Federated Learning Using GPT-4 Boosted Particle Swarm Optimization for Compact Neural Architecture Search

Di Wang

Industrial AI Group, Foxconn, Wisconsin, USA
Email: di.wang@fewidev.com

***Abstract*—In response to the growing need for privacy-preserving mobile intelligence, this study introduces a new approach that combines Generative Pre-trained Transformer 4 (GPT-4), a state-of-the-art large language model, with Particle Swarm Optimization (PSO) in a two-step process. This method is designed to find efficient neural network structures in federated learning and address issues like high communication costs and unstable network conditions. Leveraging the prowess of GPT-4 for initial population guidance in the Neural Architecture Search (NAS) process, our approach focuses on optimizing neural network architectures that demand minimal data exchange between clients and servers. This is achieved through a variable-length PSO encoding and decoding mechanism at the upper level, ensuring not only a thorough search for efficient architectures but also their optimization for compactness and effectiveness. Additionally, a standard PSO technique is applied at the lower level to optimize neural network weights, thus boosting model performance with reduced communication load. Our methodology's superiority is demonstrated via benchmark comparisons with FedAvg and FedPSO on the CIFAR-10 dataset, under both normal and compromised network scenarios.**

***Keywords*—Generative Pre-trained Transformer 4 (GPT-4), federated learning, Particle Swarm Optimization (PSO), Neural Architecture Search (NAS), communication cost**

I. Introduction

In recent times, the field of deep learning has garnered significant interest across various domains, including robot control [1, 2], task planning [3, 4], manufacturing [5, 6], and smart transportation [7, 8]. Convolutional Neural Networks (CNNs), in particular, have shown exceptional capability in extracting sophisticated feature representations, albeit requiring extensive labor and specialized knowledge. For instance, the Visual Geometry Group-16 (VGG-16) model is characterized by its extensive architecture, encompassing over 130 million parameters and necessitating approximately 500 MB of memory. To process an image measuring 224×224 pixels, it performs 15.3 billion floating-point operations. On the other hand, ResNet-50, distinguished by its meticulously crafted residual and bottleneck blocks, contains more than 25 million parameters. This model requires 98 MB of memory and executes 3.8 billion floating-point operations to analyze an image of the same dimensions.

Addressing the challenges of manual design in neural network architectures, Neural Architecture Search (NAS) emerges as a resource-intensive way aimed at identifying the optimal architecture from a broad spectrum of possibilities. The primary hurdle in NAS lies in the extensive training required for numerous potential models, often necessitating thousands of hours on advanced Graphics Processing Unit (GPU) setups [9]. For example, Zoph *et al.* [10] trained a well-designed recurrent neural network with 28 days of training on 800 GPUs. To mitigate the heavy computational demands, some strategies propose compromises, such as limiting training duration, utilizing smaller data sets, or simplifying the architecture [11–13]. Yet these approaches do not fully leverage NAS's potential for parallel processing. Moreover, these strategies often centralize training data, overlooking the benefits of a decentralized approach across various computational frameworks, thus raising concerns over data privacy and security, especially with sensitive information on personal and medical data that cannot be freely transferred or accessed online due to stringent data protection regulations in regions like the European Union [14].

Federated learning emerges as a potent solution to the challenges previously outlined, offering a way to harness the power of decentralized deep-learning model development. This approach leverages parallel computing resources effectively, enabling the exploration of sophisticated models without the need to frequently exchange large volumes of sensitive data between clients and servers. The distinctions between federated learning and traditional distributed learning are numerous and significant, encompassing aspects such as [15]: the purpose of privacy protection, mass data collection cost, the unknown training data distribution, heterogeneous computation platforms, low computation capability, security threat, and the potential for inconsistent connectivity or device failures. These complexities present

Manuscript received May 31, 2024; revised June 20, 2024; accepted July 5, 2024; published September 5, 2024.

considerable obstacles to the practical implementation of federated learning.

Integrating federated learning with NAS is crucial because it allows for the development of optimized models that respect user privacy while enhancing performance across distributed networks [15]. Federated learning maintains data privacy by processing data locally on user devices, but this can lead to challenges with non-Independent and Identically Distributed (non-IID) data, affecting model performance. NAS addresses this by automatically discovering optimal neural network architectures that are better suited to the unique data distributions and resource constraints of each device in the network. This combination not only boosts model efficiency and accuracy but also tailors models to diverse application requirements, making it a powerful approach for deploying intelligent systems in privacy-sensitive environments. Liu *et al.* [16] demonstrate that effectively integrating federated learning with NAS optimizes the capabilities of edge computing, achieving a significant reduction in completion time by 30.6%. Similar findings are found at [17].

Unlike the standard NAS processes, as shown in Fig. 1, federated learning operates under a unique paradigm where each participant relies solely on their proprietary data for model training and refinement. Meanwhile, a central server orchestrates the collective effort, merging updates from client models, steering the exploration strategy without direct access to raw data, and disseminating refined parameters and model structures back to the participants.

Among the strategies for model integration in federated settings, FedAvg [18] stands out for its simplicity, aggregating the parameter updates from client-side neural networks. Nevertheless, this method encounters challenges as model complexity increases, particularly with deeper layers, where the volume of parameters to be shared escalates communication costs significantly. In federated learning scenarios, the bandwidth consumed by data transmission often surpasses that used in computations, underscoring the need to minimize network communication times to enhance overall efficiency. This necessity is further compounded by the challenges of fluctuating network conditions, necessitating stable Wi-Fi connections and accommodating limited bandwidth to facilitate federated learning processes.

