Large Language Model Assisted Adversarial Robustness Neural Architecture Search

Rui Zhong*, Yang Cao
Graduate School of Information Science and Technology
Hokkaido University
Sapporo, Japan
{rui.zhong.u5, yang.cao.y4}@elms.hokudai.ac.jp

Jun Yu
Institute of Science and Technology
Niigata University
Niigata, Japan
yujun@ie.niigata-u.ac.jp

Masaharu Munetomo
Information Initiative Center
Hokkaido University
Sapporo, Japan
munetomo@iic.hokudai.ac.jp

***Abstract*—Motivated by the potential of large language models (LLMs) as optimizers for solving combinatorial optimization problems, this paper proposes a novel LLM-assisted optimizer (LLMO) to address adversarial robustness neural architecture search (ARNAS), a specific application of combinatorial optimization. We design the prompt using the standard CRISPE framework (i.e., Capacity and Role, Insight, Statement, Personality, and Experiment). In this study, we employ Gemini, a powerful LLM developed by Google. We iteratively refine the prompt, and the responses from Gemini are adapted as solutions to ARNAS instances. Numerical experiments are conducted on NAS-Bench-201-based ARNAS tasks with CIFAR-10 and CIFAR-100 datasets. Six well-known meta-heuristic algorithms (MHAs) including genetic algorithm (GA), particle swarm optimization (PSO), differential evolution (DE), and its variants serve as baselines. The experimental results confirm the competitiveness of the proposed LLMO and highlight the potential of LLMs as effective combinatorial optimizers. The source code of this research can be downloaded from https://github.com/RuiZhong961230/LLMO.**

***Index Terms*—Large Language Model (LLM), Adversarial Attack, Neural Architecture Search (NAS), Combinatorial Optimizer**

I. Introduction

In recent years, the remarkable advancements in deep learning have been significantly affected and decelerated by the vulnerability of neural networks (NNs) to adversarial attacks [1], [2]. These attacks, which involve imperceptible perturbations to input data, can lead to highly incorrect predictions [3]. Consequently, the deployment and configuration of NNs in security-critical applications face serious challenges [4], [5]. As the complexity of these attacks continues to evolve, the imperative to develop robust neural architectures has never been more pressing.

In the meantime, neural architecture search (NAS) has emerged as a potential and powerful tool for designing NNs automatically [6]–[9]. This technique aims to identify optimal architectures that meet specified performance criteria. However, traditional NAS methods focus primarily on mainstream metrics like accuracy, latency, and model size while the negative effects caused by the adversarial attack are commonly neglected [10]–[12]. Therefore, in the rapid development of the artificial intelligence (AI) community, the adversarial robustness of designed NNs should be considered as a crucial factor, that advances adversarial robustness NAS (ARNAS) techniques.

Although ARNAS tasks are significantly more complex and challenging than traditional NAS tasks, they are fundamentally combinatorial optimization problems. These can be effectively addressed by approximation optimizers with limited CPU time [13]. Meanwhile, large language models (LLMs), such as Gemini [14], GPT [15], [16], and LLaMA [17], [18], have demonstrated exceptional capabilities in understanding and generating human-like text, solving complex problems, and supporting various domains in AI research [19], [20]. The potential of LLMs as optimizers has also been explored: Yang et al. [21] proposed Optimization by PROmpting (OPRO), where the traveling salesman problem (TSP) is described in natural language and used as input for the LLM to generate new solutions. Liu et al. [22] introduced LLM-driven EA (LMEA), which uses LLMs as evolutionary combinatorial optimizers with minimal domain knowledge and no additional training required. In LMEA, the LLM selects parent solutions from the current population in each generation, applies search operators to generate offspring, and employs a self-adaptive selection mechanism to ensure the survival of elite solutions. Brahmachary et al. [23] presented the language-model-based evolutionary optimizer (LEO), a novel population-based optimizer using LLMs to solve numerical optimization problems, including industrial engineering challenges such as supersonic nozzle shape optimization, heat transfer, and windfarm layout optimization. Given these findings, the motivation of this study is to utilize LLMs as combinatorial optimizers to solve ARNAS tasks, potentially accelerating optimization convergence in the search for ARNAS.

