and DNN nodes transition to the new view, initiating the new consensus round.

Reward System. The reward system incentivizes primary DNN nodes to gain the majority vote of validators, by rewarding honest behavior with quality points q and penalizing the loss of majority favor. As a result, each DNN node must locally maintain a record of the rewards acquired by each DNN node so far. For a given primary a_p, the reward and the penalty are calculated as:

$$r_p = \begin{cases} q, & \text{if } \frac{\sum_{i=1}^{N-1} vo_i}{N} < 0.5 \\ 0.5r_p, & \text{if } \frac{\sum_{i=1}^{N-1} vo_i}{N} \geq 0.5 \end{cases}. \tag{4}$$

A substantial amount of the collected primary points are lost if majority support is not attained. The reward policy ensures honest primary DNN node performance. In the opposite case, it may result in reward loss, leading to the identification of potentially faulty or malicious DNN nodes for exclusion from the conflict decision process.

2) Normal Operation: In a normal operation protocol, the determined primary DNN node assesses decisions from validators, selecting relevant ones and constructing a final decision to be multicasted to honest validators. For a primary agent a_p, let $\hat{\mathcal{Y}}_p = \{\hat{\mathbf{y}}_\mathbf{p}\}$ denote the set of the valid collected DNN inference outputs so far. Then the set $\hat{\mathcal{Y}}_p$ is dynamically updated according to his observations as:

$$\hat{\mathcal{Y}}_p = \begin{cases} \hat{\mathcal{Y}}_p \cup \{\hat{\mathbf{y}}_\mathbf{j}\}, & \text{if } \hat{y}_p = \hat{y}_j \\ \hat{\mathcal{Y}}_p, & \text{otherwise} \end{cases}, \tag{5}$$

and the final decision is produced by combining the selected validators decisions using the average or median rule on $\hat{\mathcal{Y}}_p$ entries.

Once the primary has reached to a final decision, he multi-casts to all agents, including himself, an encrypted *prepare* message of the form $prepare < a_p, \hat{y}_p, \hat{y}_{final_p}, v_p, r_p >$ where a_p is the primary DNN node id, $\hat{y}_p$ is its initial predicted value for the current sample, $\hat{y}_{final_p}$ is the final agreed decision, v_p is the view index and r_p is the so far collected rewards. Once a validator receives a prepared message from the primary, it first ensures its validity by observing if the v_p matches its own view number and whether $\hat{y}_p$ matches its own locally produced prediction. If the prepare message is indeed valid, it transmits the prepared message to the rest validators. The validators await for $2f + 1$ identical messages from different agents in order to proceed to the commit phase, where a $commit < a_j, \hat{y}_j, \hat{y}_{final_p}, v_j, r_j >$ is transmitted. Once the validators ensure its validity, it transmits it back to the primary. If the primary receives $2f+1$ identical commit messages from different validators, it recognizes that the consensus is achieved for that specific sample.

3) Conflict Decision Agreement: In this subsection, the inability of DNN nodes operating within the PoQI protocol to classify an input data sample is examined. A data sample in which the majority of DNN nodes cannot reach a consensus on its label is termed a conflict sample. In such cases, DNN nodes are ordered based on the rewards $\mathcal{R}$ they have collected, in descending order, for the conflict sample.

The final decision for each conflict sample is determined using the *Group of Experts Rule*, where DNN nodes are grouped based on their decision similarity, with the most qualified group making the decision. Under this rule, each DNN node operates sequentially, in descending DNN node reward score order. The highest

rewarded DNN node acts as the primary node and produces a prediction of the form $\hat{y}_p$ using (2). Any other DNN node agreeing with this prediction, form a group of experts and jointly propose $\hat{y}_p$, with their rewards being combined as r_g. Any DNN node that has grouped with the primary is excluded from the remaining decision-making process. This process continues until all DNN nodes are grouped in group of experts, each having a decided prediction and total reward reflecting the group's status. The final decision for the i^{th} sample occurs when consensus for the next sample $i+1$ is reached. At this point, let g_i be one of the formed groups. If the primary DNN node for the next sample $a_p \in g_i$ and:

$$\frac{r_{g_i}}{\sum_{i=1}^{N} r_i} \geq 0.51 \tag{6}$$

then the primary node a_p, is responsible to decide for the i^{th} conflict sample as $\hat{y}_p = \text{argmax}\left(\mathbf{f_p}(\mathbf{x_i}; \boldsymbol{\theta_p})\right)$.