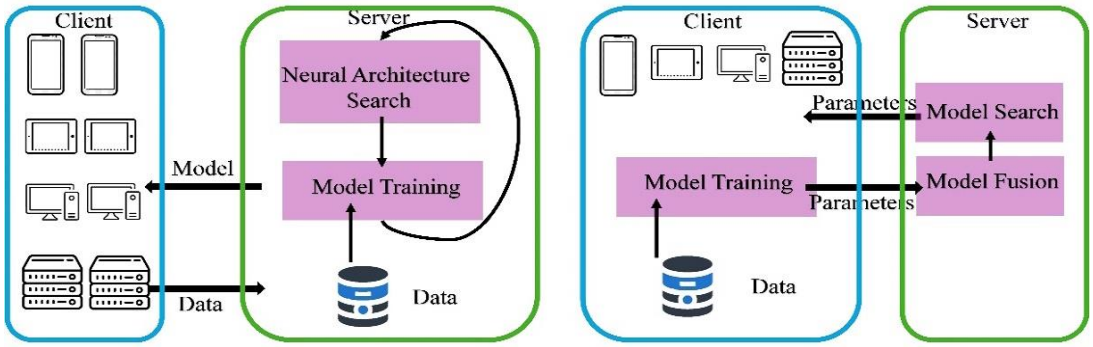

Fig. 1. Comparison of client-server training and inference architectures: traditional NAS (left) vs. Federated NAS (right).

This manuscript introduces an innovative bi-level Particle Swarm Optimization (PSO) strategy aimed at enhancing the speed of NAS and the efficiency of model training, particularly in scenarios characterized by unstable internet connectivity. PSO is selected for its simplicity, cost-effectiveness, and minimal parameter tuning requirement. In detail, at the upper level, a flexible variable-length PSO [19] is taken to automatically evolve the server's initial deep-learning model. The IP-based encoding and decoding strategy [20] is taken to represent the deep-learning model in the parameter searching space. Furthermore, the advent of generative AI breakthroughs, notably Generative Pre-trained Transformer-4 (GPT-4) [21], signifies a pivotal advancement towards achieving comprehensive "general AI", endowed with expert capabilities in both neural architecture design and programming. Zheng *et al*. [22] study the feasibility of using GPT-4 purely without fine-tuning operation to provide the directions of NAS. As is known, with the assistance of Retrieval-Augmented Generation (RAG) and fine-tuning operation [23], the GPT-4 model can provide better-quality and specific answers. In this study, we utilize specific prompts to guide GPT-4 model to generate a population of neural architectures based on the current global-optimal particle individual in PSO at the initialization stage. At the lower level, another typical PSO optimizer is taken to update the weights of neural network parameters to decrease the communication further. Instead of transmitting the blocks of neural network weights, only the best individual scores are shared without the limitation of the neural network size. To prove the effectiveness of the proposed approach under an unstable communication context, we focus on the image classification task with CNN models and compare our approach with two benchmarks on the CIFAR-10 dataset [24].

The contribution of this paper can be summarized as follows:

(1) To our best knowledge, this work is the first to harness the inference capabilities of a large language model, specifically GPT-4, within the population optimization process of PSO. We leverage GPT-4 to intelligently fine-tune neural network architectures, significantly enhancing the

quality and effectiveness of the search process in neural architecture search.

(2) We establish a comprehensive prompt engineering benchmark tailored for GPT-4 in the population search of PSO, facilitating further research and refinement in this area.

(3) We propose a new bi-level PSO approach that efficiently handles neural architecture search and parameter optimization in a federated learning context, reducing the communication overhead and enhancing computational efficiency.

(4) Utilizing a practical dataset and comparing our method against two state-of-the-art federated learning approaches, we demonstrate the robustness and reliability of our proposed framework under conditions of network instability.

The remainder of this paper is organized as follows: Section II illustrates related literature. Section III presents details of the proposed approach. In Section IV, simulations are conducted to prove the effectiveness of the proposed approach. Finally, conclusions and future studies are discussed in Section V.

II. Literature Review

Neural Architecture Search (NAS) is an evolving field focused on enhancing the automation of neural network design. This area has seen the proposal of various exploration strategies to optimize the design process including reinforcement learning [9, 10], evolutionary strategies [25], Bayesian optimization [26], gradient-based approaches like Differentiable Architecture Search (DARTS) [27], EfficientNAS [28], and Particle Swarm Optimization (PSO) [29, 30]. Specifically, Sun *et al.* [29] represent the deep-learning model with a fixed-length encoding-decoding schema. Junior *et al.* [30] propose an innovative particle-updating strategy by estimating the differences among variable-length particle neighbors.

Google originally introduced the concept of Federated Learning as a solution to enhance collaboration across Android devices. Existing works can be categorized as vertical federated learning, horizontal federated learning, federated transfer learning, cross-silo federated learning, and cross-device federated learning [31]. Because of the data heterogeneity, there are large gaps among client-trained models. Corrections and regularizations are taken to reduce the differences between local models [32, 33]. To fully utilize computation resources, Nishio *et al.* [34] select the proper clients based on their resource information. However, this will cause a bias since a large amount of work will be assigned to clients with large computation resources. Luo *et al.* [35] propose an energy-aware federated selection approach considering battery limitations. Parl *et al.* [36] propose a PSO-based federated learning approach to decrease the communication cost, where neural network parameters are optimized by a PSO. Different from our proposed approach, our bi-level PSO approach optimizes the neural network design during the federated learning process.