This paper proposes a novel LLM-assisted optimizer (LLMO) for addressing ARNAS tasks. Using the standard prompt engineering method CRISPE framework (i.e., Capacity and Role, Insight, Statement, Personality, and Experiment) [24], we instruct the LLM Gemini to iteratively search for the optimal architecture under adversarial attacks. Numerical experiments are conducted on the NAS-Bench-201-based search space with CIFAR-10 and CIFAR-100 datasets [25]. Six well-known discrete meta-heuristic algorithms (MHAs) are employed as competitor algorithms to demonstrate the feasibility and effectiveness of LLMO.

The rest of this paper is organized as follows. Section II introduces the related works including the CRISPE framework and ARNAS. Section III details our proposed LLMO, Section IV describes experimental settings and results, we analyze the performance of LLMO in Section V, and finally, Section VI concludes our work.

II. Related works

A. CRISPE framework

Many researchers have recognized the crucial role of prompts in guiding LLMs to produce satisfactory responses [26], [27]. Consequently, prompt engineering has rapidly emerged as an essential discipline, leading to the development of various techniques such as zero-shot, one-shot, few-shot prompting, and chain-of-thought prompting. This paper adopts the CRISPE framework, a structured prompt engineering approach, to design effective prompts. Here, Fig. 1 presents the components in the CRISPE framework. The implications of components are explained as follows:

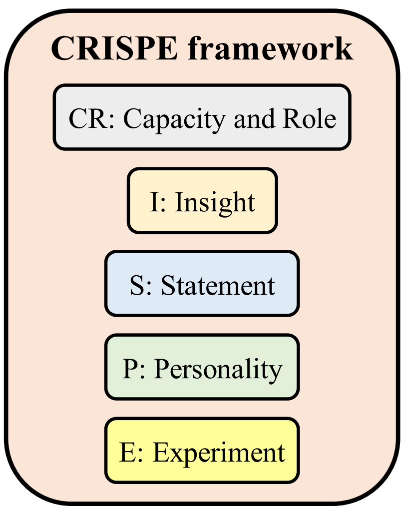

Fig. 1: Components in the CRISPE framework.

- CR: Define the capacity and role.
- I: Provide necessary background or insight.
- S: State the core of the prompt.
- P: Define the fashion of the LLM's response.
- E: Ask for multiple responses.

B. Adversarial robustness neural architecture search (ARNAS)

The ARNAS instances presented in [25] are employed as the benchmark suite in this study, where the structures are demonstrated in Fig. 2. A cell in Fig. 2 consists of four nodes (i.e., feature maps) and six edges (i.e., possible operations). Available operations contain 1x1 convolutions, 3x3 convolutions, 3x3 average pooling, skip connection, and zeroize connection. Consequently, the possible combinations of potential architectures are $5^6 = 15,625$, among them $6,466$ architectures are non-isomorphic.

Here, four representative adversarial attack methods are adopted: the fast gradient sign method (FGSM) [28], projected gradient descent (PGD) [29], adaptive PGD (APGD) [30], and square attack [31]. Detailed descriptions of these methods can be found in corresponding papers.

III. Our proposal: LLMO

A demonstration of the proposed LLMO is presented in Fig. 3[1]. In the initial step, we design and input the prompt using the CRISPE framework into Gemini. The response from Gemini is then refined as a solution to the ARNAS instance. We utilize the prediction accuracy of the feedback to automatically update the prompt. These processes are repeated until the optimization is complete.

Additionally, the overview of the designed prompt is summarized as follows. The contents within "{}" in the prompt will be replaced by the real data during optimization.

- CR: Act as a combinatorial optimizer for adversarial robustness neural architecture search.
- I: The objective of this task is to maximize the accuracy.
- S: There are {number of operations} possible operations and {number of edges} edges that need to be deployed. You need to specify a {number of edges}-bit array where the value in each index is an integer within [0, {number of operations}). The current best solution is {best solution} with the best accuracy {best accuracy}.
- P: Not applicable.
- E: Give me one solution in the array-like format.

In summary, the pseudocode of the proposed LLMO is presented in Algorithm 1. The proposed LLMO offers a significant design advantage: users do not need to understand the working mechanism of Gemini or the specific design of search operators. This means that even amateurs with no prior knowledge of evolutionary computation (EC) and ARNAS can easily use LLMO for optimization. This user-friendly approach ensures easy implementation and accessibility.

