Communication Interchange For Artificial Intelligence Systems

Razvan Cristian Voicu[†], Aarush Kaunteya Pande[†], Muhammad Hassan Tanveer, Yusun Chang
Department of Robotics & Mechatronics Engineering, Kennesaw State University, Atlanta, Georgia
Email: voicu@gatech.edu, aarush@gatech.edu, mtanveer@kennesaw.edu, ychang7@kennesaw.edu,

***Abstract*—The rise and proliferation of Artificial Intelligence (AI) technologies are bringing transformative changes to various sectors, signaling a new era of innovation in fields as diverse as medicine, manufacturing, and even day-to-day social interactions. Notable advancements are not just confined to textual understanding, as seen in models like GPT, but also extend to visual cognition through image recognition and more. Beyond surface interactions and predictions, AI finds profound applications in life-saving domains such as medical diagnostics and becomes an integral part of daily life through chatbot-based customer interactions. However, as the horizon of AI expands, a crucial yet often overlooked aspect emerges— the underlying mission-critical infrastructure required to support and deploy these models effectively. The intricacies of efficient communication systems, foundational for real-time AI model operations, take center stage in ensuring the seamless functioning of AI-driven applications. This paper explores the quintessential changes needed in communication paradigms to keep pace with the evolving AI landscape. Specifically, we highlight the pivotal role of multipath communication in enhancing the responsiveness and efficiency of AI applications [1]. As a case in point, we investigate its impact on mission-critical operations in robotics. Through experimentation and analysis, the results elucidate the substantial benefits of this approach, revealing a significant improvement in delay metrics. This work underscores the imperative of aligning communication systems with the ever-growing demands of AI, ensuring that infrastructural capabilities do not lag in the race for innovation.**

***Index Terms*—Artificial Intelligence (AI), Programmable Networking, Parallel Communication, Centralized - Distributed Systems, Real-time Control, Multipath Communication, AI Infrastructure**

I. Introduction

Artificial Intelligence (AI) systems epitomize the convergence of vast computational power and intricate algorithms to simulate human-like cognitive functions. At their core, these systems perceive their environment, make informed decisions, and carry out tasks with varying degrees of autonomy, often surpassing human capabilities in accuracy and speed.

A clear distinction emerges between traditional Machine Learning (ML) models and contemporary AI systems in the evolving technological panorama. Whereas traditional ML models are trained for specific tasks, relying on curated data and explicit feature engineering, the progression toward general knowledge AI paints a different picture. Models like GPT exemplify this shift towards general knowledge AI. These models, equipped with the capability to grasp the underlying intent of requests made to them, stand as testaments to AI's power in generating solutions with unprecedented efficiency and precision. However, this advancement is not without its challenges. Training such a GPT model demands vast datasets, and the model's architectural complexity results in a colossal size. While attempts to miniaturize these models exist, as showcased in [2], they often come at the cost of the model's comprehensive feature set.

[†]*Both Authors Contributed Equally*

With their unparalleled capabilities, AI systems are witnessing a surge in their application spectrum. Their utility spans chatting interfaces, software programming assists, dedicated customer service platforms, and even unique domains like protein generation and genetics. A significant stride in this direction is the exploration of AI as the epicenter for robotic systems. For instance, initiatives from Microsoft research labs have unveiled systems where GPT can autogenerate code, seamlessly translating it to operational commands for robotic drones, manipulators, and beyond.

As the gamut of these applications broadens, a pressing concern surfaces regarding the infrastructure backing these AI systems. While some AI models, tailored for specific tasks, can be hosted locally, comprehensive models, given their size and computational demands, often require access through APIs or dedicated applications. Local hosting of AI behemoths remains challenging for smaller organizations, leading to increased dependency on AI hosting platforms. This dependency raises pressing questions: What implications arise from potential access disruptions due to power outages or communication breakdowns? Moreover, with AI integration in delay-sensitive environments, do existing communication frameworks and infrastructure offer the necessary availability, efficiency, and speed [3], [4]?

The ensuing sections of this paper delve deeper into these concerns. Section II embarks on a literature review. Section III delineates the proposed optimization and the communication model tailored for AI systems. Section IV presents the experimental results and analyses. Finally, Section V concludes the discussion, offering insights into future research directions.

II. Literature Review

As we delve deeper into the convergence of artificial intelligence and robotics, it becomes imperative to contextualize the broader landscape of foundational research and seminal developments, from the initial breakthroughs in AI models to the complexities of their real-world deployment. This section provides an overview of the trajectory of notable AI models, primarily the GPT series, their computational intricacies, and the communication frameworks that underpin them. Additionally, the nuances of integrating these models into the realm of robotics highlight the path toward the next era of interconnected, AI-driven systems.

