# A Weighted Byzantine Fault Tolerance Consensus Driven Trusted Multiple Large Language Models Network

Haoxiang Luo, *Graduate Student Member, IEEE*, Gang Sun, *Senior Member, IEEE*, Yinqiu Liu, *Member, IEEE*, Dongcheng Zhao, Dusit Niyato, *Fellow, IEEE*, Hongfang Yu, *Senior Member, IEEE*, and Schahram Dustdar, *Fellow, IEEE*

***Abstract*— Large Language Models (LLMs) have achieved remarkable success across a wide range of applications. However, individual LLMs often produce inconsistent, biased, or hallucinated outputs due to limitations in their training corpora and model architectures. Recently, collaborative frameworks such as the Multi-LLM Network (MultiLLMN) have been introduced, enabling multiple LLMs to interact and jointly respond to user queries. Nevertheless, MultiLLMN architectures raise critical concerns regarding the reliability and security of the generated content, particularly in open environments where malicious or compromised LLMs may be present. Moreover, reliance on centralized coordination undermines system efficiency and introduces single points of failure. In this paper, we propose a novel Trusted MultiLLMN framework, driven by a Weighted Byzantine Fault Tolerance (WBFT) blockchain consensus mechanism, to ensure the reliability, security, and efficiency of multi-LLM collaboration. In WBFT, voting weights are adaptively assigned to each LLM based on its response quality and trustworthiness, incentivizing reliable behavior, and reducing the impact of malicious nodes. Extensive simulations demonstrate that WBFT significantly improves both consensus security and efficiency compared to classical and modern consensus mechanisms, particularly under wireless network conditions. Furthermore, our evaluations reveal that Trusted MultiLLMN supported by WBFT can deliver higher-quality and more credible responses than both single LLMs and conventional MultiLLMNs, thereby providing a promising path toward building robust, decentralized AI collaboration networks.**

***Index Terms*— Large language model (LLM), LLM networks, blockchain consensus, trusted LLM, multi-LLM collaboration.**

Received 2 May 2025; revised 9 July 2025 and 26 August 2025; accepted 7 October 2025. Date of publication 13 October 2025; date of current version 31 December 2025. This work was supported by the Major Key Project of PCL under Grant PCL2024A05. The associate editor coordinating the review of this article and approving it for publication was S. Yu. *(Corresponding author: Gang Sun.)*

Haoxiang Luo, Gang Sun, and Hongfang Yu are with the School of Information and Communication Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China (e-mail: lhx991115@163.com; gangsun@uestc.edu.cn; yuhf@uestc.edu.cn).

Yinqiu Liu and Dusit Niyato are with the College of Computing and Data Science, Nanyang Technological University, Singapore 639798 (e-mail: yinqiu001@e.ntu.edu.sg; dniyato@ntu.edu.sg).

Dongcheng Zhao is with the Peng Cheng Laboratory, Shenzhen 518055, China (e-mail: zhaodc11@gmail.com).

Schahram Dustdar is with the Distributed Systems Group, TU Wien, 1040 Vienna, Austria, and also with ICREA, 08002 Barcelona, Spain (e-mail: dustdar@dsg.tuwien.ac.at).

Digital Object Identifier 10.1109/TCCN.2025.3620286

## I. Introduction

LARGE language models (LLMs) have become a cornerstone of AI, showing great capabilities in natural language understanding and generation [1], [2]. These LLMs, represented by ChatGPT, Deepseek, and Claude, have been widely adopted in all aspects of society, such as education, healthcare, and information technology [3], [4], [5].

With the deepening of LLM applications, a large number of LLMs developed by different commercial, academic, or educational organizations have emerged. However, considering the diverse learning corpus, model architectures, and training goals, the answers of different LLMs to the same query will naturally differ [6], [7]. Additionally, a single LLM is difficult to adapt to various application scenarios with heterogeneous contexts and requirements [8], [9]. Some LLMs even suffer from limitations and obsolescence in the training data, resulting in biased content generation and the ability to produce low-confidence outputs or hallucinations [10], [11], [12]. To address such challenges, the collaboration of multiple LLMs is on the agenda. For instance, Wang et al. [13] leveraged GPT-3, GPT-4o, Llama3-8B, Llama3-Chinese, Doubao, SparkDesk, Qwen, and Kimi to jointly provide care programs for the elderly. To accelerate collaboration and communication between multiple LLMs, Marro et al. [14] designed and developed a versatile, efficient, and portable communication protocol for them. This kind of collaborative LLM network is named as Multi-LLMs Network (MultiLLMN).

### A. Research Motivations

Despite realizing inter-LLM collaborations, MultiLLMN causes several security concerns. First, any User Equipment (UE) can access historical queries and answers. Consequently, privacy disclosure [15] and security attacks can frequently occur, especially in applications with private data such as medical and health information. In addition, malicious LLMs in MultiLLMN, such as WormGPT [16], can not only mislead UEs, but also escalate cybersecurity threats, leading to the leakage of UE privacy and business secrets. Even ostensibly honest LLMs may run on compromised devices, resulting in tampered or dishonest responses. It will mislead the final

output of MultiLLMN. Furthermore, in the MultiLLMN architecture, we typically need to employ an authoritative third party to determine the most reliable answer from the responses of multiple LLMs. The introduction of centralized authorities undermines the response efficiency of MultiLLMN and can cause single-point failures, hindering the deployment of MultiLLMN in latency-sensitive scenarios, such as autonomous driving. These concerns pose a significant challenge to the efficient collaboration of MultiLLMN and prevent UE from obtaining credible answers. Consequently, we present the following research questions.

- **Q1**: How can users ensure that they receive the best and trusted response from MultiLLMN in an open network environment and with potentially malicious LLMs?
- **Q2**: How can MultiLLMN eliminate single points of failure and achieve efficient response aggregation through multi-LLM collaboration?

To answer these questions, we leverage blockchain to ensure the reliability and security of information transmission in MultiLLMNs. The combination of the two is an important choice under the demands of intelligence and trustworthiness [17], [18], [19]. Specifically, blockchain consensus mechanisms enable each LLM to make decisions independently of third-party authorities through a decentralized voting process [20], [21]. To address the potential Byzantine behavior of malicious LLMs, we plan to adopt and improve the Byzantine Fault-Tolerant (BFT) consensus. However, these consensus mechanisms ignore the differences in LLM capabilities and give each LLM the same voting weight in the consensus process. Typical examples include Practical Byzantine Fault Tolerance (PBFT) [22] and HotStuff [23], among others. We notice that this assumption deviates from real-world scenarios, where LLMs vary significantly in their response quality, generation capabilities, and susceptibility to malicious behavior. Existing research on blockchain-enabled LLMs has focused mainly on distributed training processes and the traceability of generated content, such as [24], [25], and [26]. This means that the design of a consensus-driven MultiLLMN remains largely unexplored. Consequently, applying equal voting rights across all LLMs often undermines the ability of MultiLLMN to produce the highest-quality and most trustworthy responses for UEs. Hence, the following research question is raised.

- **Q3**: How to assign a fair and reasonable consensus voting weight to each LLM by evaluating its trustworthiness and generation ability comprehensively, to assist MultiLLMN to generate the best quality response?

As a result, it is necessary to design a Weighted Byzantine Fault Tolerance (WBFT) consensus for MultiLLMNs to reduce the voting weight of LLMs with poor response quality, weak generation ability, and possibly malicious behavior, to obtain LLM answers more accurately.

### B. Our Contributions

To the best of our knowledge, this is the first work to present a blockchain consensus for MultiLLMNs. Specifically, the contributions of this paper can be summarized as follows:

- To avoid the limitations of a single LLM, we propose the innovative concept of MultiLLMN. Through inter-LLM collaborations, MultiLLMN mitigates biased responses and hallucinations arising from the deficiencies of individual models.
- To prevent malicious LLM behaviors and enhance the efficiency of MultiLLMNs in serving UEs, we introduce a blockchain consensus mechanism to securely drive inter-LLM collaboration. Additionally, we design an improved clustering-based optimization method that dynamically adapts the MultiLLMN network structure, facilitating the formation of a Trusted MultiLLMN and further strengthening its reliability and effectiveness.
- To facilitate reliable response selection in MultiLLMN, we develop the WBFT consensus mechanism, which assigns fair and adaptive voting weights to each LLM based on their response quality and trustworthiness. This weighting strategy improves the reliability and robustness of the consensus process.
- Through extensive simulations, we demonstrate that WBFT outperforms traditional consensus mechanisms in both security and efficiency. Furthermore, Trusted MultiLLMN generates responses of higher quality and credibility compared to those from a single LLM or a MultiLLMN without consensus participation.

### C. Structure of This Paper

The remainder of this paper is organized as follows. Section II reviews the related work. Section III describes how blockchain drives Trusted MultiLLMN to work. Then, we present the workflow of Trusted MultiLLMN enabled by blockchain in Section III. In Section IV, we demonstrate the design of WBFT. Then, we analyze the security and complexity of WBFT in Section V. Subsequently, Section VI uses an autonomous vehicle scenario as a case study to verify the application potential of our framework. Furthermore, we conduct extensive performance simulations of this framework in VII, which have validated its effectiveness and superiority in serving UE. Finally, we summarize this work in VIII.

## II. Related Work

In this section, we investigate the relevant work on inter-LLM collaboration and blockchain-enabled LLMs. Particularly, we compare existing research with our proposal in TABLE I to illustrate our contributions.

### A. Collaboration Among Multiple LLMs

A single LLM often struggles to comprehensively address real-world requirements, suffering from issues such as biased outputs, low-confidence generations, and hallucinations due to limitations in its training data. To overcome these inherent weaknesses, the inter-LLM collaboration has emerged as a promising direction and attracted great research attention.

For instance, Feng et al. [11] designed a collaboration framework involving three LLMs to mitigate the knowledge gaps of a single LLM caused by outdated or insufficient

TABLE I
SCHEME COMPARISON

| Ref. | Contributions | Possible limitations |
|---|---|---|
| [10] | Design a decentralized and centralized multi-LLM network to avoid the biased generated content by a single LLM | Lack of security and efficiency issues in the multi-LLM interaction |
| [11] | Propose a competitive approach to determine highly reliable responses among multi-LLM in addition to cooperation | Lack of security and efficiency issues in the multi-LLM interaction |
| [13] | Utilize multi-LLM to jointly provide services for elderly care | Lack of security and efficiency issues in the multi-LLM interaction |
| [24] | Explore the technical routes for credibility of LLM learning corpora, training processes, and generated content by applying blockchain | Do not overcome the single LLM limitations, such as generated content bias and hallucinations |
| [27] | Organize multi-LLM into a hierarchical structure based on levels of access and information exchange for efficient collaboration. | Lack of security issues in the multi-LLM interaction |
| [28] | Construct a collaboration method between the general basic model and the dedicated LLM | Lack of security and efficiency issues in the multi-LLM interaction |
| [29] | Enhance the reasoning ability of a single LLM through the knowledge fusion from multi-LLM | Lack of security issues in the multi-LLM interaction |
| [17] | Empower the security of the LLM distributed training process with blockchain | Do not overcome the capacity limitations of a single LLM |
| [30] | Utilize blockchain to achieve unified and secure collaboration among multiple agents in LLM | Do not overcome the capacity limitations of a single LLM |
| [31] | Empower the traceability and immutability of AIGC content with blockchain | Do not overcome the capacity limitations of a single LLM |
| [32] | Develop a reputation system based on blockchain to evaluate the generated content credibility from LLM | Do not overcome the capacity limitations of a single LLM |
| [33] | Utilize blockchain to achieve efficient and trustworthy AIGC services in Metaverse | Do not overcome the capacity limitations of a single LLM |
| Our work | Design the WBFT blockchain consensus-driven MultiLLMN to provide high-quality responses for users | Add additional blockchain costs and interaction delays between LLMs |

training data. Their framework achieved a 19.3% performance improvement across four tasks in different domains. Similarly, in [27], the authors argued that a single LLM cannot adequately represent the real-world data distribution or the diversity of human perspectives, and that this limitation cannot be resolved merely by training more powerful models. Therefore, they proposed a hierarchical multi-LLM collaboration structure spanning API, text, and logical levels. In addition, Owens et al. [10] also investigated persistent bias in LLM outputs, originating from limited training data. Despite advances in bias mitigation techniques, such as data augmentation and fine-tuning, biased output remains a challenge. Consequently, they developed a multi-LLM communication model to reduce output bias. Moreover, Shen et al. [28] introduced Co-LLM, a joint system where a general-purpose LLM invokes domain-specific expert models, achieving superior performance in instruction following, question answering, and reasoning tasks compared to standalone LLMs. Furthermore, knowledge distillation techniques have been employed to continuously transfer and consolidate knowledge from multiple LLMs into a target model, enhancing its generative capacity [29]. A recent practical application of multi-LLM collaboration appears in elderly care [13]. The authors employed eight mainstream LLMs to collectively provide seniors with services such as electronic payments, daily living assistance, recreational support, security alerts, and emotional companionship. This work highlights the strong potential of multi-LLM collaboration in real-world service scenarios.