Algorithm 1 LLMO

Require: Max. iteration:T
Ensure: Optimum: $\boldsymbol{x}_{best}^t$
Randomly initialize the solution $\boldsymbol{x}_{best}^t$
Evaluate $\boldsymbol{x}_{best}^t$ by ARNAS instance
$t = 0$
while $t < T$ **do**
  Construct prompt using CRISPE framework
  Input prompt to Gemini
  Check the feasibility of the responded solution
  Update the current best solution $\boldsymbol{x}_{best}^t$
  $t \leftarrow t + 1$
end while
return $\boldsymbol{x}_{best}^t$

[1]The symbol of Gemini is downloaded from https://pixabay.com/illustrations/chip-ai-artificial-intelligence-8530784 as a copyright-free image.

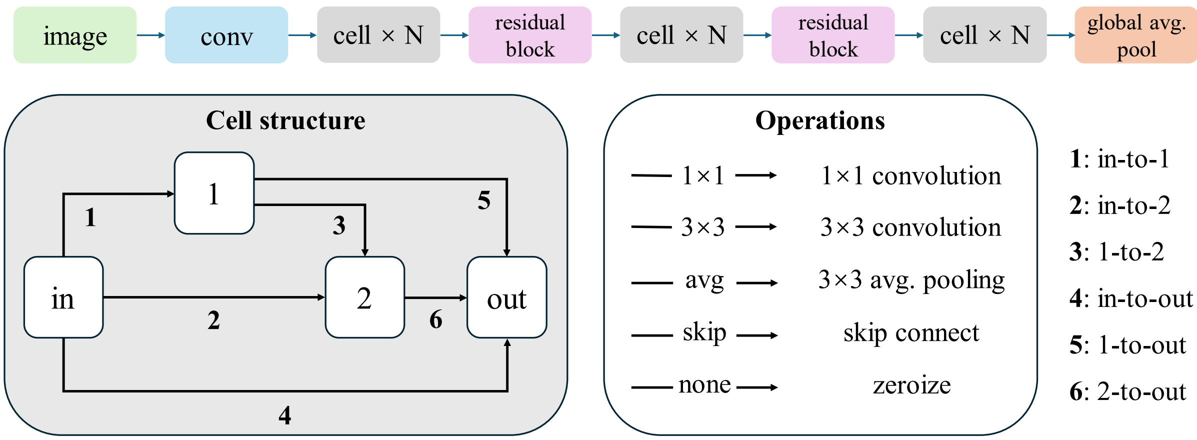

Fig. 2: The architecture of NAS-Bench-201-based ARNAS search space.

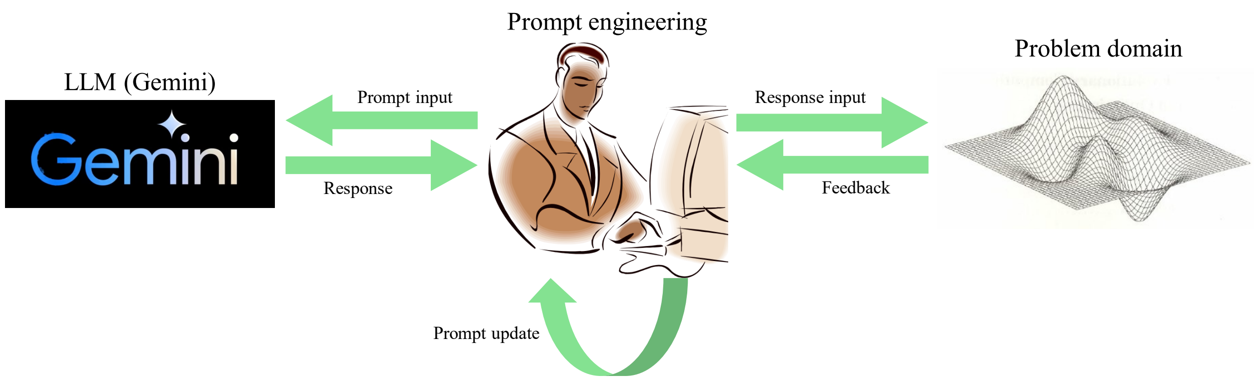

Fig. 3: A demonstration of LLMO.

IV. Numerical Experiments

This section introduces the numerical experiments on ARNAS instances to investigate the performance of LLMO competing with well-known MHAs. Section IV-A presents the detailed experimental settings and Section IV-B summarizes the detailed experimental results.