III. Bi-level PSO Architecture with GPT-4 Fine-Tuned Neural Network Designs

Our methodology leverages GPT-4, a state-of-the-art language model, to fine-tune the initial population generation in the upper-level PSO. This integration aims to optimize the neural architecture search by providing highly informed starting points, thus enhancing the efficiency and effectiveness of the search process. A lower-level PSO is taken to optimize the neural network parameters. In this section, we first explore the fundamentals of PSO, followed by the particle encoding and decoding methods in upper-level PSO for NAS. We then discuss leveraging GPT to fine-tune populations in upper-level PSO. Finally, we touch the lower-level PSO.

A. Particle Swarm Optimization

PSO enhances federated learning by minimizing computational and communication burdens. Unlike traditional methods such as FedAvg that require extensive data exchanges, PSO optimizes model updates using a population-based approach where only key updates like scores or partial weights are transmitted. This strategy significantly reduces the volume of data communicated, thereby improving network efficiency and processing speed on client devices. Additionally, PSO's ability to adapt to the distributed and heterogeneous nature of federated environments ensures robust model updates even under unstable network conditions, accelerating convergence and enhancing model performance.

PSO is recognized as a population-centric search algorithm, known for its minimal computational complexity, straightforward implementation, and rapid convergence rates. Within this framework, each entity functions as a particle, dynamically adjusting its position in the search space by updating its velocity. As shown in Eqs. (1) and (2), this process allows for efficient exploration and optimization of complex problem spaces, leveraging the collective intelligence of the swarm to navigate toward optimal solutions [37, 38].

$$v_{i,j}^{t+1} = \omega v_{i,j}^{t} + c_1 r_1 \left(pBest_{i,j} - x_{i,j}^{t}\right) + c_2 r_2 (gBest_j - x_{i,j}^{t}) \quad (1)$$

$$x_{i,j}^{t+1} = x_{i,j}^{t} + v_{i,j}^{t+1} \quad (2)$$

where $x_{i,j}$ denotes the position of the i^{th} particle in the j^{th} dimension, while $v_{i,j}$ represents the velocity of the i^{th} particle in the j^{th} dimension. The term $gBest_j$ is the global optimal fitness value among all particles at the j^{th} dimension. ω, c_1, and c_2 are trainable parameters that influence the algorithm's behavior, including inertia, cognitive component, and social component, respectively. Random factors r_1 and r_2 are introduced to enhance the exploratory capability of the swarm, ensuring a diverse search across the potential solution landscape. The core principle of PSO lies in tracking and updating the swarm's movement towards these optimal points, leveraging individual discoveries (local bests) and collective wisdom (global best) to navigate the search space efficiently [39].

B. Particle Encoding and Decoding Schema

In the upper-level PSO, a robust particle encoding and decoding scheme is essential for representing diverse neural network architectures within PSO particles. Wang *et al.* [20] have introduced a compact encoding and decoding schema inspired by internet protocols. As shown in Table I, the details of the encoding schema are presented, including the convolutional, pooling, fully connected, and disabled layers. The largest number of bits is 12. Classless Inter-Domain Routing (CIDR) [20] is taken to represent all types of CNN architectures. As shown in Table II, the IP range starts from 0.0 representing the convolutional layer with a mask length of 4, followed by the fully connected layer with a mask length of 5, pooling layer with a mask length of 5 and the disabled layer with a mask length of 5. Moreover, the disabled layer represents the combinations of different layers during the evolving process. For example, an integer string [2, 61, 18, 143, 2, 61, 18, 143, 27, 255] represents the architecture [Convolutional Layer (IP 2.61), Pooling Layer (IP 18.143), Convolutional Layer (IP 2.61), Pooling Layer (IP 18.143), Fully connected Layer (IP 27.255)]. After several rounds of exploration, the new integer string can be [2, 61, 18, 143, 2, 61, 35, 255, 27, 255] representing the architecture [Convolutional Layer (IP 2.61), Pooling Layer (IP 18.143), Convolutional Layer (IP 2.61), Disabled Layer (IP 35.255), Fully connected Layer (IP 27.255)]. More details can be found at Wang *et al*.'s work [20].