However, existing work only considers the cooperation of multiple LLMs and cannot guarantee the trustworthiness of LLM-generated content. Moreover, how multiple LLMs can work together to provide the best response needs to be studied.

### B. Blockchain-Enabled LLM

Due to the existence of malicious LLMs and attackers in MultiLLMN, the research on trustworthy LLMs has been put on the agenda. This vision often relies on blockchain technology that integrates security features such as decentralization, traceability, and immutability.

For example, to enrich LLM training datasets, Zuo et al. [17] developed a blockchain-based federated learning framework that enables various private databases to securely contribute training data. To protect LLM training processes from Byzantine behaviors, Chen et al. [30] proposed BlockAgents, which integrates blockchain into LLM training to resist adversarial threats. Their framework reduces the impact of poisoning attacks on model accuracy to less than 3% and the success rate of backdoor attacks to below 5%, demonstrating strong robustness. Additionally, Liu et al. [31] leveraged blockchain to provide trusted endorsement and protection for AI-Generated Content (AIGC) products, offering traceable verification of ownership changes through smart contracts and incentive mechanisms to promote free circulation. Furthermore, Bouchiha et al. [32] developed LLMChain, a blockchain-based reputation system designed to evaluate and monitor LLM behavior, addressing vulnerabilities such as hallucinations, unreliable reasoning, and harmful content generation. Similarly, Lin et al. [33] designed a blockchain smart contract-based verification mechanism to prevent random

TABLE II
KEY NOTATIONS

| Notations | Definitions |
|---|---|
| $A_{i,j}^r$ | The response quality weight assigned by the $i$-th LLM to the $j$-th LLM in the $r$-th round consensus |
| $B_{i,j}^r$ | The trust weight assigned by the $i$-th LLM to the $j$-th LLM in the $r$-th round consensus |
| $D_{i,h}^r$ | The encrypted data of the $h$-th response initiated by the $i$-th LLM in the $r$-th round consensus |
| $L_{i,j}^r$ | The communication latency for the $j$-th LLM to the $i$-th LLM in the $r$-th round consensus |
| $P_{i,h}^r$ | The proof of $D_{i,h}^r$ |
| $Q_{i,j}^r$ | The response quality score of the $i$-th LLM to the $j$-th LLM in the $r$-th round consensus |
| $T_{i,j}^r$ | The trust score of the $i$-th LLM to the $j$-th LLM in the $r$-th round consensus |
| $V_{i,j}^r$ | The feature vectors of the $j$-th LLM for the $i$-th LLM in the $r$-th round consensus |
| $W_{i,j}^r$ | The weight assigned by the $i$-th LLM to the $j$-th LLM in the $r$-th round consensus |
| $b_{i,h}^r$ | The block added to the chain for the $h$-th response initiated by the $i$-th LLM in the $r$-th round consensus |
| $c_{j,h}^r$ | The confirmation message of the $j$-th LLM for the leader in the commit phase |
| $v_{j,h}^r$ | The voting value of the $j$-th LLM for $D_{i,h}^r$ |

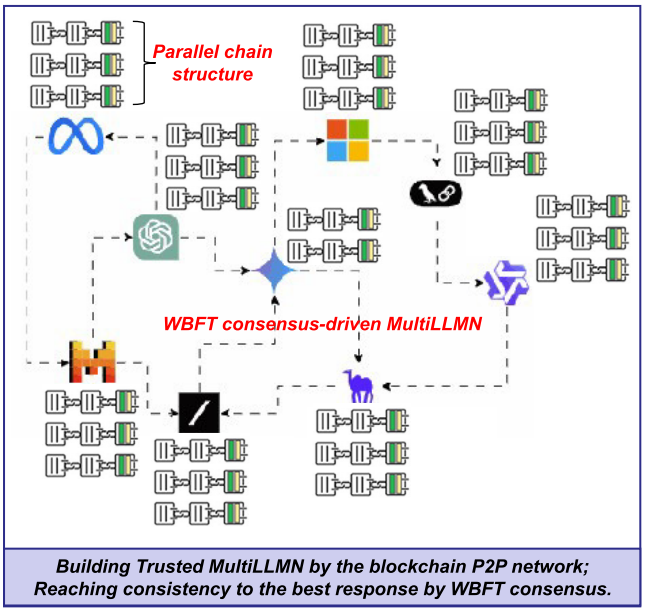



Fig. 1. The blockchain-driven Trusted MultiLLMN. This network is built on a blockchain P2P network and has a decentralized architecture. The best response for UE is derived from the WBFT consensus. The consensus results are packaged into blocks and linked to the chains maintained by distributed LLMs.

outcomes in AIGC services, thus improving service reliability in the Metaverse. Luo et al. [24] systematically analyzed LLM trustworthiness from three key perspectives (i.e., learning corpus, training processes, and generated content) and emphasized the critical role of blockchain technologies across these dimensions.

Existing work has focused primarily on ensuring the trustworthiness of individual LLMs. However, research on the trustworthiness of multi-LLM collaboration remains limited. Ensuring trust in a collaborative environment is inherently more challenging, as it should account not only for the open network environment but also for malicious behaviors induced by compromised or adversarial LLMs.

## III. BLOCKCHAIN-DRIVEN MULTIPLE-LLM NETWORK

In this section, we introduce the architecture of Trusted MultiLLMN driven by blockchains. The commonly used notations are summarized in TABLE II.

### A. Workflow of Trusted MultiLLMN

Regardless of their specific architectures, all LLMs possess the ability to generate content and assess the rationality of generated outputs. This capability serves as the foundation for LLMs to participate in blockchain consensus and to evaluate and vote on the content quality generated by other LLMs. Therefore, we regard each LLM as one full node within the blockchain-driven network, participating directly in consensus processes and contributing to the validation of information exchanges. However, malicious LLMs may produce misleading content and provide dishonest evaluations of other LLMs' outputs, introducing vulnerabilities into the MultiLLMN. This challenge will be addressed through the implementation of our proposed consensus mechanism, which is detailed in the subsequent sections.

Building upon mainstream LLMs, including Llama, WizardLM, GPT, and Gemini, we develop a blockchain-driven MultiLLMN, as illustrated in Fig. 1. This network architecture not only enables collaborative interactions among heterogeneous LLMs but also inherently enhances credibility and trustworthiness through blockchain integration. The operational workflow is outlined as follows.

*1) UE Requests:* The UE submits a response request to Trusted MultiLLMN.

*2) LLM Answer Generation:* Each LLM generates a response to UE requests. These responses are subsequently disseminated among LLMs via broadcast protocols within the blockchain's Peer-to-Peer (P2P) network infrastructure.

*3) Blockchain Consensus:* Consensus serves as a key technique in identifying the optimal response among multiple LLMs. In the proposed framework, a voting consensus mechanism is employed to evaluate and select the final proposal, which is then relayed back to the UE requirements by designated consensus leaders. The proposal is automatically determined by the WBFT consensus, which will be described in the next section.

*4) Block Package:* The optimal response, as determined by WBFT, is encapsulated within a block, which undergoes confirmation by each LLM. To guarantee the security, immutability, and traceability of the consensus outcomes, the block is meticulously designed to incorporate the hash value of the optimal solution alongside its precise timestamp.

*5) Blockchain Extension:* Blocks are correlated with respective parallel chains and stored in a distributed manner across smart devices. Specifically, each LLM extends and maintains the chain that incorporates its own responses, thereby

enhancing the processing efficacy of the Trusted MultiLLMN. Note that each chain is documented by the host device of every LLM to preserve the decentralized character of the blockchain system.

Through the above five steps, the Trusted Multi-LLMN can be established, which delivers responses with blockchain-guaranteed security and reliability.

### B. Dynamic Optimization of Trusted MultiLLMN

While the workflow of Trusted MultiLLMN ensures basic functionality, further optimization is needed to enhance response quality for UEs and improve the efficiency of blockchain consensus. To this end, we introduce a dynamic networking mechanism for Trusted MultiLLMN based on a clustering algorithm, named Hierarchical Secure Clustering (HSC). This approach not only adapts to variations in the network environment and LLM states but also significantly improves the scalability of Trusted MultiLLMN, enabling it to accommodate a larger number of LLMs.

HSC strategically categorizes all LLM nodes into two types: Core Cluster Nodes (CCNs) and Edge Cluster Nodes (ECNs). Only CCNs actively participate in the WBFT consensus mechanism, while ECNs receive consensus results through broadcast messages from their cluster's designated CCN. This hierarchical structure significantly reduces consensus latency and enhances the parallel distribution of consensus outcomes, thereby optimizing the Trusted MultiLLMN's response efficiency to UEs' requests.

For effective deployment in wireless networks, our Trusted MultiLLMN framework requires optimization to accommodate diverse operational conditions. Considering the time-varying characteristics of the wireless channel and the fluctuation of communication latency, the HSC algorithm introduces trust values and latency parameters in the feature vector. This approach effectively mitigates Byzantine LLMs' interference, ensuring consistency within the Trusted MultiLLMN, as represented by

$$V_{i,j}^r = [\omega T_{i,j}^r, (1-\omega) L_{i,j}^r]. \tag{1}$$

$V_{i,j}^r$ represents the feature vectors of the $j$-th LLM from the view of the $i$-th LLM in the $r$-th round of consensus. $T_{i,j}^r$ denotes the trust value assigned by $i$-th LLM to $j$-th LLM. $L_{i,j}^r$ denotes communication latency from $j$-th to $i$-th LLM. Finally, $\omega$ serves as the proportional regulator.