A. *Experimental settings*

We first present the theoretical optimal prediction accuracy of ARNAS instances with the CIFAR-10 and CIFAR-100 datasets in Table I. Six metaheuristic algorithms (MHAs) are employed as competitor algorithms: genetic algorithm (GA) [32], particle swarm optimization (PSO) [33], differential evolution (DE) [34], evolution strategy with covariance matrix adaptation (CMA-ES) [35], JADE [36], and success-history-adaptive DE (SHADE) [37]. The parameters of these algorithms are summarized in Table II. The population size of competitor algorithms is fixed at 30 while LLMO is a single solution based optimization approach. The maximum fitness evaluation (FE) of all optimizers is fixed at 30 and 3000, respectively. To alleviate the effect of randomness, each algorithm is implemented in 30 trial runs.

TABLE I: Summary of the optimal accuracy in the ARNAS benchmark.

Attack method	Optimum in CIFAR-10	Optimum in CIFAR-100
Clean (No attack)	94.6	73.6
FGSM	69.2	29.4
PGD	58.8	29.8
APGD	54.0	26.3
Square	73.6	40.4

B. *Experimental results*

The experimental results on the ARNAS tasks are summarized in Table III, and the convergence curves are presented in Figs. 4 and 5.

V. Discussion

The experimental results presented in Table III and Fig. 4 confirm the competitiveness of LLMO, particularly in NAS without adversarial attacks and ARNAS with FGSM attacks in both CIFAR-10 and CIFAR-100 datasets. In these four instances, our proposed LLMO outperforms the competitor algorithms, demonstrating the potential and effectiveness of LLM as an optimizer.

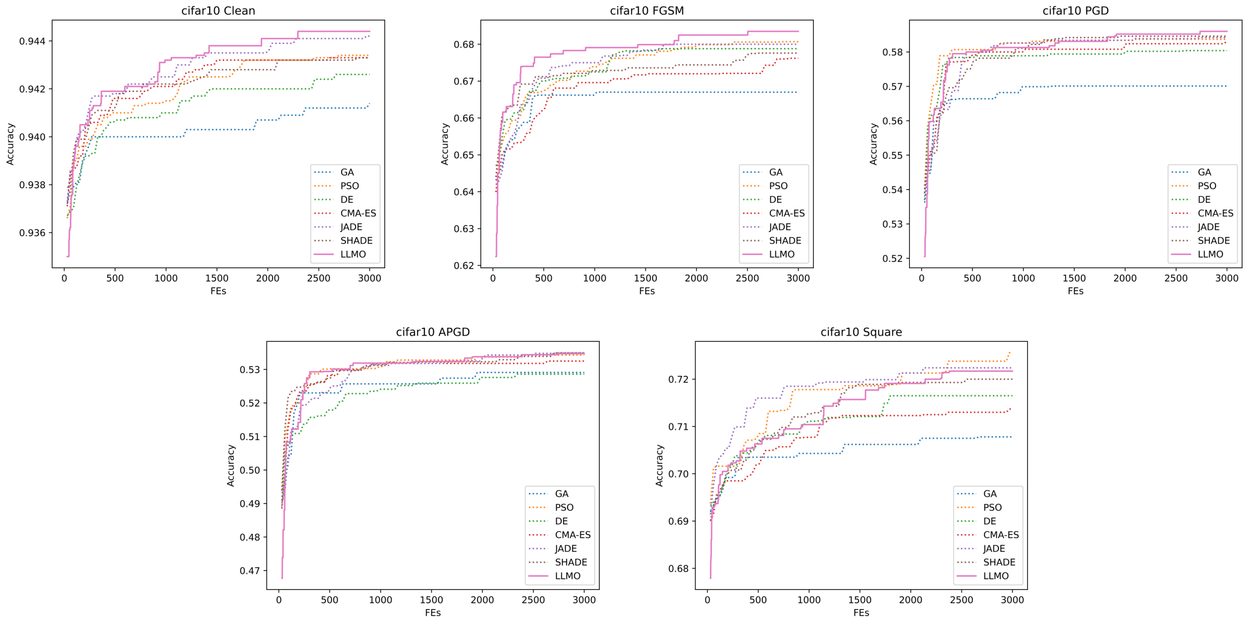

Fig. 4: Convergence curves of optimizers for ARNAS on CIFAR-10.