TABLE I. PARTICLE ENCODING AND DECODING SCHEMA OF CONVOLUTIONAL, POOLING, FULLY CONNECTED AND DISABLED LAYERS

Layer Type	Parameter	Range	Number of Bits	Example Value
Conv	Filter size	[1, 8]	3	2(001)
	Number of feature maps	[1, 128]	7	32(000 1111)
	Stride size	[1, 4]	2	2(01)
	Summary		12	001 000 1111 01
Pooling	Kernel size	[1, 4]	2	2(01)
	Stride size	[1, 4]	2	2(01)
	1(Maximal);2(Average)	[1, 2]	1	2(1)
	Place holder	[1, 128]	6	32(00 1111)
	Summary		11	01 01 0 00 1111
Fully connected	Number of Neurons	[1, 2048]	11	1024(011 11111111)
	Summary		11	011 11111111
Disabled	Place holder	[1, 2048]	11	1024(011 11111111)
	Summary		11	011 11111111

TABLE II. CIDR AND IP ADDRESS ASSIGNING OF CONVOLUTIONAL, POOLING, FULLY CONNECTED AND DISABLED LAYERS

Layer Type	CIDR	IP Range
Convolutional Layer	0.0/4	0.0–15.255
Fully Connected Layer	16.0/5	16.0–23.255
Pooling Layer	24.0/5	24.0–31.255
Disabled Layer	32.0/5	32.0–39.255

C. GPT-4 Boosted Population Initialization

In the field of population-based optimization, PSO begins by generating a varied set of individual particles, each representing a unique neural network architecture. However, designing effective neural networks requires not only domain-specific knowledge but also sophisticated techniques validated by existing works. Relying solely on random search can be inefficient, as it consumes considerable time and computational resources. Furthermore, directly translating human expertise into design rules can be complex and fraught with challenges.

To overcome these limitations, our strategy leverages the advanced capabilities of GPT-4, a cutting-edge large language model with domain knowledge in deep learning tricks and neural network designs. By using GPT-4, we move beyond simple random initialization to a more informed starting point for the optimization process. GPT-4's ability to analyze vast amounts of data and learn from diverse examples enables it to propose initial neural network architectures that are both innovative and practical. This not only speeds up the search process but also enhances the quality of the solutions, making our approach significantly more efficient and effective in exploring complex design spaces without falling into local optimal values. This strategic choice is predicated on the understanding that the initial quality of these architectures significantly impacts the efficiency and direction of the subsequent search process. Bubeck *et al.* [40] prove GPT-4's reasoning ability across multiple areas. Zhang *et al.* [22] prove that GPT-4 can generate meaningful suggestions for potentially better neural network architectures. Nevertheless, a discernible gap remains between these GPT-4 suggested architectures and the ideal, optimal structures. This discrepancy is partly due to the reliance on basic prompt engineering that lacks the incorporation of specific exploration constraints and the continuous optimization process.

To bridge this gap, our methodology includes the development of a meticulously designed prompt engineering template, which is illustrated in Fig. 2. This template is structured to guide GPT-4 in generating proposals that not only aim to surpass the performance of an initially designed CNN model for the CIFAR-10 dataset but also adhere to predefined constraints within the exploration space. The template is written as follows: "This is my currently designed CNN model for CIFAR-10 dataset, [initial model]. Please recommend [population size] new models that outperform my designed one. You should satisfy these constraints [search constraints]. Please generate your answers in this format [`plan1 ``` your first recommendation``` `plan2 ``` your second recommendation```]." An example is presented in Fig. 3, GPT-4 API call is taken in this paper to complete the query

process. In the future, we will develop a locally deployed and fine-tuned LLM model for this part, such as Llama 3 and retrieval augmented generation techniques.

D. Fitness Evaluation

In our fitness evaluation strategy, we partition the client's dataset into two subsets: a training set for model development and an evaluation set for performance assessment. To enhance the efficiency of the training phase, we employ an early-stopping mechanism that terminates the process once improvement plateaus, thereby conserving computational resources. Within the framework of the upper-level PSO, each particle is interpreted as a unique CNN architecture. These architectures undergo training for a predetermined number of epochs using the lower-level PSO in conjunction with the training dataset. Subsequently, the performance of each CNN model is thoroughly assessed on the evaluation dataset, where a series of accuracy metrics are computed. The average accuracy achieved by a model is then designated as the fitness score of the corresponding particle, reflecting the model's efficacy [41].