Then, the K-means++ algorithm [34] is applied to determine the number of clusters $K$. Specifically, we add a penalty term $\lambda$ for dynamically controlling the number of clusters based on minimizing the sum of squares of the distance between the feature vectors of the nodes and the cluster center $\mu_k$ (i.e., the tightness of the cluster), i.e.,

$$\min \sum_{k=1}^{K} \sum_{V_{i,j}^r \in C_k} \|V_{i,j}^r - \mu_k\|^2 + \lambda. \tag{2}$$

$C_k$ represents the $k$-th cluster divided by HSC. This equation can be solved using the Elbow method [35]. Next, we iteratively update the cluster centers until we identify exactly $K$ CCNs, with each CCN responsible for maintaining one distinct cluster, as expressed by:

$$\mu_k^{(t+1)} = \frac{\sum_{j \in C_k} \left(\frac{T_{i,j}^r}{L_{i,j}^r}\right)^{\gamma} \cdot V_{i,j}^r}{\sum_{j \in C_k} \left(\frac{T_{i,j}^r}{L_{i,j}^r}\right)^{\gamma}}, \tag{3}$$

where $\gamma$ is a nonlinear regulatory factor. This equation allows HSC to select LLM nodes with high reputation and low latency as CCNs to optimize the cluster structure of MultiLLMN. Furthermore, HSC can dynamically adapt to changing conditions in each consensus round for two key reasons. First, Byzantine LLMs exhibit inconsistent malicious behaviors, causing their trust values $T_{i,j}^r$ to fluctuate between rounds. Second, the inherent instability of wireless environments leads to temporal variations in communication latency $L_{i,j}^r$. These dynamic factors require HSC to continuously adjust cluster formations to maintain optimal performance and security throughout the consensus process.

### C. Threat Model

In the blockchain-driven MultiLLMN, there are both benign and adversarial players:

- **Benign Players:** They are honest LLM nodes, that adhere to the protocol, faithfully generate the best possible responses to user queries. Meanwhile, they cast honest votes on other nodes' responses based on their judgment.
- **Adversarial Players:** These nodes may be fully controlled by an external attacker or be maliciously designed (e.g., WormGPT). They are not bound by the protocol and can execute arbitrary actions to disrupt the system.

Adversarial players may exhibit one or more of the following Byzantine behaviors:

- **Publishing Malicious Content:** Generating and broadcasting incorrect, biased, harmful, or completely hallucinated responses in an attempt to mislead end-users or poison the consensus outcome.
- **Malicious Voting:** During the WBFT consensus process, Byzantine nodes may engage in dishonest voting. For instance, they may vote against high-quality proposals while voting in favor of low-quality and malicious proposals. They may also selectively send contradictory voting information to different nodes in an attempt to undermine the achievement of consensus.
- **Collusion Attack:** Multiple Byzantine LLM nodes conspire to act in concert. For example, they could collectively vote for a low-quality or malicious proposal to overwhelm honest nodes and manipulate the consensus outcome.
- **Selective Non-Participation:** The Byzantine nodes may deliberately fail to respond or vote during critical consensus rounds, attempting to delay the decision-making process and undermining the MultiLLMN liveness.

In response to the aforementioned threats, our Trusted MultiLLMN framework aims to achieve the following core security goals:

- **Correctness:** To ensure that all honest LLM nodes eventually agree on the same response, preventing forks or inconsistent outcomes.
- **Liveness:** To guarantee that the system can eventually reach a consensus and respond to user requests without being indefinitely stalled by malicious nodes.
- **Integrity of Response:** To ensure that the final response delivered to the user is the one validated by the consensus and has not been tampered with during transmission.
- **Trustworthiness:** As the central goal of the system, the WBFT consensus enables the network to effectively identify and down-weight malicious or low-quality nodes, ensuring that the final selected response is the most reliable and of the highest quality, thereby resisting the aforementioned attacks.

## IV. Weighted Byzantine Fault Tolerance Consensus

In this section, we illustrate the design of WBFT, including the weight allocation scheme and the consensus process.

### A. Weight Allocation Scheme

In WBFT, the weight assigned to each LLM comprises two components: response quality, which reflects content generation capability, and trustworthiness, which indicates whether the LLM and its associated device exhibit malicious behavior. Any LLM can serve as a consensus initiator and thus assign variable weights to peer LLMs. Formally, we denote the weight assigned by the $i$-th LLM to the $j$-th LLM during the $r$-th round of consensus by $W_{i,j}^r$, which is defined as

$$\begin{aligned} &W_{i,j}^r = \alpha A_{i,j}^r + \beta B_{i,j}^r, \\ &\alpha + \beta = 1, \\ &\sum_{j=1}^{n} W_{i,j}^r = \sum_{j=1}^{n} A_{i,j}^r = \sum_{j=1}^{n} B_{i,j}^r = 1, \end{aligned} \tag{4}$$

where $A_{i,j}^r$ and $B_{i,j}^r$ denote the response quality and trust weights assigned by the $i$-th LLM to the $j$-th LLM in the $r$-th round consensus, respectively. The parameters $\alpha$ and $\beta$ determine the relative importance of these two weight components and can be adjusted according to specific requirements. $n$ denotes the total number of LLMs participating in the Trusted MultiLLMN.

In each round of consensus, the LLM receives both responses to UE queries and voting information from other LLMs. Consequently, at the beginning of each consensus round, the consensus initiator has the prerogative to dynamically re-calibrate the response quality and trust weights of other LLMs according to their performance in the last round. Furthermore, weights $A_{i,j}^r$ and $B_{i,j}^r$ can be determined by the dual criteria of response quality and trustworthiness exhibited by all other LLMs, namely

$$\begin{aligned} A_{i,j}^r &= \frac{Q_{i,j}^r}{\sum_{j=1}^{n} Q_{i,j}^r}, \\ B_{i,j}^r &= \frac{T_{i,j}^r}{\sum_{j=1}^{n} T_{i,j}^r}, \end{aligned} \tag{5}$$

where $Q_{i,j}^r$ and $T_{i,j}^r$ represent the answer quality and trustworthiness of the $i$-th LLM to the $j$-th LLM in the $r$-th round consensus, respectively. This setting, combined with the filtering function of the HSC algorithm for low-trust LLMs, can ensure that LLMs can evaluate each other's generated content during the WBFT consensus process, thereby facilitating the smooth progress of the consensus.

### B. Consensus Design

In a MultiLLMN, UEs can access the LLM functionality through any participant. We suppose that the LLM receiving UE's requests serves as the consensus leader, while other LLMs act as followers. Notably, due to the design of HSC, only $K$ CCNs participate in the consensus process, meaning all participating follower LLMs are CCNs.

To ensure secure query-response transmission within the MultiLLMN, we pre-assign two public and private key pairs to each LLM since each can function as either a consensus leader or follower. Specifically, these pairs are $(PK_i^L, SK_i^L)$ and $(PK_i^F, SK_i^F)$. The first pair is used exclusively when LLM $i$ serves as a leader, while the second pair is employed when it acts as a follower. Each LLM's public keys are shared network-wide for encrypting query-response data, while private keys remain secured for decryption operations.

As illustrated in Fig. 2, WBFT consensus comprises two voting rounds, namely the *prepare* phase (as shown in Algorithm 1) and the *commit* phase (as shown in Algorithm 2). Below, we elaborate on the process of each phase.

- ***Prepare Phase:*** The consensus leader initially encrypts the UE's query $Q$ and its own generated response $R_L$ using the public key $PK_j^F$, subsequently broadcasting this encrypted data, denoted as $D_{i,h}^r$, to $n-1$ follower LLMs. Here, $h$ denotes the total number of responses initiated by the leader and concurrently represents the block height within the chain maintained by said leader. Each follower then employs their pre-assigned private key $SK_j^F$ to decrypt the received data and validate its authenticity. Subsequently, each follower transmits a vote, $v_{j,h}^r$, back to the leader. Each vote comprises the vote outcome and the follower's own response $R_F$ to the UE's query $Q$ and is encrypted by the leader's public key $PK_i^L$. The voting mechanism dictates that a follower assigns a vote value of 0 if its generated response exceeds the leader's in terms of quality, and 1 otherwise.
  The leader decrypts the voting information using the follower's public key $PK_j^F$ and tallies the voting values. When the cumulative weight of votes with value 1 exceeds two-thirds (including the leader's own vote weight), the leader generates a proof $P_{i,h}^r$ attesting that $D_{i,h}^r$ has been successfully verified. This proof is implemented as a threshold signature [23] of the aggregated votes.
- ***Commit Phase:*** The consensus leader broadcasts proof $P_{i,h}^r$ to all followers, signaling that the response has been verified as optimal. Each follower authenticates this proof using threshold signature techniques to verify both the proof itself and the leader's non-Byzantine

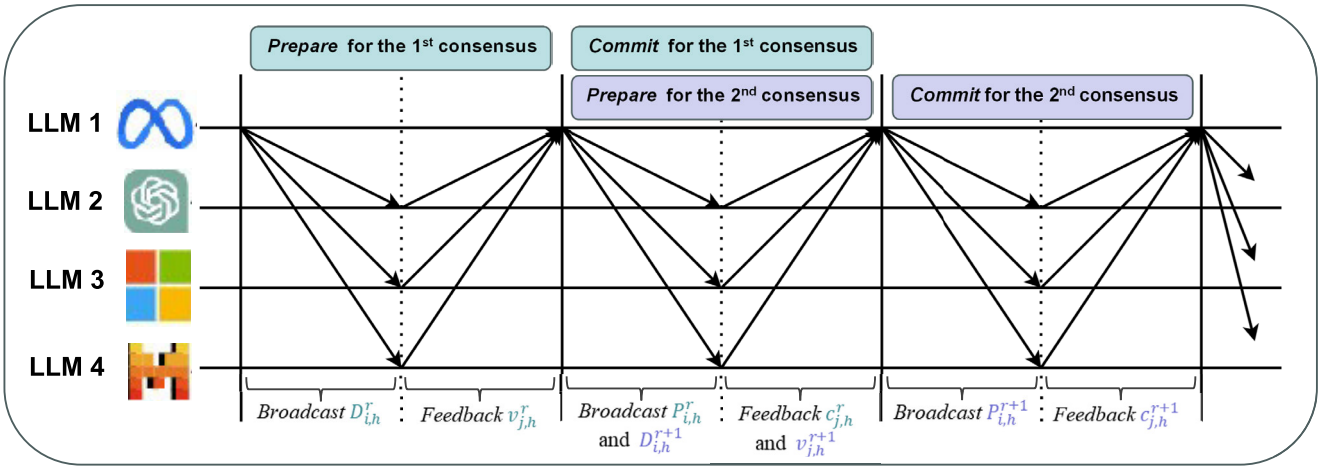



Fig. 2. The consensus process of Weighted Byzantine Fault Tolerance (WBFT) with the pipeline mechanism. It allows the *prepare* phase of the $(r+1)$-th round consensus to be initiated during the *commit* phase of the $r$-th round consensus.

**Algorithm 1** Prepare Phase

Leader Operations:
1. Generate response: $R_L \leftarrow$ generate_response$(Q)$
2. Encrypt data: $D^r_{i,h} \leftarrow$ encrypt$(Q \parallel R_L, PK^F_j)$ ;
`// h = current block height`
3. Broadcast $D^r_{i,h}$ to all followers

Follower Operations:
1. Decrypt data: $(Q, R_L) \leftarrow$ decrypt$(D^r_{i,h}, SK^F_j)$
2. Verify authenticity of $(Q, R_L)$
3. **if** *verification passes* **then**
   Generate response: $R_F \leftarrow$ generate_response$(Q)$
   4. **if** *quality*$(R_F) >$ *quality*$(R_L)$ **then**
      $v^r_{j,h} \leftarrow 0$ ; `// Follower's response is better`
   **else**
      $v^r_{j,h} \leftarrow 1$ ; `// Leader's response is better`
   5. Send encrypt$(v^r_{j,h} \parallel R_F, PK^L_i)$ to leader
**else**
   Discard $D^r_{i,h}$ ; `// Invalid data detected`

Leader Operations:
1. Receive encrypted votes from followers
2. **if** $\sum$ *weight*$(v^r_{j,h} = 1)$ + *leader_self_weight* $> 2/3$ **then**
   Generate proof: $P^r_{i,h} \leftarrow$ generate_proof$(D^r_{i,h})$
**else**
   Consensus termination.