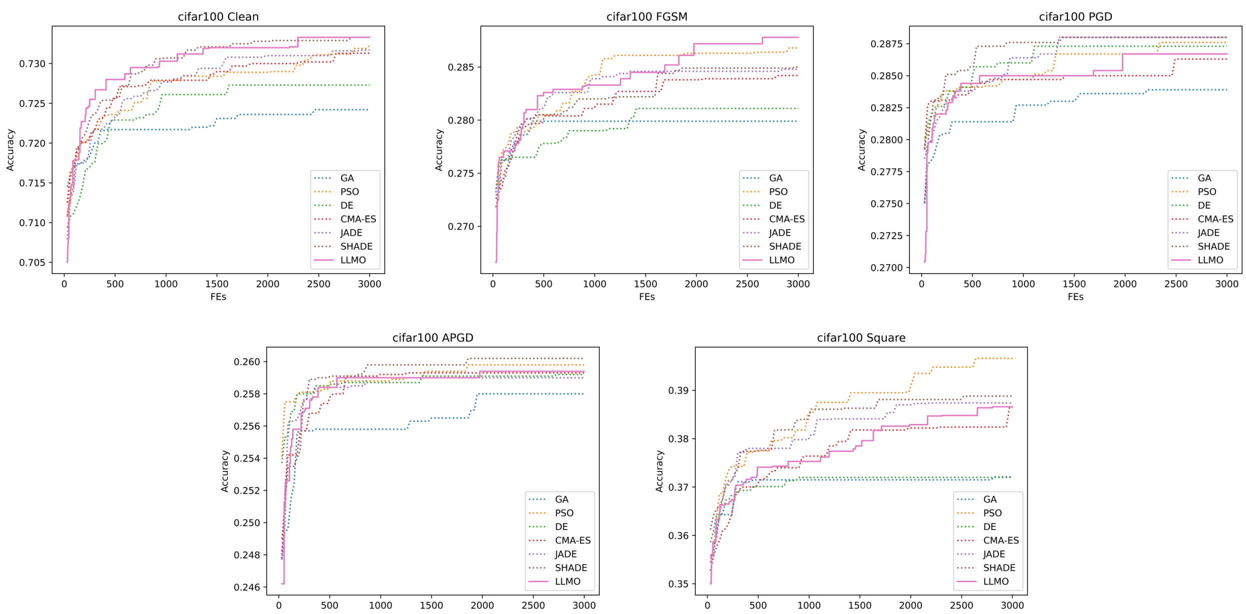

Fig. 5: Convergence curves of optimizers for ARNAS on CIFAR-100.

Additionally, while GA was originally designed for binary optimization problems, and PSO, DE, CMA-ES, JADE, and SHADE were designed for continuous optimization problems, these algorithms require transfer functions to convert the search domain. The use of transfer functions can lead to different solutions in the original search domain being mapped to identical solutions in the transferred search domain, which deteriorates the quality of constructed offspring individuals and reduces search efficiency. However, this issue does not exist in the proposed LLMO. In the prompt design, the constructed offspring individuals are directly encoded as a number of edges-bit array, where the value in each index is an integer within [0, number of operations). This direct encoding method for the generation of offspring individuals is more efficient than the approach used by MHAs that rely on transfer functions.

Furthermore, this research reveals the potential of LLMO in solving combinatorial optimization problems, and we believe

TABLE II: The parameters of competitor algorithms.

Algorithms	Parameters	Value
GA	crossover probability pc	0.9
	mutation probability pm	0.01
	selection	tournament
PSO	inertia factor w	1
	coefficients c_1 and c_2	2.05
	max. and min. speed	2 and -2
DE	mutation strategy	DE/cur-to-rand/1/bin
	scaling factor F	0.8
	crossover rate Cr	0.9
CMA-ES	σ	1.3
JADE	μ_F and μ_{Cr}	0.5 and 0.5
SHADE	μ_F and μ_{Cr}	0.5 and 0.5

that it can further adapt to various combinatorial domains such as feature selection [38], job scheduling [39], and portfolio management problems [40].

VI. Conclusion

Motivated by the ability of LLMs to solve combinatorial optimization problems such as TSP, this paper proposes a novel LLM-assisted optimizer (LLMO) to address adversarial robustness neural architecture search (ARNAS) tasks. We design the prompt using the standard CRISPE framework and iteratively refine it during optimization. The experimental results confirm the competitiveness of LLMO and highlight the potential of LLMs as effective optimizers for solving combinatorial optimization problems.

In future research, we will continue to explore the optimization capacity of LLMs in various optimization domains.

VII. Acknowledgement

This work was supported by JSPS KAKENHI Grant Number 21A402 and JST SPRING Grant Number JPMJSP2119.