Machine learning's conception traces back to the mid-20th century, with its foundational roots seen in earlier ideas. In the 1930s and 1940s, Alan Turing postulated the potential for machines to mimic human intelligence, laying a theoretical groundwork for future developments. It was in the 1950s, however, when tangible advancements occurred. The perceptron, introduced by Frank Rosenblatt in 1958, was one of the first algorithms for supervised learning of binary classifiers. The term "machine learning" was coined by Arthur Samuel, who, around the same time, was working on adaptive learning, where the machine would improve based on training data. This era marked a paradigm shift, where algorithms began to be designed not just for computation but for adaptive learning based on data. Over time, as computational capabilities improved and data became more abundant, the frameworks and models evolved, eventually leading to the advanced neural networks and deep learning methodologies prevalent today.

A. Development of GPT Models

The development of large-scale language models, particularly those of the GPT (Generative Pre-trained Transformer) lineage, has its roots in the broader evolution of deep learning and the Transformer architecture. *Vaswani et al.* [5] initially proposed the Transformer architecture, which introduced the concept of self-attention, enabling the model to weigh the importance of different words in a sequence. This architecture laid the foundation for models like BERT [6] and, subsequently, GPT [7].

OpenAI's GPT, or Generative Pre-trained Transformer, introduced a novel two-step approach to training. The initial step involved unsupervised pre-training on a vast corpus, and the second step refined the model using supervised fine-tuning on a narrower dataset [7]. This approach effectively leveraged the vast information in extensive textual data while honing the model's capabilities for specific tasks. The GPT series saw an exponential growth in size. GPT-2, with 1.5 billion parameters, was deemed substantial at its inception [8]. However, GPT-3 dwarfed its predecessor with 175 billion parameters [9] and Microsoft Megatron-Turing NLG with 530 billion parameters [10]. While not explicitly disclosed, the training time for these models is known to span weeks, employing multiple GPUs and TPUs.

Efficient model training necessitates exploiting data parallelism or model parallelism. In data parallelism, data subsets are distributed across multiple processors, each computing gradients for its subset. These gradients then amalgamate across processors. Conversely, separate parts of the neural network reside on different processors in model parallelism. Such distributed training mandates a robust communication medium for gradient, weight, or data exchanges. Tools like NVIDIA's NCCL, the Open Message Passing Interface (OpenMPI), and Microsoft's DeepSpeed [10]–[16] facilitate such exchanges. Microsoft has extensively utilized DeepSpeed in conjunction with OpenMPI to enhance inter-GPU or inter-node communications, offering optimized throughput and scalability. Training such large-scale models is also limited on the backend by the communication efficiency [17]–[19]. Similarly, communication improvements are also necessary at the application level, which this paper presents.

Training large-scale models like GPT-3 incurs significant costs. While exact figures remain proprietary, ballpark estimates place the training cost of GPT-3 in the range of several million dollars, factoring in the computational resources and the electricity required for such intensive training. In addition to the high costs, models like GPT-3 require gargantuan datasets. These datasets are often sourced from myriad online platforms and stored in distributed systems by leveraging protocols like HDFS or GFS, ensuring fault-tolerant storage while accommodating parallel data accesses. Given these steep expenses, the ability to train and maintain such a large-scale model locally becomes infeasible for many small to mid-sized companies. This economic barrier underscores the significance of accessible platforms and third-party services for utilizing advanced machine learning models.

Therefore, many entities, recognizing these challenges, opt for cloud-based access. However, this alternative has its own set of issues.

- **Model Inference and Latency:** GPT models, given their size, necessitate substantial memory and computational resources even during inference. Real-time applications, such as chatbots or on-the-fly code generation, demand rapid response times, which is challenging given the model's complexity.
- **Communication Requirements:** Accessing GPT models remotely, typically through cloud-based APIs, demands robust and low-latency communication channels. The data exchange involves the user's query and the model's context, spanning thousands of tokens for models like GPT-3. Services like OpenAI's API for GPT-3, or analogous platforms, must handle concurrent requests from numerous users, necessitating vast bandwidth and efficient load-balancing mechanisms.

Researchers are looking for solutions through unique ways of combining models, utilizing model compression strategies, or implementing other novel solutions. Techniques such as knowledge distillation and alternative methods, like the one used in Stanford's Alpaca model [2], are also gaining traction. The latter employs fine-tuning on demonstrations generated by powerful models to produce compact yet effective instruction-following models. These adjustments and strategies are essential for creating models that fit the constraints of various deployment scenarios and meet the needs of multiple applications.

However, real-time applications, especially in areas like robotics, demand more than just speed; they necessitate high-quality responses. The smaller, fine-tuned models derived from techniques like knowledge distillation can diminish the response quality because of their limited in-depth parameterization, which is crucial for optimized responses. Therefore, current systems are at a crossroad in catering to these applications. Services like OpenAI's API for GPT-3/4 or similar platforms face the challenge of handling concurrent requests from numerous users requiring vast bandwidth and efficient load-balancing mechanisms. Given the fluctuating bandwidth needs based on the application specifics and frequency of access, ensuring uninterrupted and high-speed communication is challenging for real-time delay-intolerant applications.