**Algorithm 2** Commit Phase

Leader Operations:
1. Broadcast $P^r_{i,h}$ to all followers

Follower Operations:
1. Decrypt $P^r_{i,h}$ by threshold signature
2. Verify the Byzantine character of the leader
3. **if** *verification passes* **then**
   expunge $P^{(r-1)}_{i,h}$
   Transmit their confirmation messages back to the leader
**else**
   Discard $P^r_{i,h}$ ; `// Invalid data detected`

Leader Operations:
1. Receive confirmation messages from followers
2. **if** $\sum$ *weight*$(c^r_{j,h} = 1)$ + *leader_self_weight* $> 2/3$ **then**
   Reach consistency: $R_L$ with its hash value and timestamp packaged as $b^r_{i,h}$ is appended to the chain.
**else**
   Consensus termination.

status. Upon successful validation, followers remove the previous proof $P^r_{i,h-1}$ to reduce storage requirements, then transmit confirmation messages $c^r_{j,h}$ to the leader. Once the leader receives confirmations representing at least a two-thirds weighted majority, the response $R_L$ with its hash value and timestamp is encapsulated in a block $b^r_{i,h}$ and appended to the corresponding chain.

In contrast, if the leader is detected exhibiting malicious behavior, the MultiLLMN system recommends that the UE obtain responses from LLMs on different devices to avoid potential misinformation from the compromised leader.

Fig. 2 illustrates the overall consensus process. After completing these two rounds, the leader can evaluate the followers' response quality and trustworthiness based on the analysis of generated answers and voting feedback. This evaluation enables dynamic recalibration of voting weights for subsequent consensus rounds, highlighting a key advantage of LLMs, i.e., their ability to exercise independent judgment within the consensus mechanism.

Observing that our dual-round voting consensus mechanism naturally segments into distinct phases, we leverage this structural feature to implement a pipelined processing framework [36], [37] to further improve efficiency. As shown in Fig. 2, while the commit phase for block $b^r_{i,h}$ is being executed, the prepare phase for the subsequent block $b^r_{i,h'}$ can be initiated concurrently, where $h'$ denotes the next block height in the blockchain managed by the $j$-th LLM. This pipeline architecture significantly reduces the waiting time between consecutive requests.

The optimal responses from independent consensus processes initiated by different leaders are assembled into separate blocks and added to chains maintained by their respective leaders, creating a parallel chain architecture. Although each LLM primarily manages its own chain, it also maintains copies of all other chains to ensure system-wide consistency and enable distributed storage of query-response across the MultiLLMN.

## V. Performance Analysis

In this section, we analyze the consensus security, efficiency, and complexity of WBFT.

### A. Consensus Security

Without loss of generality, we measure consensus security by the consensus success rate, as discussed in [38] and [39]. Specifically, we define $v^r_{j,h} = 1$ to indicate that a follower believes it cannot generate a response superior to that of the leader, whereas $v^r_{j,h} = 0$ implies the follower considers its own response to be of higher quality. Thus, successful consensus requires the following condition, denoted as Event $Y$:

$$Y = \sum_{j=1}^{K} W^r_{i,j} v^r_{j,h} > \frac{2}{3}. \tag{6}$$

Event $Y$ involves the summation of multiple products of random variables, making its direct computation highly complex. To address this challenge, we adopt two approaches to approximate the probability of Event $Y$. First, the Monte Carlo method provides a heuristic estimation based on random sampling. Second, when the number of LLMs, denoted by $n$, is sufficiently large, the Law of Large Numbers [40] implies that the distribution of assigned LLM weights can be approximated by a normal distribution over the bounded interval $(0, 1)$. Consequently, the weights $W^r_{i,j}$ are assumed to follow a distribution $N_1\left(\mu, \sigma^2\right)$. Meanwhile, the voting outcomes $v^r_{j,h}$ adhere to a Bernoulli distribution. Since the assigned weights and the voting outcomes are independent, $W^r_{i,j}$ and $v^r_{j,h}$ are mutually independent random variables. By invoking the properties of linear combinations of independent random variables, the expected value and variance of Event $Y$ can be expressed as:

$$\begin{aligned} E(Y) &= E\left(\sum_{j=1}^{n} W^r_{i,j} v^r_{j,h}\right) = \sum_{j=1}^{n} E\left(W^r_{i,j} v^r_{j,h}\right) \\ &= \sum_{j=1}^{n} E\left(W^r_{i,j}\right) E\left(v^r_{j,h}\right) = \sum_{j=1}^{n} \mu p = p, \end{aligned} \tag{7}$$

$$\begin{aligned} D(Y) &= D\left(\sum_{j=1}^{n} W^r_{i,j} v^r_{j,h}\right) = \sum_{j=1}^{n} D\left(W^r_{i,j} v^r_{j,h}\right) \\ &= \sum_{j=1}^{n} \left[E\left(W^2_{i,j}\right) D\left(v^r_{j,h}\right) + E^2\left(W^r_{i,j}\right) D\left(v^r_{j,h}\right)\right] \\ &= \sum_{j=1}^{n} \left[p\sigma^2 + \mu^2 p(1-p)\right] = np\sigma^2 + \frac{1}{n} p(1-p), \end{aligned} \tag{8}$$

where $p$ represents the probability of $v^r_{j,h} = 1$.

Following the Central Limit Theorem [41], when the number of LLMs $n$ becomes sufficiently large, linear combinations of independent and identically distributed random variables approach a normal distribution. Consequently, the probability of Event $Y$ approximates a normal distribution with parameters $N_2\left(p, np\sigma^2 + \frac{1}{n}p(1-p)\right)$. Therefore, the consensus security of WBFT, i.e., $P(Y > 2/3)$ can be expressed as

$$\begin{aligned} p_{\text{WBFT}} &= 1 - P(Y \le 2/3) \\ &= 1 - \Phi\left(\frac{2/3 - p}{\sqrt{np\sigma^2 + \frac{1}{n}p(1-p)}}\right), \end{aligned} \tag{9}$$

where $\Phi$ denotes the normal distribution $N(0, 1)$. By restricting the voting rights of LLMs with low credibility and weak generation capabilities, WBFT will achieve higher consensus security.

### B. Consensus Efficiency

We consider two metrics to reflect consensus efficiency, namely throughput and latency.

To facilitate the adaptable deployment and utilization of MultiLLMN, thereby augmenting its scope of application, we implement the proposed WBFT-driven Trusted MultiLLMN within wireless network contexts. According to [42], [43], [44], and [45], we can conclude that in wireless networks, the consensus latency is intrinsically linked to the consensus success rate, namely

$$1 - P_l = f_Q\left(\frac{NTBC - NTBR + \frac{\log NTB}{2}}{(\log e)\sqrt{NTB}}\right), \tag{10}$$

where $P_l$ denotes the channel transmission success rate. Consequently, $1 - P_l$ means the transmission failure rate associated with this channel. This failure probability can be generalized to the scenario where $v^r_{j,k} = 0$, as it also implies that the leader has not received the follower's endorsement of its response. Furthermore, $f_Q$ denotes the Q-function. In this context, $T$ and $N$ represent the latency of a channel and the number of subcarriers, respectively, with $N$ set to 1 in this paper. Additionally, $B$, $R$, and $C$ represent bandwidth, transmission rate, and channel capacity, respectively.

Both the *prepare* and *commit* phases of the WBFT consensus involve two communication directions: *downlink* transmissions, where the leader broadcasts messages, and *uplink* transmissions, where followers send feedback. The communication structure of each phase is consistent with that of the Raft consensus algorithm [46], [47], [48]. Accordingly,

the latency associated with the WBFT consensus can be expressed as

$$t_c = 2nT. \tag{11}$$

For consensus throughput, it can be expressed as the number of UE requests processed by MultiLLMN per unit of time, whose unit is Transactions per Second (TPS). It can be said that, benefiting from the parallel pipeline processing of UE responses, WBFT has a higher throughput than traditional consensus.

### C. Correctness and Liveness Analysis

WBFT adopts a structure similar to PBFT, ensuring that as long as the number of faulty nodes does not exceed one-third of the total, the consensus satisfies three core properties: non-forking, consistency, and liveness. Assuming a total of $n = 3f + 1$ nodes, with at most $f$ Byzantine nodes, the corresponding proofs are presented as follows.

*Property 1 (Nonforking):* For any two valid committed blocks $b^{r_1}_{i_1,h_1}$ and $b^{r_2}_{i_2,h_2}$, if $i_1 = i_2$ then $h_1 = h_2$, where $b^r_{i,h}$ denotes the $h$-th block generated by $i$-th LLM in the $r$-th consensus round.

*Proof:* Suppose contradictory blocks $b^r_{i_1,h_1}$ and $b^r_{i_2,h_2}$ ($h_1 \neq h_2$) both receive $(2f + 1)/n$ vote weights. With $n = 3f + 1$, we can acquire

$$2 \cdot (2f + 1)/n - (3f + 1)/n = (f + 1)/n. \tag{12}$$

It implies $(f + 1)/n$ vote weights for both blocks, violating the protocol rules. Hence, $h_1$ should equal $h_2$. ■

*Property 2 (Consistency):* If the block $b^r_{i,h}$ is committed by any honest node, all honest nodes will eventually commit it.

*Proof:* Commitment requires proof $P^r_{i,h+1}$ to obtain $(2f + 1)/n$ vote weights. Let $VW_{\mathsf{received}}$ denote the received affirmative vote weights ($VW_{\mathsf{received}} \geq (2f + 1)/n$). At least $(2f + 1)/n - f/n = (f + 1)/n$ votes come from $VW_{\mathsf{received}}$. Since weights of Byzantine nodes $\leq f$, majority voting can ensure eventual global consistency. ■

*Theorem 1:* For any response $R_L$ submitted by the honest nodes, there exists a finite time $T_{\mathsf{finite}}$ such that $R_L$ will be included in block $b^r_{i,j}$ generated by an honest leader.

*Proof:* The liveness proof requires analyzing two mutually exclusive scenarios:

*Case 1. Transaction Recovery Protocol:* Suppose that leader $L^r_i$ proposes block $b^r_{i,h}$ containing $R_L$ with the following conditions:

1) The follower node $F^r_j$ receives $b^r_{i,b}$ but lacks $R_L$.
2) $F^r_j$ extracts the $R_L$ hash value $h(.)$ from $b^r_{i,h}$.
3) By broadcasting $h(R_L)$, $F^r_j$ requests $R_L$ from neighbors via $\mathsf{GetData}(h(R_L))$ messages.
4) The honest nodes that receive $R_L$ respond with $\mathsf{TxResponse}(R_L)$, validated through:

$$\mathsf{VerifyTx}(h(R_L), \mathsf{sig}_{\mathsf{tx}}) = \mathsf{True}. \tag{13}$$

5) After receiving $R_L$, $F^r_j$ executes:

$$\mathsf{Vote}(b^r_{i,h}) \iff \mathsf{ValidateBlock}(b^r_{i,h}) = \mathsf{True}. \tag{14}$$

This process guarantees that all honest nodes can reconstruct $b^r_{i,h}$ in a bounded time $\Delta_1$.