References

[1] J. Liu, R. Cheng, and Y. Jin, "Bi-fidelity evolutionary multiobjective search for adversarially robust deep neural architectures," *Neurocomputing*, vol. 550, p. 126465, 2023.

[2] L. Hsiung, Y.-Y. Tsai, P.-Y. Chen, and T.-Y. Ho, "Towards compositional adversarial robustness: Generalizing adversarial training to composite semantic perturbations," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2023, pp. 24 658–24 667.

[3] S. Huang, Z. Lu, K. Deb, and V. N. Boddeti, "Revisiting residual networks for adversarial robustness," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2023, pp. 8202–8211.

[4] J. Su, D. V. Vargas, and K. Sakurai, "One pixel attack for fooling deep neural networks," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 5, pp. 828–841, 2019.

[5] R. Wang, Y. Li, and S. Liu, "Exploring diversified adversarial robustness in neural networks via robust mode connectivity," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2023, pp. 2346–2352.

[6] J. Mellor, J. Turner, A. Storkey, and E. J. Crowley, "Neural architecture search without training," in *Proceedings of the 38th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, M. Meila and T. Zhang, Eds., vol. 139. PMLR, 18–24 Jul 2021, pp. 7588–7598.

[7] Y. Xue, C. Chen, and A. Słowik, "Neural architecture search based on a multi-objective evolutionary algorithm with probability stack," *IEEE Transactions on Evolutionary Computation*, vol. 27, no. 4, pp. 778–786, 2023.

[8] X. Chu, S. Lu, X. Li, and B. Zhang, "Mixpath: A unified approach for one-shot neural architecture search," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2023, pp. 5972–5981.

[9] Z. Lu, R. Cheng, Y. Jin, K. C. Tan, and K. Deb, "Neural architecture search as multiobjective optimization benchmarks: Problem formulation and performance assessment," *IEEE Transactions on Evolutionary Computation*, vol. 28, no. 2, pp. 323–337, 2024.

[10] C. Devaguptapu, D. Agarwal, G. Mittal, P. Gopalani, and V. N. Balasubramanian, "On adversarial robustness: A neural architecture search perspective," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, October 2021, pp. 152–161.

[11] R. Hosseini, X. Yang, and P. Xie, "Dsrna: Differentiable search of robust neural architectures," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021, pp. 6196–6205.

[12] L. Bortolussi, G. Carbone, L. Laurenti, A. Patane, G. Sanguinetti, and M. Wicker, "On the robustness of bayesian neural networks to adversarial attacks," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–14, 2024.

[13] R. Zhong, Y. Xu, C. Zhang, and J. Yu, "Efficient multiplayer battle game optimizer for adversarial robust neural architecture search," 2024.

[14] G. Team, R. Anil, S. Borgeaud, Y. Wu, J.-B. Alayrac, J. Yu, R. Soricut, J. Schalkwyk, A. M. Dai, A. Hauth *et al.*, "Gemini: A family of highly capable multimodal models," *arXiv preprint arXiv:2312.11805*, 2023.

[15] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., vol. 33. Curran Associates, Inc., 2020, pp. 1877–1901.

[16] OpenAI, "Gpt-4 technical report," 2023.

[17] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, A. Rodriguez, A. Joulin, E. Grave, and G. Lample, "Llama: Open and efficient foundation language models," 2023.

[18] P. Gao, J. Han, R. Zhang, Z. Lin, S. Geng, A. Zhou, W. Zhang, P. Lu, C. He, X. Yue, H. Li, and Y. Qiao, "Llama-adapter v2: Parameter-efficient visual instruction model," 2023.

[19] A. Thirunavukarasu, D. Ting, K. Elangovan, L. Gutierrez, T. Tan, and D. Ting, "Large language models in medicine," *Nature Medicine*, 07 2023.

[20] R. Zhong, Y. Xu, C. Zhang, and J. Yu, "Leveraging large language model to generate a novel metaheuristic algorithm with crispe framework," 2024.

[21] C. Yang, X. Wang, Y. Lu, H. Liu, Q. V. Le, D. Zhou, and X. Chen, "Large language models as optimizers," 2024.

[22] S. Liu, C. Chen, X. Qu, K. Tang, and Y.-S. Ong, "Large language models as evolutionary optimizers," 2024.