B. GPT in Diverse Domains

The versatility of GPT models has paved the way for their application across many environments. One of the pioneering applications lies in robotics, spanning domains such as autonomous driving [20], manufacturing, and assistive technologies. Robots equipped with GPT models can decipher complex instructions, adapt to dynamic environments, and collaborate with humans in shared workspaces.

For instance, Microsoft Research Labs has ventured into translating GPT-generated code directly into operational commands for robotic entities. Such endeavors accentuate the adaptability of GPT models, extending beyond natural language tasks to real-world implementations.

Furthermore, GPT models have made significant financial strides, especially in high-frequency trading, where decision latency is minimal. These models analyze vast swaths of data, from market trends to news articles, and make near-instantaneous trading decisions. The delay-intolerant nature of such applications, where microseconds can translate to significant financial implications, underscores the need for optimized communication infrastructures.

C. Communication Requirements for AI-driven Systems

With the proliferation of AI-centric applications, especially those exhibiting delay-intolerant behaviors, the underlying communication infrastructure becomes a keystone. These applications require diverse setups and configurations depending on their nature and operational constraints.

For instance, robotic systems may differ in their connectivity configurations based on their application. Manufacturing robots often lean towards wired setups, owing to their demands for high-speed, deterministic communication, ensuring minimal latency. Any hint of communication uncertainty can be detrimental, potentially disrupting intricate manufacturing processes. In contrast, robots deployed in exploration, surveillance, or those operating as drones prioritize wireless setups, valuing flexibility and unhindered mobility.

Furthermore, robotic systems may exhibit either centralized or distributed architectures. Centralized systems rely on a singular controller orchestrating the actions of all robotic entities. While this ensures synchronous operations, it poses potential bottlenecks, especially in communication-intensive scenarios. Distributed systems, however, empower each robot or a cluster of robots with decision-making capabilities based on local data. Such setups favor scalability and resilience but introduce complexities ensuring coordinated actions and communication demands.

For robotic systems, communication becomes paramount. The primary controller juggles its responsibilities of querying AI models like GPT for insights while simultaneously dispatching directives to robotic agents [21]. Delays in AI query responses can inadvertently introduce lags in command dispatches, leading to operational inefficiencies. Ongoing communication advancements like parallel communication mechanisms can alleviate such concerns where dedicated communication channels cater to AI model queries while others maintain seamless communication with robotic agents. Multi-path Parallel communication ensures that delays in one link don't impede operations in another, which is especially vital in time-sensitive mission-critical deployments [22], [23].

III. Proposed Solution: Parallel Programmable Networking for AI-Driven Robotic Systems

The paper introduces multiple strategies for enhancing AI-controlled robotic systems. One proposed strategy involves using multi-path parallel communication, designating a specific link for AI queries while maintaining separate links for consistent robotic communication. Furthermore, this work presents a novel methodology employing diverse models to expedite response times. This technique integrates a turbo model for swift responses and a sophisticated model for response optimization rather than solely relying on the more intricate model for outcomes. Such an approach facilitates innovative methods, for instance, transmitting unrefined code to the robot for immediate directives while concurrently strategizing an optimized solution for the ensuing action.

Additionally, the work introduces a buffering mechanism that operates between the robot's immediate directive and its subsequent optimized step. This setup enables the control system to refine the succeeding actions while the robot executes its present task in a distinct processing environment. This design ensures that robots can access optimized commands in real-time, leveraging AI's control capabilities. Furthermore, given their parallel processing capability, this distinct processing framework allows robots to adapt during operations should unforeseen events arise.

The essence of the proposed solution revolves around leveraging the programmable networking paradigm to manage communication channels efficiently, optimizing the synergy between AI-driven components and robotic units.

A. Programmable Parallel Communication

In the evolving landscape of communication, horizontal architectures such as CoopNet have emerged to address the dynamic needs of the current communication era [1], [3], [4], [24], [25]. These architectures permit the exploitation of available communication links, offering dynamic flexibility suited to highly mobile and dynamic environments, especially when unguided mediums come into play [26]. Such adaptability