*Case 2. Voting Retry Protocol:* If leader $L^r_i$ fails to receive more than $2/3$ of the vote weights in favor within the time window $T_{\mathsf{init}}$ due to network delays:

1) When the network jitter, $VW_{\mathsf{received}} < (2f + 1)/n$.
2) Leader triggers timeout event:

$$T_{\mathsf{retry}} = T_{\mathsf{init}} + \delta_{\mathsf{backoff}}, \tag{15}$$

where $\delta_{\mathsf{backoff}}$ follows exponential increasing sequence.
3) At $T_{\mathsf{finite}} = T_{\mathsf{retry}}$, leader reinitializes voting with new timestamp:

$$b^r_{i,h'} \leftarrow \mathsf{Repropose}(b^r_{i,h}, \mathsf{ts}_{\mathsf{new}}). \tag{16}$$

4) The process repeats until:

$$\exists\, T_{\mathsf{finite}} \leq T_{\mathsf{max}}, \quad VW_{\mathsf{committed}} \geq (2f + 1)/n, \tag{17}$$

where $VW_{\mathsf{committed}}$ represents the affirmative vote weights of *commit* phase.

Therefore, the probability of success after $k$ times of attempts can be expressed as

$$p_{\mathsf{success}}(k) = 1 - \left(\frac{f}{3f + 1}\right)^k \xrightarrow{k\to\infty} 1. \tag{18}$$

■

*Corollary 1:* The proposed WBFT achieves liveness with probability 1 under partial synchrony assumptions, where message delays are bounded by known time $\Delta_{\mathsf{network}}$.

### D. Consensus Complexity

Compared to the communication complexity of $O(n^2)$ exhibited by PBFT [22], the proposed WBFT significantly reduces the communication complexity to $O(K)$. This improvement is achieved by restricting communication to interactions solely between the leader and the followers, thereby avoiding the extensive inter-follower consultations required during the two-round voting process [49]. In PBFT, follower-to-follower negotiations typically involve repeated flooding broadcasts, which substantially increase communication overhead.

Furthermore, WBFT maintains an advantage even over the $O(n)$ complexity of other parallel multi-chain consensus mechanisms, such as Vote as Proof (VaaP) [50], [51]. Specifically, WBFT not only narrows the scope of consensus participation through the HSC algorithm but also restricts the voting influence of malicious LLMs. By reducing the likelihood of leader re-elections, caused by difficulties in securing the required two-thirds voting weight, WBFT further lowers the overall consensus complexity.

## VI. Use Case: Collaborative Decision for Autonomous Vehicles

To concretely illustrate the necessity of the "blockchain and MultiLLMN" architecture, we present a use case with extremely high requirements for safety, reliability, and real-time performance: collaborative decision-making for a fleet of autonomous vehicles during a complex, unexpected incident.

### A. Scenario Description

Consider a fleet of autonomous vehicles from different manufacturers, e.g., Tesla, Waymo, AITO, driving through a city intersection or on a highway. Suddenly, a multi-car pile-up occurs ahead, causing some vehicles to lose control, severely obstructing the path. In this critical situation, the fleet must make a globally optimal, collaborative avoidance decision within milliseconds to ensure the safety of all vehicles and passengers. This decision involves not only individual path planning but also inter-vehicle communication, intent sharing, and maneuver coordination, such as which car brakes, which car changes lanes, and in what sequence.

### B. The Function of WBFT-MultiLLMN

Relying on a single LLM for decision-making in this scenario is dangerous and unreliable. Multi-LLM collaboration is essential for the following reasons. Different car manufacturers use distinct sensor suites, datasets, and algorithms to train their onboard decision-making models. For instance, Tesla's model may rely more heavily on visual data and be trained on more aggressive driving styles,[1] whereas Waymo's model heavily depends on LiDAR and is known for its conservative strategies.[2] In a complex accident scene, a single model could suffer from misjudgment due to limitations in its training data, e.g., reduced visibility for a vision-based model in smoke, or hallucinations from misinterpreting sensor signals. A MultiLLMN allows cross-validation and complementarity to form a more comprehensive situational awareness, leading to safer decisions.

Meanwhile, traditional centralized servers cannot meet the extreme security and trust requirements of this scenario [52]. Blockchain, particularly our WBFT consensus, provides indispensable guarantees. In a vehicle-to-everything environment, we must assume that some vehicles could become malicious nodes due to being hacked or malfunctioning. Such a node might broadcast false obstacle information to induce the entire fleet to brake unnecessarily, causing chaos. In a centralized system, an attack on the central server would cripple the entire fleet. In contrast, our WBFT consensus, through its unique weighted voting system, dynamically assesses each node's influence based on its historical behavior. A malicious node broadcasting anomalous data would see its voting weight drastically reduced due to a low response quality score, effectively isolating it from the decision-making process.

### C. WBFT for Collaborative Decision-Making

In this situation, the AI decision-making system of each autonomous vehicle is regarded as an LLM node. When a collaborative decision-making event is triggered, the consensus process is as follows:

- **Proposal generation:** Each LLM node will independently analyze the current traffic conditions and generate a collaborative avoidance operation instruction as a proposal.
- **Voting and consensus:** The system broadcasts the proposal of one node to all other nodes in sequence. Other nodes act as voters and compare the received proposals with their own generated proposals based on the following evaluation indicators, and cast their votes. The voting weight strictly follows the WBFT mechanism defined in our paper, which takes into account both the response quality of the node and its long-term credibility.
- **Result output:** When a proposal receives more than two-thirds of the weighted support votes, a consensus is reached. This proposal will then be adopted as the final collaborative action plan of the fleet.

[1]https://gearmusk.com/2025/04/29/tesla-raw-vision-gambit/

[2]https://waymo.com/blog/2022/09/informing-smarter-lidar-solutions-/

## VII. Performance Evaluation

In this section, we validate the efficiency of the proposed blockchain-driven Trusted MultiLLMN and WBFT.

### A. Parameter Acquisition and Setting

In the first round of consensus in MultiLLMN, individual LLMs lack prior knowledge of the response quality and trustworthiness of their peers. As a result, it is difficult for LLMs to assign appropriate voting weights to one another.

To address this initial challenge of response quality assessment, we design a comprehensive evaluation framework. Specifically, we construct question sets spanning five distinct scenarios and engage a panel of 15 volunteers, carefully selected to ensure geographical diversity, to evaluate the content generation capabilities (i.e., response quality) of ten widely recognized LLMs. The volunteer distribution includes China (4 people), the United States (4 people), the United Kingdom (2 people), Singapore (2 people), Australia (1 person), the Netherlands (1 person), and Saudi Arabia (1 person).

The ten LLMs evaluated in our experiments: Llama 3.3, WizardLM 2, GPT-4o, Gemini 2 Flash, ERNIE Bot 4.0, SparkDesk V4.0, Qwen 2.5, Doubao Pro 4k, Hunyuan-Large, and Kimi. All LLMs are deployed on a high-performance server equipped with a 96-core Intel(R) Xeon(R) Gold 5220R CPU @ 2.20GHz and 1 TB of memory. The experimental scenarios are designed to cover a broad range of LLM competencies, including memory retention, everyday task performance, artistic creativity, logical reasoning, and code generation. The final category is specifically designed to support ongoing optimization in wireless network domains. The evaluation questions are as follows.

*1) Memory Ability:* After 20 rounds of interaction, each LLM is prompted to recall and restate the initial questions and data presented in the first round. This test evaluates the model's ability to retain and recall historical information accurately, which is essential to maintain contextual consistency over extended dialogues.

*2) Daily Life Ability:* A set of practical, real-world tasks is designed, including weather forecasting, fraud detection, cooking advice, and other everyday scenarios. This evaluation assesses each LLM's grasp of common-sense knowledge and its capacity to assist users in routine daily activities.

TABLE III
LLMs GENERATION CAPABILITY SCORES

| Volunteers order | Llama 3.3 | WizardLM 2 | GPT-4o | Geimini 2 Flash | ERNIE Bot 4.0 | SparkDesk V4.0 | Qwen 2.5 | Doubao pro 4k | Hunyuan Large | Kimi |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 82 | 80 | 77 | 75 | 79 | 81 | 82 | 80 | 79 | 82 |
| 2 | 85 | 83 | 84 | 84 | 82 | 83 | 84 | 83 | 83 | 84 |
| 3 | 75 | 78 | 85 | 76 | 78 | 72 | 90 | 85 | 80 | 93 |
| 4 | 70 | 75 | 80 | 73 | 75 | 70 | 86 | 82 | 78 | 90 |
| 5 | 78 | 77 | 83 | 75 | 77 | 73 | 88 | 84 | 82 | 82 |
| 6 | 72 | 76 | 81 | 82 | 76 | 81 | 85 | 83 | 79 | 81 |
| 7 | 76 | 79 | 84 | 84 | 79 | 74 | 87 | 75 | 81 | 83 |
| 8 | 73 | 78 | 82 | 73 | 77 | 72 | 86 | 84 | 78 | 90 |
| 9 | 82 | 74 | 89 | 78 | 85 | 79 | 93 | 78 | 86 | 86 |
| 10 | 80 | 82 | 87 | 86 | 83 | 77 | 91 | 86 | 84 | 79 |
| 11 | 81 | 83 | 88 | 77 | 84 | 78 | 92 | 77 | 85 | 85 |
| 12 | 83 | 84 | 89 | 78 | 86 | 83 | 93 | 78 | 86 | 76 |
| 13 | 80 | 82 | 87 | 76 | 83 | 82 | 91 | 86 | 84 | 84 |
| 14 | 82 | 83 | 88 | 77 | 85 | 78 | 92 | 75 | 85 | 80 |
| 15 | 80 | 82 | 87 | 86 | 83 | 82 | 91 | 76 | 84 | 74 |
| Average | 78.6 | 79.7 | 84.7 | 78.6 | 80.8 | 77.6 | 88.7 | 80.8 | 82.3 | 83.3 |
| Standardization | 0.19 | 0.31 | 0.83 | 0.19 | 0.42 | 0.13 | 0.98 | 0.42 | 0.59 | 0.70 |
| Weight of $A_{i,j}^{r}$ | 0.04 | 0.065 | 0.175 | 0.04 | 0.088 | 0.027 | 0.206 | 0.088 | 0.124 | 0.147 |

*3) Artistic Ability:* LLMs are prompted to engage in creative tasks such as composing poetry, generating aesthetic commentary, producing musical pieces, and simulating traditional forms of art. Although we encourage LLM to produce diverse outputs in the field of art, within this framework, we expect LLM to generate art products that are most consistent with the strictly defined prompt inputs. For instance, those Chinese ink paintings depicting grasslands, zebras, and lions, or the realistic oil paintings of the blue and white porcelain on the table. The goal is to assess the models' expressive and creative capabilities in supporting human artistic endeavors.

*4) Logical Reasoning Ability:* This dimension involves mathematical problem-solving, optimization tasks relevant to network systems, and performance analysis in wireless communication. It is designed to measure the logical reasoning and analytical capabilities of each LLM in handling structured, domain-specific challenges.

*5) Code Generation Ability:* Building on the logical reasoning tasks, each LLM is asked to generate implementation code in various programming languages, including MATLAB, Python, and C++. The results are compared to evaluate the LLM's effectiveness in translating abstract problem-solving into executable solutions, demonstrating their potential in software development contexts.

Table III presents the average scores assigned by 15 volunteers to 10 representative LLMs across five distinct evaluation dimensions. Each dimension is scored on a scale from 0 to 100. Following score aggregation, a standardization process is applied to prepare the data for subsequent analysis. Specifically: *a)* the raw scores are transformed into a standard normal distribution with a mean of 0 and a standard deviation of 1 (i.e., Z-distribution); and *b)* a cumulative distribution function is used to map the standardized values into a uniform distribution over the interval [0, 1]. Based on normalized values, the quality weights of the LLM responses are calculated by Eq. (5).