[23] S. Brahmachary, S. M. Joshi, A. Panda, K. Koneripalli, A. K. Sagotra, H. Patel, A. Sharma, A. D. Jagtap, and K. Kalyanaraman, "Large language model-based evolutionary optimizer: Reasoning with elitism," 2024.

[24] M. Wang, M. Wang, X. Xu, L. Yang, D. Cai, and M. Yin, "Unleashing chatgpt's power: A case study on optimizing information retrieval in flipped classrooms via prompt engineering," *IEEE Transactions on Learning Technologies*, pp. 1–13, 2023.

TABLE III: The experimental results of prediction accuracy in the ARNAS benchmark.

Prob.		GA	PSO	DE	CMA-ES	JADE	SHADE	LLMO
CIFAR-10	Clean	94.14	94.34	94.26	94.33	94.42	94.33	**94.44**
	FGSM	66.70	68.07	67.88	67.62	68.00	67.76	**68.35**
	PGD	57.01	58.37	58.04	58.32	58.42	58.46	**58.60**
	APGD	52.91	53.43	52.86	53.25	**53.50**	53.49	53.47
	Squares	70.78	**72.56**	71.65	71.37	72.24	72.00	72.17
CIFAR-100	Clean	72.42	73.23	72.73	73.13	73.17	73.33	**73.33**
	FGSM	27.99	28.68	28.11	28.42	28.48	28.50	**28.78**
	PGD	28.39	28.76	28.73	28.63	28.80	**28.80**	28.67
	APGD	25.80	25.98	25.92	25.93	25.90	**26.02**	25.94
	Squares	37.21	**39.66**	37.20	38.64	38.74	38.88	38.66

[25] S. Jung, J. Lukasik, and M. Keuper, "Neural architecture design and robustness: A dataset," in *ICLR*, 2023.

[26] J. Wei, X. Wang, D. Schuurmans, M. Bosma, b. ichter, F. Xia, E. Chi, Q. V. Le, and D. Zhou, "Chain-of-thought prompting elicits reasoning in large language models," in *Advances in Neural Information Processing Systems*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, Eds., vol. 35. Curran Associates, Inc., 2022, pp. 24 824–24 837.

[27] J. White, Q. Fu, S. Hays, M. Sandborn, C. Olea, H. Gilbert, A. Elnashar, J. Spencer-Smith, and D. C. Schmidt, "A prompt pattern catalog to enhance prompt engineering with chatgpt," 2023.

[28] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," 2015.

[29] A. Kurakin, I. Goodfellow, and S. Bengio, "Adversarial machine learning at scale," 2017.

[30] F. Croce and M. Hein, "Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks," 2020.

[31] M. Andriushchenko, F. Croce, N. Flammarion, and M. Hein, "Square attack: a query-efficient black-box adversarial attack via random search," 2020.

[32] J. H. Holland, "Genetic algorithms," *Scientific American*, vol. 267, no. 1, pp. 66–73, 1992.

[33] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95 - International Conference on Neural Networks*, vol. 4, 1995, pp. 1942–1948 vol.4.

[34] R. Storn, "On the usage of differential evolution for function optimization," in *Proceedings of North American Fuzzy Information Processing*, 1996, pp. 519–523.

[35] N. Hansen, S. D. Müller, and P. Koumoutsakos, "Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (cma-es)," *Evolutionary Computation*, vol. 11, no. 1, pp. 1–18, 2003.

[36] J. Zhang and A. C. Sanderson, "Jade: Adaptive differential evolution with optional external archive," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 5, pp. 945–958, 2009.

[37] R. Tanabe and A. Fukunaga, "Success-history based parameter adaptation for differential evolution," in *2013 IEEE Congress on Evolutionary Computation*, 2013, pp. 71–78.

[38] Z. Rui, Z. Chao, and Y. Jun, "Chaotic vegetation evolution: leveraging multiple seeding strategies and a mutation module for global optimization problems," *Evolutionary Intelligence*, pp. 1–25, 01 2024.

[39] M. Xu, Y. Mei, F. Zhang, and M. Zhang, "Niching genetic programming to learn actions for deep reinforcement learning in dynamic flexible scheduling," *IEEE Transactions on Evolutionary Computation*, pp. 1–1, 2024.

[40] Y. Ma, W. Wang, and Q. Ma, "A novel prediction based portfolio optimization model using deep learning," *Computers & Industrial Engineering*, vol. 177, p. 109023, 2023.