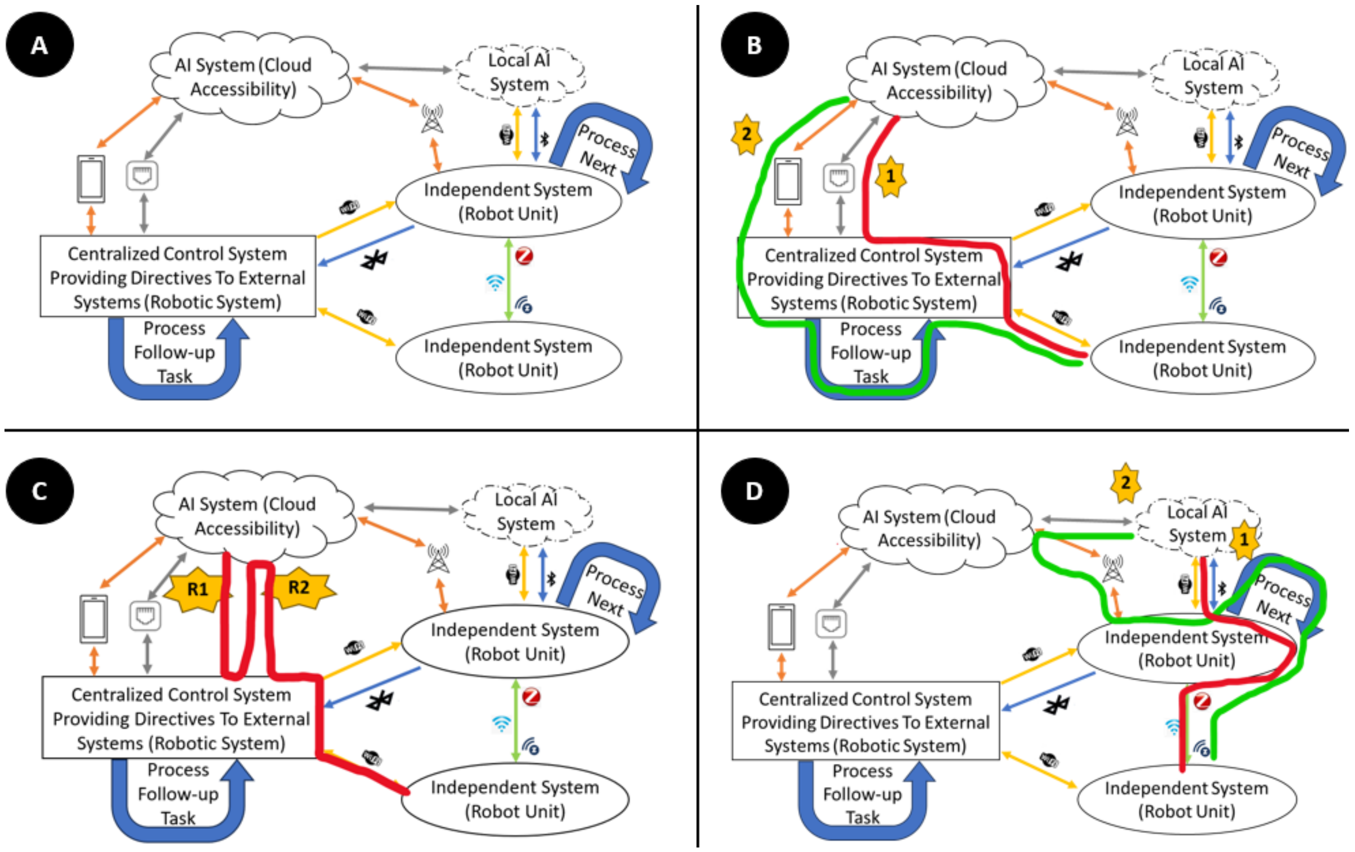

A
B
C
D
AI System (Cloud Accessibility)
Local AI System
Process Next
Independent System (Robot Unit)
Centralized Control System Providing Directives To External Systems (Robotic System)
Process Follow-up Task
1
2
R1
R2

Fig. 1: Example Communication Topology, Profiles, & Programmable Networking Paths

ensures the swift establishment of transient communication channels, fostering enhanced collaboration.

Delving into the horizontal communication landscape, CoopNet emerges as an innovative solution. Crafted as an API-style communication architecture, it addresses the shortcomings of traditional frameworks [3]. By minimizing dependency on complex processes and socket connections, CoopNet streamlines the developer interface, all while maintaining compatibility with existing frameworks. Its strength resides in adeptly orchestrating connection dynamics from initiation to termination. CoopNet's distinguishing feature is its capability to efficiently direct traffic over multiple parallel interfaces and links, enhancing communication efficiency and adaptability. As showcased by CoopNet, programmable networking helps to meet unique application demands. This adaptability facilitates parallel communication threads crafted to cater to real-time needs, spanning domains from AI to robotics. This dynamism is enabled by strategic methodologies, optimizing the allocation of network resources for contemporary challenges.

To grasp the essence of dynamic programmable communication, it's vital to understand its significance in typical robotic applications. Figure 1A presents a conventional topology with various communication links. A centralized system can connect through guided (wired) medium or unguided (wireless) via cellular, WiFi, Bluetooth, and more. Independent units, especially in robotics or IoT, can access other communication mediums, including IoT-specific z-wave, ZigBee, LoraWAN, and more. In a typical setup, a centralized system sends directives to individual robot units based on some existing model or algorithm.