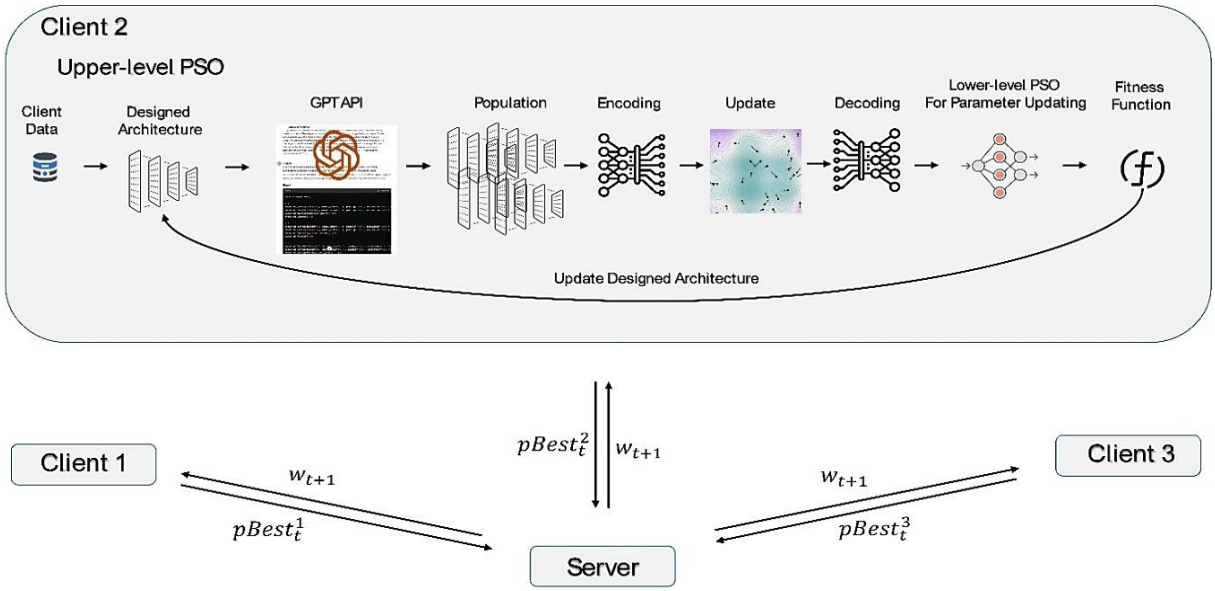

Fig. 2. Client-server architecture overview of the proposed Bi-level PSO with GPT-4 boosted population initialization for NAS.

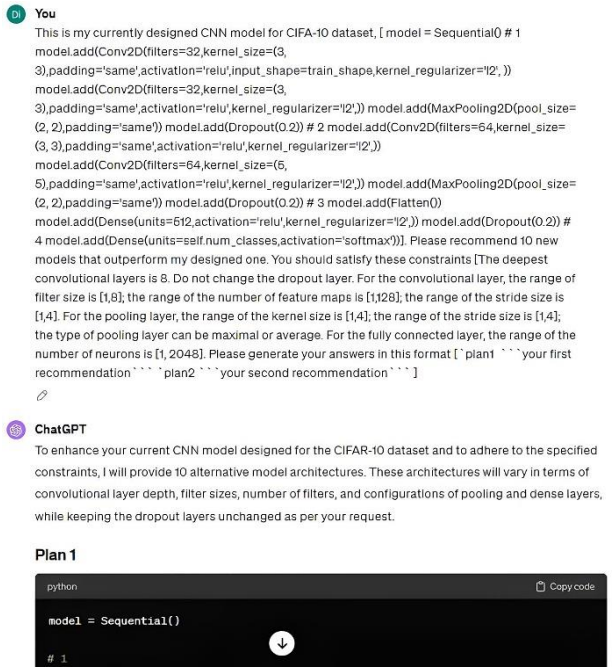

Fig. 3. GPT-4 prompt engineering benchmark for NAS.

On the other hand, the objective of the lower-level PSO is to search for the most effective neural network weights. This phase leverages the mean loss incurred by the CNN model on the training dataset as the guiding metric for the optimization process. By evaluating the model's average loss, the lower-level PSO effectively identifies the weight configurations that minimize error, thereby serving as a precise fitness function. This bifurcated approach ensures a comprehensive and efficient exploration of both architectural and weight parameters, aiming to elevate the overall performance of the CNN models within the federated learning environment.

E. Lower-Level PSO

The objective of the lower-level PSO is to optimize the neural network parameters through a PSO-based training approach. Detailed in Algorithm 1, this procedure iteratively optimizes the weights of each neural network layer. During this optimization cycle, a variable v is strategically combined with the inertia weight ω from the preceding iteration to compute the updated ω value for the current cycle. This mechanism ensures a dynamic adjustment of the inertia component, facilitating a balanced exploration and exploitation of the search space tailored to each layer's specific optimization needs. $\boldsymbol{\omega}_{\mathbf{t}}^{gBest}$ is the updated global best weights at time step t. γ denotes the learning rate while L represents the cross entropy loss function.

Algorithm 1. Pseudocode of the lower-level PSO

Initialize $\boldsymbol{v}, \boldsymbol{\omega}, \boldsymbol{\omega}^{p^{Best}}, \boldsymbol{\alpha}, \boldsymbol{c_1}, \boldsymbol{c_2}$
For each weight layer $\boldsymbol{l} = \boldsymbol{1}, \boldsymbol{2}, \ldots \boldsymbol{do}$
  $\boldsymbol{v_l} \leftarrow \boldsymbol{\alpha v_l} + \boldsymbol{c_1 r_1}(\boldsymbol{\omega}^{p^{Best}} - \boldsymbol{v_l}) + \boldsymbol{c_2 r_2}(\boldsymbol{\omega_t}^{g^{Best}} - \boldsymbol{v_l})$
End for
Update weights with $\boldsymbol{\omega} \leftarrow \boldsymbol{\omega} + \boldsymbol{v}$
For each epoch I from 1 to E do
  For each batch b do
    $\boldsymbol{\omega} \leftarrow \boldsymbol{\omega} - \boldsymbol{\gamma \nabla L}(\boldsymbol{\omega}; \boldsymbol{b})$
  End for
End for

IV. Experiments

To evaluate the effectiveness of our innovative bi-level Federated PSO methodology, enhanced by GPT-4's fine-tuning of neural network populations, we conducted a series of experiments. These experiments were designed to measure the accuracy, convergence efficiency, and reliability of our approach, particularly in scenarios characterized by unstable network conditions. The initial experiment sought to verify if our approach could achieve high accuracy and rapid convergence with reduced network communication overhead, especially when compared to the FedAvg method, yet with the advantage of a more sophisticated neural network architecture. These assessments were conducted using the CIFAR-10 dataset, with our results benchmarked against two established approaches: FedAvg and FedPSO [36].