If this process fails, the *Most Honest Rule* is applied, where the agent with the highest recorded rewards makes the final decision. Under this rule, if the primary DNN node group fails to meet the 0.51 quality threshold, the decision for the i^{th} conflict sample is made by the agent with the highest reward score (e.g., $a_p = \text{argmax}(r_j)$).

IV. Experiments and Discussion

In our experimental design, we assume a decentralized network comprising several DNN nodes that communicate with each other. Each DNN node contains a pre-trained Convolutional Neural Network (CNN) model, tailored to its unique task or domain. In the experiments, we aim to explore the collective intelligence and collaborative potential of the PoQI consensus protocol. We compare the results with conventional centralized DNN aggregation methods like majority voting and weighted averaging. Lastly, to assess the Byzantine Fault Tolerance (BFT) property of our protocol, we conduct experiments to determine both the maximum number of misbehaving nodes our system can effectively handle and the behavior of majority voting and weighted average in such settings.

We conduct experiments using three benchmark datasets: SVHN [22], CIFAR-10 [23], and F-MNIST [24]. SVHN comprises $26,032$ test images of labeled street view numbers distributed across 10 classes. Both CIFAR-10 and F-MNIST consist of $10,000$ labeled testing images uniformly distributed across 10 classes. Across all datasets, the experiments involve a total of 7 base DNN nodes for Cifar-10 and SVHN and 5 for F-MNIST each equipped with one of the following CNN architectures: VGG11, VGG16, ResNet20, ResNet32, MobileNet v2, ShuffleNet v2, and RepVgg-a1. Unless otherwise specified, all models are pre-trained on ImageNet and fine-tuned on each dataset for a total of 100 epochs to achieve state-of-the-art results.

TABLE I
Accuracy (%) comparison between the PoQI Consensus Protocol and conventional centralized aggregation methods across different datasets, assuming all nodes act honestly, highlighting results obtained from one node.

Model	Dataset		
	F-MNIST	Cifar-10	SVHN
ResNet 20	90.63	92.18	90.90
ResNet 32	90.99	92.65	91.38
VGG 11	87.85	91.53	88.28
VGG 16	90.35	93.68	93.18
MobileNet v2	91.02	92.57	90.83
ShuffleNet v2	-	89.96	89.69
RepVGG	-	94.51	93.56
Weighted Average	**92.51**	95.12	94.09
Majority Voting	92.01	95.05	93.75
PoQI	92.33	**95.27**	**94.12**

Table I presents results from benchmark datasets, aiming to approximate outcomes achieved by centralized methods, under the assumption that all DNN nodes act honestly. Regarding the centralized aggregation methods, we applied them in a straightforward manner directly to each DNN node to measure their collective performance. Specifically, for the weighted average approach, we assigned different weights to each node based on its individual inference performance and then calculated the average of these weighted outputs to obtain the final result, thereby giving more importance to higher-performing nodes. For the majority voting approach, we treated each node's inference output equally and determined the final result based on the most frequently occurring output among the nodes. For SVHN, PoQI slightly outperforms centralized voting rules. CIFAR-10 sees the highest accuracy with the PoQI consensus protocol, surpassing centralized methods. However, on Fashion-MNIST, weight average performs best, followed closely by the PoQI consensus protocol. These findings indicate that our proposed PoQI consensus protocol produces similar results compared to the baseline aggregation methods.