On the other hand, independent units can operate in a distributed mode, leveraging ad-hoc or IoT-driven communication for reactive actions such as obstacle avoidance. However, real-time robotic control through AI systems is an upcoming trend with limited exploration and varying needs, including delay-intolerant processing and communication.

Considering the demands of delay-intolerant systems, programmable parallel networking offers innovative solutions (The red line represents the initial transaction, while the green line shows the subsequent AI directive transaction):

- Figure 1B illustrates a centralized control unit using its quickest and most stable channel for immediate directives while employing a secondary channel for subsequent instructions.
- Figure 1C illustrates a two-tiered method where a turbo-based AI model initially offers a rapid response (R1), followed by a refined output from a comprehensive model (R2). This strategy proves more efficient and optimal than using either model alone.
- Lastly, Figure 1D showcases a scenario where an on-site AI system issues directives. While the local AI can quickly respond, it may not always offer the most optimal solution. Thus, the AI might request input from a more sophisticated model to optimize its decision.

Furthermore, in mission-critical settings, this figure also demonstrates the distributed operation where an AI can reside within an independent unit.

This multi-faceted approach underscores the potential and versatility of programmable communication in dynamic environments.

B. Efficient Access to AI Resources in Centralized Systems

To achieve the desired efficiency in robotic control driven by AI, local centralized and distributed systems must be adept at tapping into local and cloud AI resources based on urgency and complexity. Moreover, all systems would benefit from adeptly harnessing programmable networking features and multipath parallel communication. Thus, these systems can ensure prompt decision-making and adaptive responsiveness across various robotic applications, as contributed in this article, portrayed in Algorithm 1.

C. Algorithm: Dynamic Resource Allocation with Programmable Networking

Algorithm 1 Adaptive AI Resource Access and Dynamic Interface Allocation

```
Require: System initialization and configuration
Ensure: Adaptively prioritize and access AI resources
 1: while system is running do
 2:   monitor_interface_usage()
 3:   find_fastest_idle_link()
 4:   if API_request_pending() then
 5:     set_fastest_link_for_mission_crit()
 6:   else if followup_directive_pending() then
 7:     allocate_secondary_idle_link()
 8:   end if
 9:   if local_controller_needs_AI() then
10:     if is_time_sensitive() then
11:       generate_response_from_local_AI()
12:       send_to_cloud_AI_system()
13:     else
14:       send_to_cloud_AI_system()
15:     end if
16:   end if
17:   if local_AI_available_and_optimized() then
18:     request_from_local_AI()
19:   else if not_optimized_or_unavailable() then
20:     request_from_turbo_model()
21:     if needs_optimization() then
22:       request_best_model_for_optimization()
23:     end if
24:   end if
25:   if directive_from_centralized() then
26:     provide_overview_directive()
27:   else
28:     delegate_to_distributed_units()
29:   end if
30:   balance_load_among_interfaces()
31: end while
```

D. Modeling The System

To further elucidate the performance and efficiency of the proposed system, consider the following model for efficiency calculation:

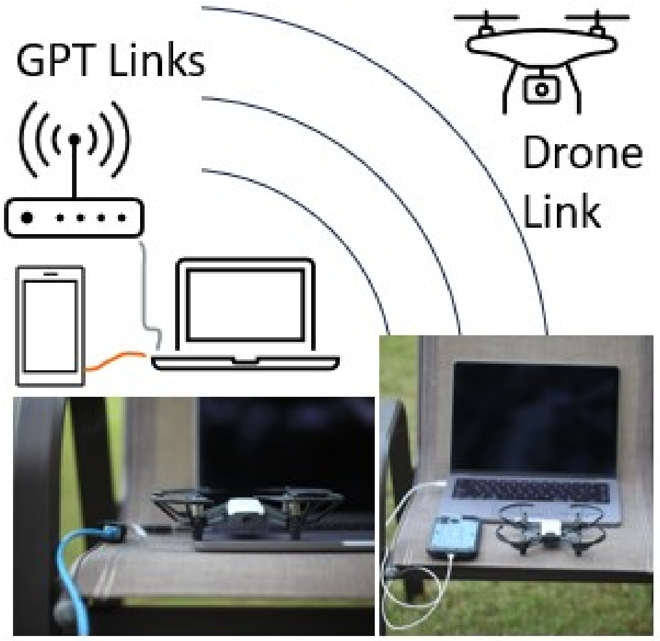


Fig. 2: Experimental Setup Topology

$$E = \frac{\sum_{i=1}^{n} T_i}{T_{max}} \times \frac{R_{api} + R_{local}}{R_{total}} \times K \qquad (1)$$

Where:

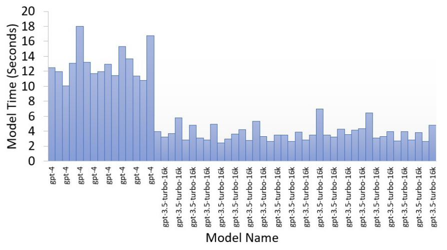
- E: System efficiency.
- T_i: Time for processing i^{th} task.
- T_{max}: Maximum time allowed for task processing to avoid delays.
- R_{api}, R_{local}: Rates of API request and local communication successes.
- R_{total}: Total communication requests.
- K: Scaling factor for load adaptability, based on historical or real-time data.

