Educational Robotics With On-Device AI: An Interactive System Based on the Phi-3 Mini Small Language Model Running on Conventional Hardware

Vitor Amadeu Souza [1]

[1] Department of Electrical Engineering
University Center of Volta Redonda (UniFOA)
Volta Redonda, RJ, Brazil
vitor.amadeu@foa.org.br
https://orcid.org/0009-0002-1857-6799

Abstract. *This work presents an educational robotics system integrating speech recognition, a Small Language Model (SLM), and speech synthesis on GPU-free hardware. It uses Google Speech Recognition, Phi-3 Mini via Ollama with 4-bit quantization, pyttsx3, and Pygame. Experiments on an Intel Core i5-3570 with 16 GB RAM evaluated five interactions. The average latency was 30.16 s, with speech recognition (6.80 s), SLM processing (8.06 s), and synthesis (15.30 s). Speech synthesis was the main bottleneck (50.73%), followed by SLM (26.72%) and recognition (22.55%). Results demonstrate feasibility on low-cost hardware while highlighting the need for latency optimization for more natural interaction.*

1. Introduction

The increasing integration of transformer-based language models into robotic systems has expanded the possibilities for natural interaction between humans and machines, particularly in educational contexts where communication accessibility and naturalness are essential for student engagement and the effectiveness of pedagogical activities. As highlighted by Wang *et al.* [Wang et al., 2025], the combination of conversational intelligence provided by language models with the physical embodiment of social robots creates a powerful synergy that enables adaptive, emotionally sensitive, and highly personalized pedagogical interventions in diverse educational settings.

Large Language Models (LLMs) are deep learning models built upon large-scale transformer architectures, typically containing tens or hundreds of billions of parameters. They are capable of generating contextually coherent and semantically precise textual content through probabilistic prediction of token sequences. Although these models demonstrate remarkable linguistic capabilities, the local execution of large-scale LLMs in educational robotics faces significant technical and economic barriers: models such as GPT-5 or Llama-3 70B require high-performance dedicated GPUs, representing substantial investments that are often inaccessible to educational institutions, or rely on cloud-based API services with recurring per-request costs, making them economically unsustainable for continuous use in classrooms where multiple students interact simultaneously.

In response to these constraints, Small Language Models (SLMs) have emerged as a promising alternative for applications requiring natural language processing under computational limitations. SLMs typically contain between 1 and 10 billion parameters

and achieve competitive performance in many natural language tasks through strategies such as carefully curated high-quality training datasets, knowledge distillation from larger models, and targeted architectural optimizations [Abdin et al., 2024]. The Phi-3 Mini model, developed by Microsoft Research with 3.8 billion parameters trained on 3.3 trillion tokens, demonstrates that SLMs can achieve 69% accuracy on the MMLU benchmark and a score of 8.38 on MT-Bench, approaching the performance of substantially larger models such as Mixtral 8x7B (47B parameters) and GPT-3.5 (175B parameters), while requiring only 2–3 GB of RAM in quantized versions and running efficiently on conventional CPUs without the need for dedicated GPUs [Abdin et al., 2024]. This computational efficiency makes SLMs particularly suitable for educational robotics in resource-constrained environments.

In the Brazilian educational context, where the deployment of advanced technologies frequently faces severe budgetary constraints [Ribeiro et al., 2015], it becomes essential to develop SLM-based solutions that can operate effectively on the computing infrastructure commonly available in schools, without relying on high-cost specialized hardware or constant internet connectivity for cloud services. Recent research in educational robotics has identified that schools often operate with limited budgets, making it difficult to invest in advanced robotic systems, and that the high cost of such systems may exacerbate the digital divide between well-resourced institutions and those with fewer resources [AI for Good, 2024].