TABLE II
Per DNN node accuracy (%) comparison between the PoQI Consensus Protocol and conventional aggregation methods, assuming a subset of faulty nodes acting arbitrarily

Dataset	Faulty Nodes	Method	Accuracy (%)						
			N1	N2	N3	N4	N5	N6	N7
Cifar-10	0	Weighted Average	95.12	95.12	95.12	95.12	95.12	95.12	95.12
		Majority Voting	95.05	95.05	95.05	95.05	95.05	95.05	95.05
		PoQI	**95.27**	**95.27**	**95.27**	**95.27**	**95.27**	**95.27**	**95.27**
Cifar-10	1	Weighted Average	16.40	15.87	15.92	16.24	15.35	16.11	-
		Majority Voting	94.63	94.86	94.76	**95.02**	94.72	94.56	-
		PoQI	**94.99**	**94.99**	**94.99**	94.99	**94.99**	**94.99**	-
SVHN	1	Weighted Average	15.27	15.41	15.37	15.33	-	15.13	15.52
		Majority Voting	93.21	93.36	93.17	93.12	-	93.04	**93.77**
		PoQI	**93.42**	**93.42**	**93.42**	**93.42**	-	**93.42**	93.42
SVHN	3	Weighted Average	-	11.14	11.40	-	11.16	11.36	-
		Majority Voting	-	92.56	93.12	-	92.94	91.82	-
		PoQI	-	**93.18**	**93.18**	-	**93.18**	**93.18**	-

Furthermore, we aim to determine the fault tolerance property for our protocol and the traditional decision-making methods such as weighted average and majority

voting. For the decentralized DNN Inference framework to be properly functioning and act as a unitary system, requires for every sample, the nodes to agree on the same DNN inference estimation. In our second set of experiments II, we introduce a subset of faulty DNN nodes attempting to disrupt the consensus process by transmitting randomized DNN inference results. In the case of Cifar-10 with 0 faulty agents we can observe that majority voting is stable, producing the same results on every DNN node, thus the system as a whole is in agreement and works as it is supposed to work. However, in the rest of the experiments, we can observe that even with a 1 faulty DNN node that is arbitrarily sending randomized DNN inference results, the weighted average is completely failing while the majority voting is not stable anymore. This means that there is at least one data sample, on which DNN nodes are no longer in agreement and they produce randomized results. This instability on majority voting proves that is not fault-tolerant at all, and even with one faulty agent, it can not be used anymore to establish a commonly accepted agreement on the system. On the other hand, our protocol is able to coordinate the decision-making process of the DNN nodes, even in the present of faulty nodes, since the decision for each sample is performed by the primary DNN node and every honest DNN node must and will comply with his decision. We assume that the faulty nodes are acting completely arbitrarily so there results are not reported.

V. Conclusion

In this paper, the Proof of Quality Inference (PoQI) is introduced. A pioneering consensus protocol, that seamlessly integrates deep learning inference within the Practical Byzantine Fault Tolerant (P-BFT) algorithm framework. By harnessing DNNs to assess the quality and authenticity of inference outcomes, PoQI empowers distributed nodes to achieve consensus on a shared prediction history autonomously, without dependence on centralized infrastructures. Through the adoption of PBFT, PoQI ensures byzantine fault tolerance, facilitating swift and efficient agreement on prediction validity. The theoretical analysis and empirical evaluations presented herein underscore the effectiveness of PoQI in establishing trust among unreliable DNN nodes. This research not only contributes to the burgeoning field of distributed systems but also paves the way for transformative applications in decentralized networks where trust and consensus are paramount. To this end, future research directions could investigate the possibility of integrating the proposed method with other deep learning tasks, such as semantic segmentation and object detection. Additionally, it would be intriguing to explore the potential of incorporating the PoQI protocol within the systematic design of an AI-powered blockchain system.

Acknowledgment

The research leading to these results has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No 951911 (AI4Media). This publication reflects only the authors' views. The European Commission is not responsible for any use that may be made of the information it contains.

References

[1] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[2] Y. Ko, K. Choi, J. Seo, and S.-W. Kim, "An in-depth analysis of distributed training of deep neural networks," in *2021 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*. IEEE, 2021, pp. 994–1003.

[3] G. Liu, F. Dai, X. Xu, X. Fu, W. Dou, N. Kumar, and M. Bilal, "An adaptive dnn inference acceleration framework with end–edge–cloud collaborative computing," *Future Generation Computer Systems*, vol. 140, pp. 422–435, 2023.