E. Experiment Topology and Setup

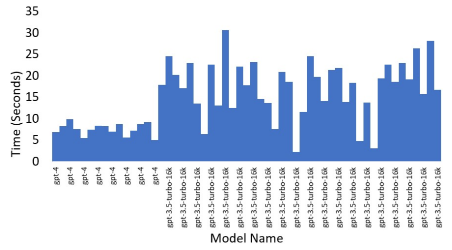
The experimental setup contributes to Microsoft Research's Prompt-Robotics framework [27], focusing on addressing real-time constraints in robotic controls. Figure 2 visually represents the equipment integral to the experimental design. This visual aid further underscores the nuanced communication strategies and the amalgamation of AI-driven directives with robotic execution. Utilizing a programmable educational drone, the system facilitates command execution via wireless communication (WiFi).

The study involves assessing communication and processing delays through several approaches: single interface communication (Cellular or Wired Ethernet) with GPT4 requests, multi-interface (WiFi, Cellular, and Wired Ethernet) GPT4 processing, and an innovative multi-interface parallel GPT3.5 turbo processing enhanced by GPT4 optimization, as demonstrated in Figures 1A, 1B, and 1C. Exploration of local GPT integration, depicted in Figure 1D, remains a topic for future research.

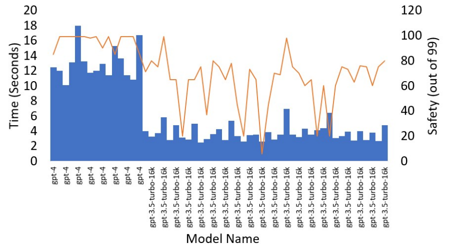
Results indicate the need for multipath parallel communi-



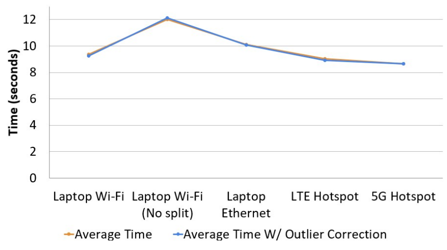
(a) Latency Utilizing Turbo-Opt & Best Models



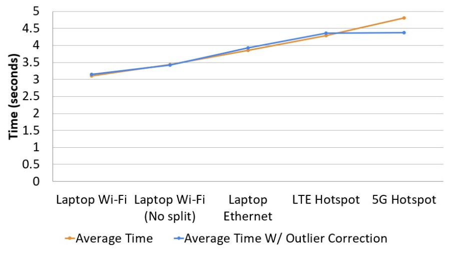
(b) Computed Efficiency



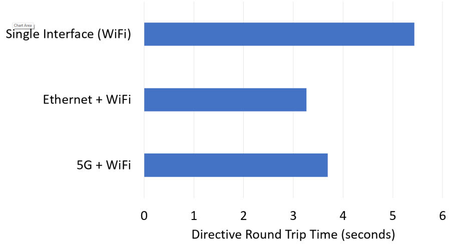
(c) Safety Score



(d) Total Request Time (GPT4)



(e) Total Request Time (GPT3.5 Optimized w/ Safety)



(f) Parallel Communication Improvement

Fig. 3: Optimized Model Access and Processing Improvements to AI Systems Controlling Delay-Intolerant Units

cation for timely responses and robotic control. Nevertheless, they also reveal the necessity for additional advancements, such as model selection and optimization, effective instruction queue processing for immediate yet unoptimized requests, and other strategies to mitigate the currently unsatisfactory delays in real-time applications.

IV. Experimental Data and Analysis

This research evaluates the efficacy of a programmable networking approach, focusing on its ability to manage multiple communication channels simultaneously for AI-enhanced robotic control. The central objective revolves around discerning the effects of this approach on communication and processing delay while preserving the continuous communication that is indispensable for precise robotic operations.

V. Results

Figure 3a showcases the strategic decision-making in AI model deployment. This research emphasizes that the magnitude and intricacy of an AI model don't inherently translate to swifter outcomes. There's a compelling case for employing a more nimble albeit less precise model, like the turbo or local variant. The result demonstrates improved efficacy when a less resource-demanding model's outputs undergo further refinement using a more sophisticated model. The data substantiates this, revealing a reduction in latency by a factor of four, even while maintaining the same optimal response quality as shown in Figure 3ac, including the safety score for correct responses.