Our experimental setup was executed on a system powered by an Intel Core i7-6850K processor and an NVIDIA GeForce GTX 1080Ti graphics card. The base CNN model, adapted from Park *et al.*'s work [36], was modified to include a dropout layer with a 0.2 probability and a standard batch normalization layer after each architectural block, as detailed in Table III. The CIFAR-10 dataset, consisting of 50,000 training images and 10,000 test images across 10 classes with each image sized at 32×32 pixels, served as the benchmark dataset for our experiments. For the FedAvg approach, we configured the system with 10 clients, with "C" values set either to 0.5 or 1, running for 30 epochs at the server level, 5 epochs at the client level, and a batch size of 10. Similarly, the FedPSO setup involved 10 clients, with the same epoch configuration and batch size as FedAvg. The PSO parameters were set to $\alpha = 0.3$, $c_1 = 0.7$, and $c_2 = 1.4$. These settings are the same with [36]. For our proposed bi-level PSO approach, the lower-level PSO configuration mirrors that of FedPSO, whereas the upper-level PSO's parameters, c_1 is 1.496, and c_2 is 1.496. The initial weight for velocity update is 0.7298.

TABLE III. Initial Parameters Settings for the CNN

Layer Type	Shape
Conv2D	3×3×32
Pooling	32
Conv2D	3×3×64
Pooling	64
FC	1024×512
FC	512×10

The performance outcomes of our proposed bi-level federated PSO approach and GPT-4 boosted population searching strategies, in comparison to benchmark methods, are delineated in Table IV. The data reveals that our approach substantially surpasses the benchmarks, registering test accuracies that are 13.82%, 18.89%, and 22.78% superior to FedPSO, FedAvg with C = 1, and FedAvg with C = 0.5, respectively.

Further insights into the efficacy of our approach are provided in Table V, which depicts the most effective CNN architecture identified through our optimization process. Distinct from the architectures generated by traditional NAS techniques, such as those reported by Huang *et al.* [19], our architecture benefits from the early involvement of GPT-4 in the initialization phase. This strategic incorporation leads to the discovery of architectures that are not only compact but also logically coherent, showcasing the advantage of leveraging advanced AI in the design process.

TABLE IV. Comparative Accuracy of Various Algorithms during the Testing Phase

Approach	Test Accuracy
FedPSO	70.12%
FedAvg (C = 1)	67.14%
FedAvg (C = 0.5)	65.00%
Our Approach	79.81%

TABLE V. Best Searched CNN Architecture

Layer Type	Shape
Conv2D	filter size: 3×3, stride: 1, feature maps: 32
Conv1D	filter size: 1×1, stride: 1, feature maps: 78
Pooling	size: 2×2, stride: 2, type: Max
Conv2D	filter size: 3×3, stride: 1, feature maps: 126
Conv2D	filter size: 2×2, stride: 1, feature maps: 106
Conv2D	filter size: 3×3, stride: 1, feature maps: 111
Pooling	size: 2×2, stride: 2, type: Max
Conv2D	filter size: 3×3, stride: 1, feature maps: 120
Conv1D	filter size: 1×1, stride: 1, feature maps: 89
Pooling	kernel size: 2×2, stride: 2, type: Average
FC	neurons: 1309

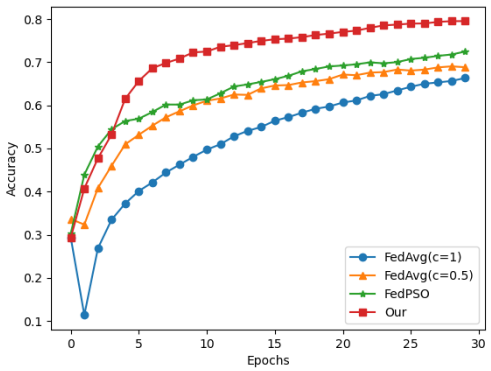

Fig. 4. Comparative accuracy of various algorithms during the training phase.

Moreover, Fig. 4 offers a visual representation of the test accuracy progression throughout the training sessions. This figure clearly illustrates the superior convergence performance of our algorithm, evidencing a steady and rapid improvement in accuracy. This trend demonstrates

that our proposed algorithm does not merely improve incrementally but does so at a pace that outstrips conventional models, thanks to the strategic use of GPT-4 for initial population enhancement and the effective optimization capabilities of our federated PSO approach. When discussing convergence speed, it's important to note that distributed training methods like federated learning are influenced by unique factors, unlike centralized training. Initially, multiple devices with limited computational resources perform local computations and share updates with a central server. The data in these scenarios is often non-IID, which can decelerate convergence, particularly under constrained computational capacities. Additionally, transmitting updates across networks incurs further delays, potentially exceeding the time spent on computations. Therefore, comparing the convergence speeds of federated and centralized learning methods directly is both challenging and generally not meaningful. Compared to FedAvg, our proposed algorithm incurs less communication delay due to transmitting smaller data volumes. However, in comparison with FedPSO, our algorithm experiences increased communication delays due to the integration of the GPT-4 API call. The GPT-4 API call delays can be further decreased if a local LLM model is utilized.