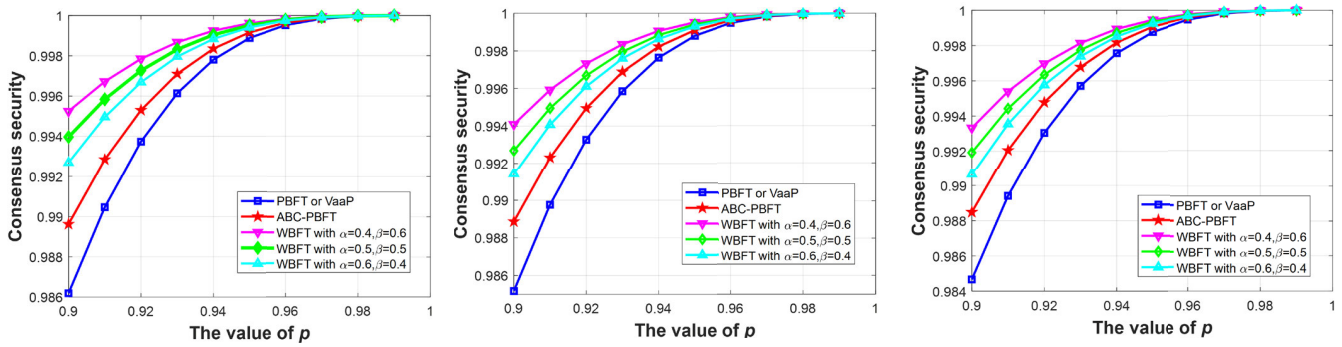
It should be noted that the trust value falls outside the scope of this study, as numerous previous studies have already explored the trust and reputation assessment schemes for network nodes [53], [54]. In our framework, trust is characterized as the probability of malicious behavior exhibited by an LLM, and three sets of trust value distributions are set up to verify the WBFT consensus performance. Specifically, we assume that trust weights follow a normal distribution, such as $N(0.1, 0.2)$, or another suitable distribution, with values constrained to the interval (0, 1).

### B. Consensus Performance

In this part, we compare the security, latency, and throughput of WBFT against three established consensus mechanisms, i.e., PBFT [22], Vote as Proof (VaaP) [50], and Artificial Bee Colony-PBFT (ABC-PBFT) [55], under varying proportions of response quality and trust weights. Specifically, we evaluate three configurations: $\alpha = 0.4$, $\beta = 0.6$; $\alpha = 0.5$, $\beta = 0.5$; and $\alpha = 0.6$, $\beta = 0.4$. These comparison baselines are widely adopted as effective defenses against Byzantine attacks and represent three distinct categories of Byzantine Fault Tolerance (BFT): classical BFT (PBFT), parallel BFT (VaaP), and BFT with selected reliable nodes (ABC-PBFT).

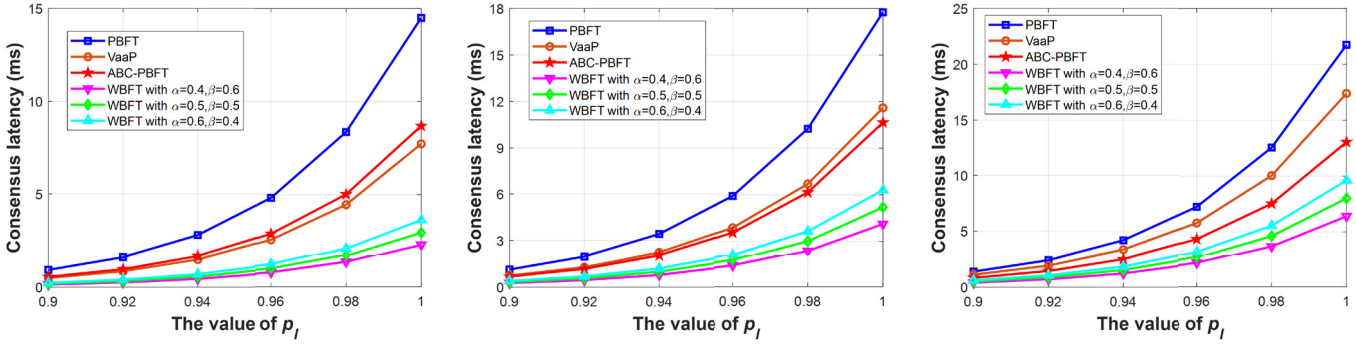
In the WBFT and ABC-PBFT schemes, five LLMs are selected as consensus nodes using the HSC and ABC algorithms, respectively. Consensus security reflects the robustness of MultiLLMN in the presence of malicious LLMs, while latency and throughput capture the system's efficiency in responding to UE requests. To evaluate performance under different trust conditions, we construct three trust weight distributions for the 10 LLMs, i.e., $N(0.1, 0.6)$, $N(0.1, 0.4)$, and $N(0.1, 0.2)$. The mean is kept constant at 0.1 to ensure that the total trust weight across all LLMs remains normalized to 1.

*1) Consensus Security:* Figs. 3 (a)-(c) show that WBFT has significantly better consensus security than other consensus.




(a) The initial trust weights ~ $N(0.1, 0.6)$ (b) The initial trust weights ~ $N(0.1, 0.4)$ (c) The initial trust weights ~ $N(0.1, 0.2)$

Fig. 3. Consensus security. (a) The initial trust weights of LLMs follow a $N(0.1, 0.6)$ distribution. (b) The initial trust weights of LLMs follow a $N(0.1, 0.4)$ distribution. (c) The initial trust weights of LLMs follow a $N(0.1, 0.2)$ distribution.




(a) The initial trust weights ~ $N(0.1, 0.6)$ (b) The initial trust weights ~ $N(0.1, 0.4)$ (c) The initial trust weights ~ $N(0.1, 0.2)$

Fig. 4. Consensus latency. (a) The initial trust weights of LLMs follow a $N(0.1, 0.6)$ distribution. (b) The initial trust weights of LLMs follow a $N(0.1, 0.4)$ distribution. (c) The initial trust weights of LLMs follow a $N(0.1, 0.2)$ distribution.

When $p = 0.9$, the consensus comparison already has a success rate of over $98\%$, and WBFT can still achieve an improvement of $1.2\%$. This is because WBFT effectively reduces the impact of Byzantine attacks by suppressing the voting power of malicious nodes. PBFT and VaaP result in the same performance due to a fixed fault tolerance threshold $2/3$. Although ABC-PBFT benefits from selecting some reliable LLMs as consensus nodes, it has higher consensus security than these two. However, the consensus nodes it selects are static and cannot be dynamically adjusted according to the network conditions and LLM trust, compared to WBFT. Therefore, its consensus security is lower than that of WBFT.

Additionally, in various combinations of weight ratios, the trust weight is of greater importance to consensus security than the response quality weight. The reason for this is that when the value $\beta$ is high, the WBFT consensus is better able to exert the voting power of a trusted LLM.

Furthermore, we observe that as the variance $\sigma^2$ of the initial trust weight distribution $B^r_{i,j}$ decreases, the consensus security of WBFT also drops. This phenomenon can be attributed to the fact that variance reflects the dispersion of trust weights across LLMs. A higher variance implies that some LLMs may receive significantly higher trust weights, increasing their influence in the consensus process and thereby enhancing the likelihood of achieving consistency. Mathematically, as shown in Eq. (9), the mean trust value remains fixed at 0.1, which is below the Byzantine fault tolerance threshold of $2/3$. Taking the derivative of the consensus probability with respect to the variance yields a positive value, indicating that an increase in variance shifts the distribution rightward and raises the probability of surpassing the consensus threshold. Additionally, we can observe that WBFT exhibits greater improvements in consensus security compared to the other two baselines when the trust weight variance is high. This suggests that the influence of highly trusted LLMs is more effectively leveraged in WBFT under conditions of greater trust dispersion.

*2) Consensus Latency:* To evaluate consensus latency, we set the bandwidth $B$ to 15 kHz, the channel capacity $C$ to 15 kbps, and the transmission rate $R$ to 10 kbps, respectively. We suppose that the MultiLLMN has 4 clusters. Then, we use the probability $P_l$ as a horizontal coordinate to study and compare the latency of each consensus and its influence parameters.

Figs. 4 (a)-(c) respectively illustrate the latency of WBFT and other baselines under different initial trust weight distributions. Under any circumstances, the latency of WBFT is lower than that of other consensus. Specifically, in various simulation scenarios, WBFT can achieve a latency optimization of $27\%$ to $83\%$ compared to the three compared consensus. It indicates that it can efficiently respond to the demands of user devices and provide reliable answers. This is because the two-round voting consensus and our pipeline mechanism improve consensus efficiency. Meanwhile, compared with VaaP, WBFT has a higher consensus security, thus avoiding as much as possible

the delay of leader re-election. Importantly, clustering also shrinks the scope of consensus voting, thereby further reducing consensus latency.

Additionally, the ratio of response quality and trust weights also affects consensus latency by affecting consensus security. Specifically, the experimental results demonstrate that latency is further reduced with increasing $\beta$ value, primarily due to the positive correlation between increased trust weight and increased probability of consensus success. It effectively mitigates the occurrence of re-election scenarios where a leader struggles to secure the required $2/3$ vote weight threshold. Moreover, the variance $\sigma^2$ of the initial trust weight distribution $B^r_{i,j}$ impacts consensus latency as well. Specifically, as the variance decreases, the latency increases. This is because lower variance reduces consensus security, leading to a higher likelihood of leader re-elections and consequently prolonging the consensus process. We further observe that under high-variance conditions, WBFT achieves significantly lower latency compared to the other consensus mechanisms, which can be attributed to its stronger consensus security, as previously discussed.

An additional observation concerns the performance of ABC-PBFT and VaaP under varying trust weight variances. When $\sigma^2$ is large, ABC-PBFT benefits from its ability to pre-select reliable LLMs, enabling faster consensus and outperforming VaaP in terms of efficiency. However, when $\sigma^2$ is small and trust differences among LLMs become less pronounced, ABC-PBFT loses its advantage, and its latency performance converges with or falls behind that of VaaP. This further highlights the effectiveness of the HSC algorithm used in WBFT, which demonstrates superior capability in identifying trusted, low-latency consensus nodes compared to the ABC-based approach.

*3) Consensus Throughput:* Consensus throughput is evaluated under the same simulation environment and configuration settings as those used for consensus latency, with results presented in Figs. 5 (a)–(c).

In general, throughput exhibits an inverse relationship to latency. Nevertheless, consistent with latency results, WBFT consistently demonstrates the highest throughput performance among all baselines. Especially when $p_l = 0.9$, the WBFT can achieve a throughput optimization effect of over 90% compared to the consensus method. The reasons are twofold. First, WBFT employs a multi-chain and pipelined architecture, enabling it to handle multiple UE requests concurrently. Second, its high consensus security reduces the likelihood of disruptions caused by malicious LLMs. Furthermore, the proposed HSC algorithm effectively narrows the consensus scope, further contributing to improved throughput.

Additionally, the ratio of response quality to trust weights, as well as the initial trust weight distribution variance $\sigma^2$ of LLMs, exhibit effects on throughput that are consistent with their impacts on consensus security and latency. For VaaP and ABC-PBFT, variations in trust weight variance $\sigma^2$ do not alter the relative throughput performance between the two schemes, unlike their impact on latency. VaaP consistently achieves higher throughput than ABC-PBFT, primarily due to its parallel block processing mechanism.