Figure 3b sheds light on the transformative efficiency enhancements in robotic control that emerge from applying the algorithm detailed in this research. This structured approach supports rapidly handling mission-critical tasks and meticulous refinement of follow-up directives, utilizing parallel preprocessing techniques and preemptive model access. Adding another dimension to the insights, Figure 3c offers a comparative analysis. It compares the foundational system blueprint with this research's communication and processing advancements. Including a safety check score further amplifies the reliability and precision of the directives. While it's noteworthy that certain instances yield less-than-ideal responses, the structured follow-up directive parallel processing mechanism ensures these minor discrepancies get addressed.

Delving into more specific metrics, Figure 3d presents latency benchmarks across different communication channels. The data underscores a transformative advantage when deploying 5G and Ethernet for handling requests, complemented by an auxiliary channel dedicated to unit control. It's evident that GPT4's request times, despite its prowess, remain substantially elongated compared to the streamlined approach delineated in this research. Figure 3e shows total request timings for a more granular perspective evaluated with enhanced safety protocols; without safety, there is a delay reduction by approximately .63 seconds. Here, the synergistic use of GPT3.5 Turbo, further bolstered by GPT4 optimizations, emerges as a decidedly more efficient strategy than an exclusive reliance on GPT4.

Figure 3f provides the results of parallel communication vs single interface use. The results show that multi-interface parallel communication offers a significant improvement for various reasons, including the sequential transmission demand pausing intermediate and future planning directives and lack of accessibility. However, it is essential to note that the communication delay is minimal in milliseconds to the GPT response times in the order of seconds. Still, multi-interface parallel vs single-interface communication for the experimental testbed provided an improvement of over 81% (saving an average of 47mS). In the broader context of communication strategy, the distinct separation of links dedicated to requests and unit control, amalgamated with a multi-link modality for handling requests, constitutes a game-changer. Moreover, combining communication with optimal techniques, including processing and GPT access, makes it more feasible to have real-time applications through AI control.

VI. Conclusion

This research highlights current communication infrastructure limitations and requirements that impact the utilization of large AI models. The strategies and solutions presented in this work aim to facilitate this transition, ensuring that AI and robotics integration occurs seamlessly, efficiently, and effectively. In a world where the interaction between AI models and robotic systems is becoming increasingly dynamic, the initiatives proposed in this research help set benchmarks for performance, accessibility, and operational efficiency. Every element, from communication pathways to AI model selection and optimization, needs addressing to ensure that the emergent AI-driven robotic ecosystems are responsive, adaptive, and reliable for use in various applications like military and healthcare [28]. As the sophistication of AI systems advances, the necessity for streamlined access to these models becomes a paramount consideration, particularly in applications demanding instantaneous responses such as robotic control. The insights offered in this paper underscore the significant role of parallel communication in enhancing the effectiveness and responsiveness of AI-enabled systems, aligning to achieve near real-time control. Future investigations will incorporate advanced interface capabilities and local AI access within robotic frameworks. By doing so, the authors anticipate an environment that supports more instantaneous and real-time operations. The integration of cutting-edge interfaces promises to enhance the operational efficiency of robotic systems and mitigate latency issues, ushering in an era where AI-driven robotics operate with unprecedented speed and precision.

References

[1] R. C. Voicu and Y. Chang, "Towards programmable networking with coopnet: A horizontal parallel multipath approach," in *2023 International Conference on Computing, Networking and Communications (ICNC)*. IEEE, 2023, pp. 111–116.

[2] R. Taori, I. Gulrajani, T. Zhang, Y. Dubois, X. Li, C. Guestrin, P. Liang, and T. B. Hashimoto, "Stanford alpaca: An instruction-following llama model," 2023.

[3] R. C. Voicu, J. A. Copeland, and Y. Chang, "Multiple path infrastructure-less networks a cooperative approach," in *2019 International Conference on Computing, Networking and Communications (ICNC)*. IEEE, 2019, pp. 835–841.

[4] R. C. Voicu and Y. Chang, "Stages of coopnet: A multipath parallel link architecture for next-gen networks," in *2021 International Wireless Communications and Mobile Computing (IWCMC)*. IEEE, 2021, pp. 592–597.

[5] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, vol. 30, 2017.

[6] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[7] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving language understanding by generative pre-training," 2018.

[8] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," *OpenAI Blog*, vol. 1, no. 8, 2019.

[9] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, 2020.

[10] S. Smith, M. Patwary, B. Norick, P. LeGresley, S. Rajbhandari, J. Casper, Z. Liu, S. Prabhumoye, G. Zerveas, V. Korthikanti *et al.*, "Using deepspeed and megatron to train megatron-turing nlg 530b, a large-scale generative language model," *arXiv preprint arXiv:2201.11990*, 2022.