Subsequently, we evaluated the resilience of our proposed method under conditions of unstable network communication, where data packets might be lost during transmission between clients and the server. For this purpose, we designed experiments under two distinct scenarios characterized by data dropping rates of 10% and 20%, respectively. As presented in Table VI, the performance of our approach in an environment with a 10% data loss rate exceeded that of the FedPSO and FedAvg (C = 1) benchmarks by 11.74% and 25.73%, respectively. Furthermore, in a more challenging scenario with a 20% dropping rate, our method maintained its superior performance, achieving results that were 11.84% and 25.24% better than those of FedPSO and FedAvg (C = 1), respectively.

TABLE VI. COMPARATIVE ACCURACY OF VARIOUS ALGORITHMS DURING TESTING IN UNSTABLE COMMUNICATION ENVIRONMENTS

Approach	Failure Rate 10%	Failure Rate 20%
FedPSO	69.18%	68.41%
FedAvg (C = 1)	61.48%	61.09%
Our Approach	77.30%	76.51%

These findings underscore the robustness of our bi-level federated PSO approach and GPT-4 boosted population searching strategies, demonstrating its ability to sustain high levels of accuracy even in the face of significant communication challenges. The resilience of our approach in such unstable network conditions highlights its potential for practical deployment in real-world applications where network reliability cannot always be guaranteed.

V. CONCLUSION

In this study, we introduce a cutting-edge and streamlined approach to NAS, termed the bi-level federated PSO approach with GPT-4 boosted population searching strategies, tailored for federated learning environments. Our primary objective is to minimize communication overhead, particularly in settings plagued by unstable network conditions. This algorithm optimizes server-side model aggregation by prioritizing the exchange of score values rather than the models themselves, with only the highest-scoring client model being transmitted to the server for further refinement.

At the upper level of our framework, GPT-4 is leveraged as a sophisticated expert system to guide the initial population setup, informed by predefined design concepts. This process is supported by elaborate prompt engineering templates that incorporate NAS-specific constraints, ensuring a focused and efficient search. Additionally, we adopt a variable-length PSO strategy that employs a novel architecture encoding and decoding scheme, designed to generate compact yet effective neural network architectures. At the lower level, a conventional PSO methodology is utilized to fine-tune the weights of the identified neural network architectures, optimizing their performance.

To validate the efficacy of our proposed model, we conducted comparative analyses against two established benchmarks, FedAvg and FedPSO, using the CIFAR-10 dataset. Furthermore, to ascertain the resilience of our approach under challenging network conditions, we examined its performance across two scenarios characterized by data loss rates of 10% and 20%.

Looking ahead, our research will explore the application of our method to multi-objective optimization challenges, aiming to strike an optimal balance between performance efficacy and energy consumption. Additionally, we plan to take into account the diversity in client capabilities, such as variations in battery life and computational power. Expanding our investigation, we will also delve into the potential of integrating a GPT-4 driven multi-agent deep reinforcement learning strategy within the federated learning framework, promising to further enhance the adaptability and efficiency of distributed machine learning models.

CONFLICT OF INTEREST

The author declares that they have no relevant conflicts of interest.

REFERENCES

[1] D. Wang and M. Hu, "Deep deterministic policy gradient with compatible critic network," *IEEE Trans. Neural Networks Learn. Syst.*, 2021.

[2] D. Wang, "Meta reinforcement learning with hebbian learning," in *Proc. 2022 IEEE 13th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, 2022, pp. 52–58.

[3] D. Wang, M. Hu, and Y. Gao, "Multi-criteria mission planning for a solar-powered multi-robot system," in *Proc. the ASME Design Engineering Technical Conference*, 2018, vol. 2A-2018. doi: 10.1115/DETC2018-85683

[4] D. Wang, "Reinforcement learning for combinatorial optimization," in *Encyclopedia of Data Science and Machine Learning*, IGI Global, 2023, pp. 2857–2871.