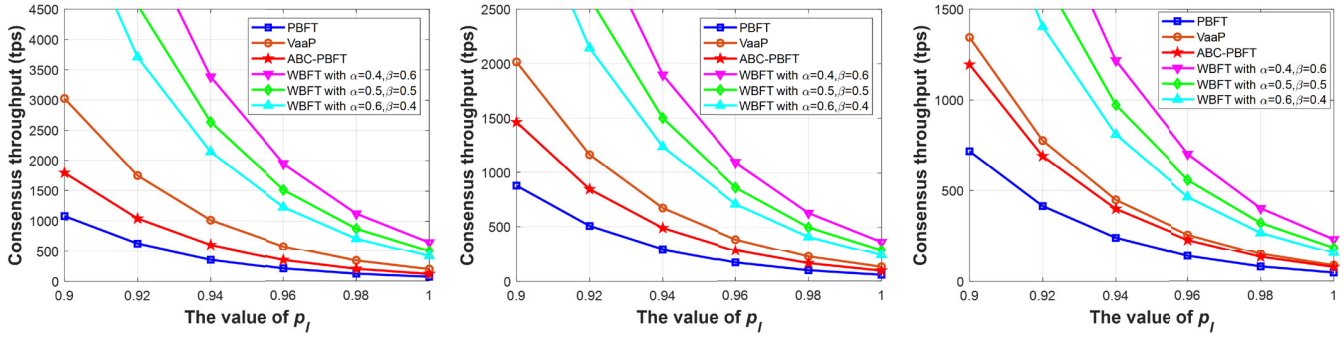
*4) Ablation Study:* Through comparative experiments, we have demonstrated that WBFT achieves superior consensus security and efficiency compared to existing consensus mechanisms. To further validate the contribution of individual modules within WBFT, we conduct ablation tests focusing on two key components: the HSC algorithm and the weight-allocation scheme. The simulation parameters for these tests are consistent with those used in earlier consensus performance evaluations.

The results of the ablation study in terms of consensus security, latency, and throughput are illustrated in Figs. 6 (a)–(c). Note that the configuration without the HSC algorithm is denoted as w/o HSC, and the configuration without the weight-allocation scheme is denoted as w/o Weighted. These results are obtained under the conditions where the initial trust weights follow $N(0.1, 0.6)$ and $\alpha = \beta = 0.5$. From Fig. 6, we can observe that the absence of either the HSC algorithm or the weight allocation mechanism leads to degraded consensus performance. Specifically, the HSC algorithm enhances consensus by excluding LLMs with low trust values or high communication latency, thereby reducing the influence of malicious LLMs and improving overall consensus efficiency. Meanwhile, the weight-allocation scheme decreases the voting influence of less reputable LLMs, further strengthening consensus security. It also increases the consensus success rate and reduces latency by minimizing leader reselection events caused by consensus failures. In conclusion, these findings underscore the indispensable role of the HSC algorithm and weight allocation scheme in achieving the performance improvements observed with WBFT.

### C. UE Satisfaction

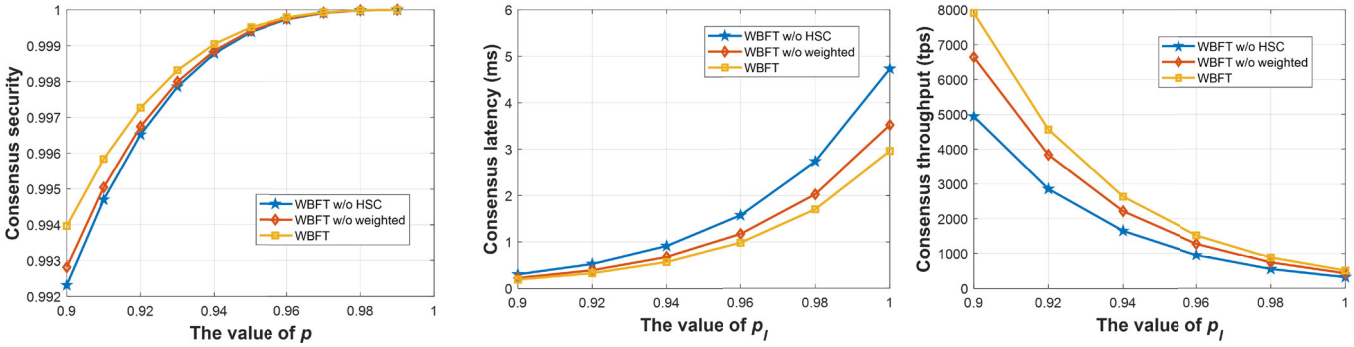
In this part, we evaluate UE satisfaction with three different architectures: a single LLM, a MultiLLMN without blockchain consensus, and the Trusted MultiLLMN proposed in this work. The evaluation is carried out using the set of questions described in Section VII-A. User satisfaction is determined by surveying the same group of 15 volunteers and averaging the results for each task. Each volunteer rated their responses on a 100-point scale. Specifically, the single LLM scenario involves volunteers randomly using one of the LLMs evaluated in Section VII-A. The MultiLLMN configuration refers to a centralized architecture without blockchain consensus participation, leaving the system vulnerable to the influence of malicious LLMs. In contrast, the Trusted MultiLLMN represents the WBFT consensus-driven decentralized system we have designed. The comparison among these three architectures essentially serves as an ablation study to assess the individual and combined contributions of LLM collaboration and WBFT consensus to overall generation quality.

Fig. 7 (a) illustrates the UE satisfaction of three architectures across five evaluation dimensions. Notably, the single LLM setup yields the lowest satisfaction scores and the highest standard deviation, highlighting its limited ability to generalize across diverse tasks and the inherent variability among different LLMs. In comparison, WBFT-driven Trusted MultiLLMN achieves higher average satisfaction and reduced variability



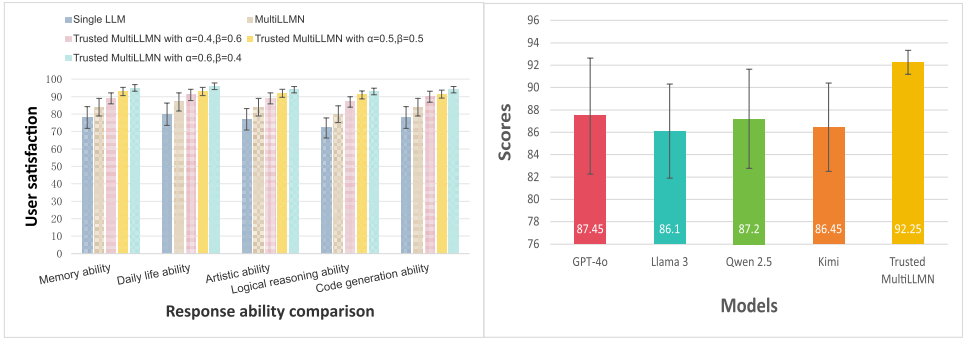

(a) The initial trust weights ~ $N(0.1, 0.6)$ (b) The initial trust weights ~ $N(0.1, 0.4)$ (c) The initial trust weights ~ $N(0.1, 0.2)$

Fig. 5. Consensus throughput. (a) The initial trust weights of LLMs follow a $N(0.1, 0.6)$ distribution. (b) The initial trust weights of LLMs follow a $N(0.1, 0.4)$ distribution. (c) The initial trust weights of LLMs follow a $N(0.1, 0.2)$ distribution.




(a) Consensus security (b) Consensus latency (c) Consensus throughput

Fig. 6. Ablation test of WBFT when the initial trust weights follow a $N(0.1, 0.6)$ with $\alpha = \beta = 0.5$. (a) Consensus security. (b) Consensus latency. (c) Consensus throughput.




(a) User satisfaction scores (b) Performance comparison in autonomous driving

Fig. 7. (a) User sanctification. (b) Performance comparison in autonomous driving.

compared to baseline MultiLLMN. Specifically, the user satisfaction of the MultiLLMN driven by WBFT can be improved by an average of 20.5% compared to that of a single LLM, and by 8.8% compared to the MultiLLMN without consensus participation. This improvement is attributed to the consensus mechanism, which mitigates the risk of misinformation from Byzantine LLMs and strengthens overall system reliability. Furthermore, increasing the weight assigned to response quality in the WBFT consensus process is found to correlate with higher UE satisfaction.

### *D. Performance in Autonomous Driving*

To verify the effectiveness of the WBFT-driven Trusted MultiLLMN for the cases described in Section VI, we randomly select 20 complex traffic accidents from an open road accident dataset[3] and write detailed descriptions. Each description included key information such as the location of the accident, the status of the involved vehicles, speed, road conditions, etc. For LLM, its task is to generate clear avoidance instructions.

Then, we select four outstanding LLMs for this experiment, including GPT-4o, Llama 3, Qwen 2.5, and Kimi. For these 20 scenarios, we have each of these four LLMs independently generate a command. Subsequently, we regard the commands as proposals and enable the other LLMs to vote based on the safety of the avoidance results. To simplify the simulation

[3]https://github.com/swalihopc/Road-Traffic-Accident-Data

process, we assume all LLMs start with equal trust weights, and let $\alpha = \beta$ in WBFT. Furthermore, the 15 volunteers rated each instruction as well as the final consensus instruction. The score range is also from 1 to 100, and we take the average score.

As shown in Fig. 7 (b), the mean score of the WBFT-driven Trusted MultiLLMN (92.25) is significantly higher than that of any individual LLM (whose scores average between 86 and 88). This indicates that the consensus effectively aggregates the strengths of each model and filters out potentially suboptimal decisions, leading to a superior collective outcome. The standard deviation of the WBFT consensus result (1.07) is far smaller than that of any individual LLM (ranging from 3.95 to 5.18). This shows that the output of the consensus mechanism is highly stable and reliable, consistently maintaining a high quality of decision-making in autonomous driving and drastically reducing the risk of catastrophic errors.

## VIII. Conclusion

In this paper, we have proposed a MultiLLMN framework to address the inherent limitations of individual LLMs, such as hallucinations, biased outputs, and limited generalization caused by static or incomplete training data. To ensure that collaboration among multiple LLMs produces reliable and trustworthy responses, we have presented the Trusted MultiLLMN driven by blockchain. Moreover, we have proposed the WBFT consensus mechanism. Unlike traditional consensus methods that assign equal voting rights or rely on fixed trust assumptions, WBFT dynamically integrates both response quality and trust levels to guide consensus decisions.

Our simulation results have demonstrated that WBFT not only enhances consensus security in the presence of potentially malicious LLMs but also significantly reduces latency and increases throughput compared to PBFT, VaaP, and ABC-PBFT. Furthermore, the Trusted MultiLLMN built on WBFT exhibits stronger responsiveness and reliability than both standalone LLMs and collaborative LLM structures that lack consensus coordination. It can achieve a 20.5% increase in user satisfaction for one of them and an 8.8% increase for the other. Moreover, it also performed exceptionally well in the case study of accident avoidance in autonomous driving. Due to the high costs associated with LLM and blockchain consensus, in the future, we will further explore the lightweight deployment and resource orchestration of this emerging framework in wireless edges to facilitate the resolution of practical wireless problems.

## References


[1] J. Hu et al., "Federated large language model: Solutions, challenges and future directions," *IEEE Wireless Commun.*, vol. 32, no. 4, pp. 82–89, Aug. 2025.

[2] Y. Chang et al., "A survey on evaluation of large language models," *ACM Trans. Intell. Syst. Technol.*, vol. 15, no. 3, pp. 1–45, 2024.

[3] P. P. Ray, "ChatGPT: A comprehensive review on background, applications, key challenges, bias, ethics, limitations and future scope," *Internet Things Cyber-Phys. Syst.*, vol. 3, pp. 121–154, Jan. 2023.

[4] F. Jiang et al., "Large AI model empowered multimodal semantic communications," *IEEE Commun. Mag.*, vol. 63, no. 1, pp. 76–82, Jan. 2025.

[5] J. Wang et al., "Generative AI enabled robust data augmentation for wireless sensing in ISAC networks," 2025, *arXiv:2502.12622*.