[11] S. Rajbhandari, J. Rasley, O. Ruwase, and Y. He, "Zero: Memory optimizations toward training trillion parameter models," in *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*. IEEE, 2020, pp. 1–16.

[12] J. Rasley, S. Rajbhandari, O. Ruwase, and Y. He, "Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020, pp. 3505–3506.

[13] S. Rajbhandari, C. Li, Z. Yao, M. Zhang, R. Y. Aminabadi, A. A. Awan, J. Rasley, and Y. He, "Deepspeed-moe: Advancing mixture-of-experts inference and training to power next-generation ai scale," in *International Conference on Machine Learning*. PMLR, 2022, pp. 18 332–18 346.

[14] C. Li, Z. Yao, X. Wu, M. Zhang, and Y. He, "Deepspeed data efficiency: Improving deep learning model quality and training efficiency via efficient data sampling and routing," *arXiv preprint arXiv:2212.03597*, 2022.

[15] Z. Yao, R. Y. Aminabadi, O. Ruwase, S. Rajbhandari, X. Wu, A. A. Awan, J. Rasley, M. Zhang, C. Li, C. Holmes *et al.*, "Deepspeed-chat: Easy, fast and affordable rlhf training of chatgpt-like models at all scales," *arXiv preprint arXiv:2308.01320*, 2023.

[16] H. Tang, S. Gan, A. A. Awan, S. Rajbhandari, C. Li, X. Lian, J. Liu, C. Zhang, and Y. He, "1-bit adam: Communication efficient large-scale training with adam's convergence speed," in *International Conference on Machine Learning*. PMLR, 2021, pp. 10 118–10 129.

[17] Y. Lu, C. Li, M. Zhang, C. De Sa, and Y. He, "Maximizing communication efficiency for large-scale training via 0/1 adam," *arXiv preprint arXiv:2202.06009*, 2022.

[18] Q. Anthony, A. A. Awan, J. Rasley, Y. He, A. Shafi, M. Abduljabbar, H. Subramoni, and D. Panda, "Mcr-dl: Mix-and-match communication runtime for deep learning," *arXiv preprint arXiv:2303.08374*, 2023.

[19] G. Wang, H. Qin, S. A. Jacobs, C. Holmes, S. Rajbhandari, O. Ruwase, F. Yan, L. Yang, and Y. He, "Zero++: Extremely efficient collective communication for giant model training," *arXiv preprint arXiv:2306.10209*, 2023.

[20] R. C. Voicu, H. I. Abbasi, H. Fang, B. Kihei, J. A. Copeland, and Y. Chang, "Fast and reliable broadcasting in vanets using snr with ack decoupling," in *2014 IEEE International Conference on Communications (ICC)*. IEEE, 2014, pp. 574–579.

[21] N. Wake, A. Kanehira, K. Sasabuchi, J. Takamatsu, and K. Ikeuchi, "Chatgpt empowered long-step robot control in various environments: A case application," *arXiv preprint arXiv:2304.03893*, 2023.

[22] S. Biswas, "Prospective role of chat gpt in the military: According to chatgpt," *Qeios*, 2023.

[23] G. Singh, "Leveraging chatgpt for real-time decision-making in autonomous systems," *Eduzone: International Peer Reviewed/Refereed Multidisciplinary Journal*, vol. 12, no. 2, pp. 101–106, 2023.

[24] R. C. Voicu and Y. Chang, "Network packetization in multi-path environments & next-gen networks," in *2019 15th International Wireless Communications & Mobile Computing Conference (IWCMC)*. IEEE, 2019, pp. 1994–2000.

[25] H. I. Abbasi, R. C. Voicu, J. A. Copeland, and Y. Chang, "Cooperative bsm architecture to improve transportation safety in vanets," in *2017 13th International Wireless Communications and Mobile Computing Conference (IWCMC)*. IEEE, 2017, pp. 1016–1022.

[26] R. C. Voicu and Y. Chang, "Cooperative networking using passive discovery for dynamic environments," in *2022 International Wireless Communications and Mobile Computing (IWCMC)*. IEEE, 2022, pp. 1279–1284.

[27] S. Vemprala, R. Bonatti, A. Bucker, and A. Kapoor, "Chatgpt for robotics: Design principles and model abilities," *Microsoft Auton. Syst. Robot. Res*, vol. 2, p. 20, 2023.

[28] D. Oniani, J. Hilsman, Y. Peng, R. K. Poropatich, C. Pamplin, L. Legault, Y. Wang *et al.*, "From military to healthcare: Adopting and expanding ethical principles for generative artificial intelligence," *arXiv preprint arXiv:2308.02448*, 2023.