[5] L. Yun, D. Wang, and L. Li, "Explainable multi-agent deep reinforcement learning for real-time demand response towards sustainable manufacturing," *Appl. Energy*, vol. 347, 121324, 2023.
[6] D. Wang, J. Zhao, M. Han, and L. Li. (2023). 4D printing-enabled circular economy: Disassembly sequence planning using reinforcement learning. *SSRN*. [Online]. Available: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=4429186
[7] D. Wang, "Explainable deep reinforcement learning for knowledge graph reasoning," in *Recent Developments in Machine and Human Intelligence*, IGI Global, 2023, pp. 168–183.
[8] D. Wang, "An adversarial-robust graph representation learning for energy-aware vehicle routing considering privacy," in *Proc. 2024 8th International Conference on Green Energy and Applications (ICGEA)*, 2024, pp. 80–86.
[9] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning transferable architectures for scalable image recognition," in *Proc. the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8697–8710.
[10] B. Zoph and Q. V. Le, "Neural architecture search with reinforcement learning," arXiv preprint, arXiv1611.01578, 2016.
[11] M. Tan *et al.*, "Mnasnet: Platform-aware neural architecture search for mobile," in *Proc. the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 2820–2828.
[12] A. Gordon *et al.*, "Morphnet: Fast & simple resource-constrained structure learning of deep networks," in *Proc. the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 1586–1595.
[13] D. Wang, "Robust adversarial deep reinforcement learning," in *Deep Learning, Reinforcement Learning, and the Rise of Intelligent Systems*, IGI Global, 2024, pp. 106–125.
[14] J. Yuan *et al.*, "Federated neural architecture search," arXiv preprint, arXiv2002.06352, 2020.
[15] H. Zhu, H. Zhang, and Y. Jin, "From federated learning to federated neural architecture search: A survey," *Complex Intell. Syst.*, vol. 7, no. 2, pp. 639–657, 2021.
[16] J. Liu, J. Yan, H. Xu, Z. Wang, J. Huang, and Y. Xu, "Finch: Enhancing federated learning with hierarchical neural architecture search," *IEEE Trans. Mob. Comput.*, 2023.
[17] Y. Venkatesha, Y. Kim, H. Park, and P. Panda, "Divide-and-conquer the NAS puzzle in resource-constrained federated learning systems," *Neural Networks*, vol. 168, pp. 569–579, 2023.
[18] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial Intelligence and Statistics*, 2017, pp. 1273–1282.
[19] J. Huang, B. Xue, Y. Sun, and M. Zhang, "A flexible variable-length particle swarm optimization approach to convolutional neural network architecture design," in *Proc. 2021 IEEE Congress on Evolutionary Computation (CEC)*, 2021, pp. 934–941.
[20] B. Wang, Y. Sun, B. Xue, and M. Zhang, "Evolving deep convolutional neural networks by variable-length particle swarm optimization for image classification," in *Proc. 2018 IEEE Congress on Evolutionary Computation (CEC)*, 2018, pp. 1–8.
[21] J. Achiam *et al.*, "GPT-4 technical report," arXiv preprint, arXiv2303.08774, 2023.
[22] M. Zheng *et al.*, "Can GPT-4 perform neural architecture search?" arXiv preprint, arXiv2304.10970, 2023.
[23] A. Gupta *et al.*, "RAG vs Fine-tuning: Pipelines, tradeoffs, and a case study on agriculture," arXiv preprint, arXiv2401.08406, 2024.
[24] D. Wang, "Obstacle-aware simultaneous task and energy planning with ordering constraints," in *Proc. 2023 11th International Conference on Information and Communication Technology (ICoICT)*, 2023, pp. 289–294.
[25] K. N. Pujari, S. S. Miriyala, P. Mittal, and K. Mitra, "Better wind forecasting using evolutionary neural architecture search driven green deep learning," *Expert Syst. Appl.*, vol. 214, 119063, 2023.
[26] K. Kandasamy, W. Neiswanger, J. Schneider, B. Poczos, and E. P. Xing, "Neural architecture search with bayesian optimisation and optimal transport," *Adv. Neural Inf. Process. Syst.*, vol. 31, 2018.
[27] H. Liu, K. Simonyan, and Y. Yang, "Darts: Differentiable architecture search," arXiv preprint, arXiv1806.09055, 2018.
[28] H. Pham, M. Guan, B. Zoph, Q. Le, and J. Dean, "Efficient neural architecture search via parameters sharing," in *Proc. International Conference on Machine Learning*, 2018, pp. 4095–4104.
[29] Y. Sun, B. Xue, M. Zhang, and G. G. Yen, "A particle swarm optimization-based flexible convolutional autoencoder for image classification," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 30, no. 8, pp. 2295–2309, 2018.
[30] F. E. F. Junior and G. G. Yen, "Particle swarm optimization of deep neural networks architectures for image classification," *Swarm Evol. Comput.*, vol. 49, pp. 62–74, 2019.
[31] P. M. Mammen, "Federated learning: Opportunities and challenges," arXiv preprint, arXiv2101.05428, 2021.
[32] H. Zhang, T. Wu, S. Cheng, and J. Liu, "Fedcos: A scene-adaptive enhancement for federated learning," *IEEE Internet Things J.*, vol. 10, no. 5, pp. 4545–4556, 2022.
[33] Q. Li, B. He, and D. Song, "Model-contrastive federated learning," in *Proc. the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 10713–10722.
[34] T. Nishio and R. Yonetani, "Client selection for federated learning with heterogeneous resources in mobile edge," in *Proc. ICC 2019, 2019 IEEE International Conference on Communications (ICC)*, 2019, pp. 1–7.
[35] B. Luo, X. Li, S. Wang, J. Huang, and L. Tassiulas, "Cost-effective federated learning design," in *Proc. IEEE INFOCOM 2021, IEEE Conference on Computer Communications*, 2021, pp. 1–10.
[36] S. Park, Y. Suh, and J. Lee, "FedPSO: Federated learning using particle swarm optimization to reduce communication costs," *Sensors*, vol. 21, no. 2, 600, 2021.
[37] D. Wang, M. Hu, and J. D. Weir, "Simultaneous task and energy planning using deep reinforcement learning," *Inf. Sci. (Ny).*, 2022.
[38] D. Wang and M. Hu, "Contrastive learning methods for deep reinforcement learning," *IEEE Access*, 2023.
[39] D. Wang, "Out-of-distribution detection with confidence deep reinforcement learning," in *Proc. 2023 International Conference on Communications, Computing, Cybersecurity, and Informatics (CCCI)*, 2023, pp. 1–7.
[40] S. Bubeck *et al.*, "Sparks of artificial general intelligence: Early experiments with GPT-4. arXiv," arXiv preprint, arXiv2303.12712, 2023.
[41] D. Wang, "Multi-agent reinforcement learning for safe driving in on-ramp merging of autonomous vehicles," in *Proc. 2024 14th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*, 2024, pp. 644–651.