[6] L. Zhou, W. Schellaert, F. Martínez-Plumed, Y. Moros-Daval, C. Ferri, and J. Hernández-Orallo, "Larger and more instructable language models become less reliable," *Nature*, vol. 634, no. 8032, pp. 61–68, Oct. 2024.

[7] E. Gibney, "What are the best AI tools for research? Nature's guide," *Nature*, Feb. 2025.

[8] M. Ding et al., "Easy2Hard-bench: Standardized difficulty labels for profiling llm performance and generalization," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 37, 2024, pp. 44323–44365.

[9] J. Lu, Z. Pang, M. Xiao, Y. Zhu, R. Xia, and J. Zhang, "Merge, ensemble, and cooperate! A survey on collaborative strategies in the era of large language models," 2024, *arXiv:2407.06089*.

[10] D. M. Owens et al., "A multi-LLM debiasing framework," 2024, *arXiv:2409.13884*.

[11] S. Feng, W. Shi, Y. Wang, W. Ding, V. Balachandran, and Y. Tsvetkov, "Don't hallucinate, abstain: Identifying LLM knowledge gaps via multi-LLM collaboration," 2024, *arXiv:2402.00367*.

[12] X. Wang et al., "Wireless hallucination in generative AI-enabled communications: Concepts, issues, and solutions," 2025, *arXiv:2503.06149*.

[13] S. Wang, J. Deng, Q. Li, J. Wu, and Z. Zhao, "Performance analysis on the applications of large language models: A case for elderly care," in *Proc. IEEE Int. Conf. High Perform. Comput. Commun. (HPCC)*, Dec. 2024, pp. 145–151.

[14] S. Marro et al., "A scalable communication protocol for networks of large language models," 2024, *arXiv:2410.11905*.

[15] H. Luo, Y. Wu, G. Sun, H. Yu, and M. Guizani, "ESCM: An efficient and secure communication mechanism for UAV networks," *IEEE Trans. Netw. Service Manage.*, vol. 21, no. 3, pp. 3124–3139, Jun. 2024.

[16] M. F. M. Firdhous, W. Elbreiki, I. Abdullahi, B. H. Sudantha, and R. Budiarto, "WormGPT: A large language model chatbot for criminals," in *Proc. 24th Int. Arab Conf. Inf. Technol. (ACIT)*, Dec. 2023, pp. 1–6.

[17] X. Zuo et al., "Federated TrustChain: Blockchain-enhanced LLM training and unlearning," 2024, *arXiv:2406.04076*.

[18] H. Luo et al., "A trustworthy multi-LLM network: Challenges, solutions, and a use case," 2025, *arXiv:2505.03196*.

[19] H. Luo et al., "Toward edge general intelligence with multiple-large language model (multi-LLM): Architecture, trust, and orchestration," *IEEE Trans. Cognit. Commun. Netw.*, early access, Sep. 23, 2025, doi: 10.1109/TCCN.2025.3612760.

[20] H. Luo, G. Sun, C. Chi, H. Yu, and M. Guizani, "Convergence of symbiotic communications and blockchain for sustainable and trustworthy 6G wireless networks," *IEEE Wireless Commun.*, vol. 32, no. 2, pp. 18–25, Apr. 2025.

[21] H. Luo et al., "Wireless blockchain meets 6G: The future trustworthy and ubiquitous connectivity," *IEEE Commun. Surveys Tuts.*, early access, Jul. 31, 2025, doi: 10.1109/COMST.2025.3593918.

[22] M. Castro and B. Liskov, "Practical Byzantine fault tolerance," in *Proc. 3rd USENIX Symp. Operating Syst. Design Implement.*, 1999, pp. 173–186.

[23] M. Yin, D. Malkhi, M. K. Reiter, G. G. Gueta, and I. Abraham, "HotStuff: BFT consensus with linearity and responsiveness," in *Proc. ACM Symp. Princ. Distrib. Comput.*, Jul. 2019, pp. 347–356.

[24] H. Luo, J. Luo, and A. V. Vasilakos, "BC4LLM: A perspective of trusted artificial intelligence when blockchain meets large language models," *Neurocomputing*, vol. 599, Sep. 2024, Art. no. 128089.

[25] X. Zuo, M. Wang, T. Zhu, S. Yu, and W. Zhou, "Large language model federated learning with blockchain and unlearning for cross-organizational collaboration," 2024, *arXiv:2412.13551*.

[26] C. Geren, A. Board, G. G. Dagher, T. Andersen, and J. Zhuang, "Blockchain for large language model security and safety: A holistic survey," *ACM SIGKDD Explorations Newslett.*, vol. 26, no. 2, pp. 1–20, Jan. 2025.

[27] S. Feng et al., "When one LLM drools, multi-LLM collaboration rules," 2025, *arXiv:2502.04506*.

[28] S. Z. Shen, H. Lang, B. Wang, Y. Kim, and D. Sontag, "Learning to decode collaboratively with multiple language models," 2024, *arXiv:2403.03870*.

[29] F. Wan, X. Huang, D. Cai, X. Quan, W. Bi, and S. Shi, "Knowledge fusion of large language models," 2024, *arXiv:2401.10491*.

[30] B. Chen, G. Li, X. Lin, Z. Wang, and J. Li, "BlockAgents: Towards Byzantine-robust LLM-based multi-agent coordination via blockchain," in *Proc. ACM Turing Award Celebration Conf.*, Jul. 2024, pp. 187–192.

[31] Y. Liu et al., "Blockchain-empowered lifecycle management for AI-generated content products in edge networks," *IEEE Wireless Commun.*, vol. 31, no. 3, pp. 286–294, Jun. 2024.

[32] M. A. Bouchiha et al., "LLMChain: Blockchain-based reputation system for sharing and evaluating large language models," in *Proc. IEEE 48th Annu. Comput., Softw., Appl. Conf. (COMPSAC)*, Jul. 2024, pp. 439–448.

[33] Y. Lin et al., "Blockchain-based efficient and trustworthy AIGC services in metaverse," *IEEE Trans. Services Comput.*, vol. 17, no. 5, pp. 2067–2079, Sep. 2024.

[34] H. Li and J. Wang, "Collaborative annealing power K-means++ clustering," *Knowl.-Based Syst.*, vol. 255, Jan. 2022, Art. no. 109593.

[35] F. Liu and Y. Deng, "Determine the number of unknown targets in open world based on elbow method," *IEEE Trans. Fuzzy Syst.*, vol. 29, no. 5, pp. 986–995, May 2021.

[36] Y. Yu et al., "TierFlow: A pipelined layered BFT consensus protocol for large-scale blockchain," in *Proc. IEEE 23rd Int. Conf. Trust, Secur. Privacy Comput. Commun. (TrustCom)*, Dec. 2024, pp. 624–635.

[37] D. Luo, G. Sun, H. Yu, and M. Guizani, "Blockchain-based cross-domain authentication with dynamic domain participation in IoT," *IEEE Internet Things J.*, vol. 12, no. 5, pp. 5385–5395, Mar. 2025.

[38] H. Luo et al., "ESIA: An efficient and stable identity authentication for Internet of Vehicles," *IEEE Trans. Veh. Technol.*, vol. 73, no. 4, pp. 5602–5615, Apr. 2024.

[39] W. Li, C. Feng, L. Zhang, H. Xu, B. Cao, and M. A. Imran, "A scalable multi-layer PBFT consensus for blockchain," *IEEE Trans. Parallel Distrib. Syst.*, vol. 32, no. 5, pp. 1146–1160, May 2021.

[40] P. Révész, *The Laws of Large Numbers*, vol. 4. New York, NY, USA: Academic, 2014.

[41] R. M. Dudley, *Uniform Central Limit Theorems*, vol. 142. Cambridge, U.K.: Cambridge Univ. Press, 2014.

[42] B. Chang, L. Zhang, L. Li, G. Zhao, and Z. Chen, "Optimizing resource allocation in URLLC for real-time wireless control systems," *IEEE Trans. Veh. Technol.*, vol. 68, no. 9, pp. 8916–8927, Sep. 2019.

[43] D. Yu, W. Li, H. Xu, and L. Zhang, "Low reliable and low latency communications for mission critical distributed industrial Internet of Things," *IEEE Commun. Lett.*, vol. 25, no. 1, pp. 313–317, Jan. 2021.

[44] H. Luo, Q. Zhang, H. Yu, G. Sun, and S. Xu, "Symbiotic PBFT consensus: Cognitive backscatter communications-enabled wireless PBFT consensus," in *Proc. IEEE Global Commun. Conf.*, Dec. 2023, pp. 910–915.

[45] J. Cao, S. Leng, L. Zhang, M. Imran, and H. Chai, "A V2V empowered consensus framework for cooperative autonomous driving," in *Proc. IEEE Global Commun. Conf.*, Rio de Janeiro, Brazil, May 2022, pp. 5729–5734.

[46] H. Luo, G. Sun, H. Yu, B. Lei, and M. Guizani, "An energy-efficient wireless blockchain sharding scheme for PBFT consensus," *IEEE Trans. Netw. Sci. Eng.*, vol. 11, no. 3, pp. 3015–3027, May 2024.

[47] H. Luo, Q. Zhang, G. Sun, H. Yu, and D. Niyato, "Symbiotic blockchain consensus: Cognitive backscatter communications-enabled wireless blockchain consensus," *IEEE/ACM Trans. Netw.*, vol. 32, no. 6, pp. 5372–5387, Dec. 2024.

[48] H. Luo, X. Yang, H. Yu, G. Sun, B. Lei, and M. Guizani, "Performance analysis and comparison of nonideal wireless PBFT and RAFT consensus networks in 6G communications," *IEEE Internet Things J.*, vol. 11, no. 6, pp. 9752–9765, Mar. 2024.

[49] H. Luo, K. Yang, G. Sun, H. Yu, Q. Huang, and Y. Zhang, "A multi-chain consensus for power big data transaction in generation-grid-load-storage integrated networks," in *Proc. IEEE Global Commun. Conf.*, Dec. 2024, pp. 2455–2460.

[50] X. Fu, H. Wang, and P. Shi, "Votes-as-a-proof (VaaP): Permissioned blockchain consensus protocol made simple," *IEEE Trans. Parallel Distrib. Syst.*, vol. 33, no. 12, pp. 4964–4973, Dec. 2022.

[51] D. Luo, Y. Zhang, G. Sun, H. Yu, and D. Niyato, "An efficient consensus algorithm for blockchain-based cross-domain authentication in bandwidth-constrained wide-area IoT networks," *IEEE Internet Things J.*, vol. 11, no. 19, pp. 31917–31931, Oct. 2024.

[52] S. Tu, H. Yu, A. Badshah, M. Waqas, Z. Halim, and I. Ahmad, "Secure Internet of Vehicles (IoV) with decentralized consensus blockchain mechanism," *IEEE Trans. Veh. Technol.*, vol. 72, no. 9, pp. 11227–11236, Sep. 2023.

[53] H. Guo, X. Chen, X. Zhou, and J. Liu, "Trusted and efficient task offloading in vehicular edge computing networks," *IEEE Trans. Cognit. Commun. Netw.*, vol. 10, no. 6, pp. 2370–2382, Dec. 2024.

[54] J. Liao et al., "Graph attention network-based block propagation with optimal AoB and reputation in Web 3.0," *IEEE Trans. Cognit. Commun. Netw.*, vol. 10, no. 6, pp. 2427–2441, Dec. 2024.

[55] J. Xu, Y. Zhao, H. Chen, and W. Deng, "ABC-GSPBFT: PBFT with grouping score mechanism and optimized consensus process for flight operation data-sharing," *Inf. Sci.*, vol. 624, pp. 110–127, May 2023.