[4] D. Rosendo, A. Costan, P. Valduriez, and G. Antoniu, "Distributed intelligence on the edge-to-cloud continuum: A systematic literature review," *Journal of Parallel and Distributed Computing*, vol. 166, pp. 71–94, 2022.

[5] M. Malka, E. Farhan, H. Morgenstern, and N. Shlezinger, "Decentralized low-latency collaborative inference via ensembles on the edge," *arXiv preprint arXiv:2206.03165*, 2022.

[6] Y. Yu, J. Deng, Y. Tang, J. Liu, and W. Chen, "Decentralized ensemble learning based on sample exchange among multiple agents," in *Proceedings of the 2019 ACM International Symposium on Blockchain and Secure Critical Infrastructure*, 2019, pp. 57–66.

[7] N. Shlezinger and Y. C. Eldar, "Deep task-based quantization," *Entropy*, vol. 23, no. 1, p. 104, 2021.

[8] J. Kittler, M. Hatef, R. P. Duin, and J. Matas, "On combining classifiers," *IEEE transactions on pattern analysis and machine intelligence*, vol. 20, no. 3, pp. 226–239, 1998.

[9] L. Lam and S. Suen, "Application of majority voting to pattern recognition: an analysis of its behavior and performance," *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, vol. 27, no. 5, pp. 553–568, 1997.

[10] L. Lamport, R. Shostak, and M. Pease, "The byzantine generals problem," in *Concurrency: the works of leslie lamport*, 2019, pp. 203–226.

[11] A. Bessani, J. Sousa, and E. E. Alchieri, "State machine replication for the masses with bft-smart," in *2014 44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*. IEEE, 2014, pp. 355–362.

[12] Y. Xiao, N. Zhang, W. Lou, and Y. T. Hou, "A survey of distributed consensus protocols for blockchain networks," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 2, pp. 1432–1465, 2020.

[13] M. J. Fischer, N. A. Lynch, and M. S. Paterson, "Impossibility of distributed consensus with one faulty process," *Journal of the ACM (JACM)*, vol. 32, no. 2, pp. 374–382, 1985.

[14] K. P. Kihlstrom, L. E. Moser, and P. M. Melliar-Smith, "The securering protocols for securing group communication," in *Proceedings of the Thirty-First Hawaii International Conference on System Sciences*, vol. 3. IEEE, 1998, pp. 317–326.

[15] D. Malkhi and M. Reiter, "Byzantine quorum systems," *Distributed computing*, vol. 11, no. 4, pp. 203–213, 1998.

[16] V. Gramoli, "From blockchain consensus back to byzantine consensus," *Future Generation Computer Systems*, vol. 107, pp. 760–769, 2020.

[17] M. Castro and B. Liskov, "Practical byzantine fault tolerance," in *Third Symposium on Operating Systems Design and Implementation (OSDI)*. New Orleans, Louisiana: USENIX Association, Co-sponsored by IEEE TCOS and ACM SIGOPS, Feb. 1999.

[18] A. Miller, Y. Xia, K. Croman, E. Shi, and D. Song, "The honey badger of bft protocols," in *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, 2016, pp. 31–42.

[19] A. Spiegelman, N. Giridharan, A. Sonnino, and L. Kokoris-Kogias, "Bullshark: Dag bft protocols made practical," in *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, 2022, pp. 2705–2718.

[20] R. Neiheiser, M. Matos, and L. Rodrigues, "Kauri: Scalable bft consensus with pipelined tree-based dissemination and aggregation," in *Proceedings of the ACM SIGOPS 28th Symposium on Operating Systems Principles*, 2021, pp. 35–48.

[21] S. Nakamoto, "Bitcoin whitepaper," *URL: https://bitcoin. org/bitcoin. pdf-(: 17.07. 2019)*, vol. 9, p. 15, 2008.

[22] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Ng, "Reading digits in natural images with unsupervised feature learning," *NIPS*, 01 2011. [Online]. Available: http://ufldl.stanford.edu/housenumbers

[23] A. Krizhevsky, "Learning multiple layers of features from tiny images," 2009.

[24] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," 2017.