The current literature on educational robotics with language models reveals a critical gap regarding performance characterization of SLMs operating exclusively on conventional CPU hardware without GPU acceleration. Most studies reported in the literature rely on high-performance GPUs that achieve substantially higher inference speeds or depend on cloud-based LLM services which, although offering superior linguistic capabilities, introduce dependencies on connectivity, recurring API costs, and sensitive data privacy issues—particularly relevant in educational environments. This lack of systematic performance characterization makes it difficult for researchers and developers to assess the feasibility of deploying intelligent conversational systems on accessible hardware available in educational institutions.

Interaction latency is a critical factor that directly influences user experience and the pedagogical effectiveness of conversational robotic systems. For human-like natural voice interactions, latencies as low as 300 milliseconds may be desirable, whereas delays exceeding 1.5 seconds can rapidly degrade user experience [Cresta, 2024]. Modern text-to-speech (TTS) systems can achieve latencies below 150 milliseconds in optimized implementations [BentoML, 2024], while consistency-based TTS models such as CM-TTS have demonstrated real-time, high-quality speech synthesis with reduced latency [Li et al., 2024]. However, systematic characterization of latency across the complete voice-to-voice interaction pipeline, including automatic speech recognition (ASR), SLM inference, and TTS, remains insufficiently documented in the literature for accessible hardware configurations operating solely on conventional CPUs. Optimizing latency in voice-based systems requires precise measurement and tuning of each component, including ASR, SLMs, and TTS modules [Cresta, 2024].

This work aims to address this gap by implementing, characterizing, and systematically analyzing a complete educational robotic system based on an SLM operating

exclusively on accessible CPU hardware, with automatic collection of fine-grained performance metrics at each stage of the interaction pipeline. The specific objectives are: (1) to develop and implement a functional conversational robotic prototype integrating automatic speech recognition, natural language processing via Phi-3 Mini, and speech synthesis operating fully locally without cloud dependence; (2) to quantitatively characterize latency introduced by each individual component of the interaction pipeline through automatic timestamp-based measurements on representative CPU hardware typical of resource-constrained educational institutions; (3) to analyze the relationship between the complexity of SLM-generated responses and processing latency, identifying performance patterns and computational bottlenecks; and (4) to provide reproducible benchmarks that may serve as reference for researchers and developers implementing similar systems in diverse educational contexts.

2. Theoretical Framework

Large Language Models (LLMs) represent the most recent stage in the evolution of natural language processing. They are grounded in the Transformer architecture introduced by Vaswani *et al.* in the seminal work *Attention Is All You Need* [Vaswani et al., 2017]. The self-attention mechanism proposed by the authors enables the model to compute contextualized representations of each token by simultaneously considering all others. This process allows the model to capture long-range dependencies without the limitations inherent to earlier recurrent architectures [Vaswani et al., 2017].

The Phi-3 Mini model, developed by Microsoft Research and used in this study, belongs to the emerging category of Small Language Models (SLMs), which offer advanced language capabilities with substantially reduced computational requirements. With 3.8 billion parameters trained on 3.3 trillion tokens, Phi-3 Mini achieves competitive performance on benchmarks such as MMLU (69% accuracy) and MT-Bench (8.38 points), reaching levels comparable to larger models such as Mixtral 8×7B and GPT-3.5 [Abdin et al., 2024]. The model's efficiency results from a carefully curated training dataset composed of educationally filtered texts and synthetic data designed to enhance mathematical reasoning, programming skills, and general-knowledge competencies [Abdin et al., 2024].

This configuration makes Phi-3 Mini suitable for execution on conventional hardware. Quantization techniques reduce weight precision from FP16 to INT4, decreasing memory requirements from approximately 7.6 GB to about 2.3 GB, accelerating inference while preserving most of the model's response quality [Abdin et al., 2024].

The inference layer used in this study, the Ollama framework, provides a streamlined interface for local execution of language models. It relies on the `llama.cpp` backend, a C++ implementation optimized for CPUs with instruction-set extensions such as AVX2. This approach enables fully offline operation without dependence on GPUs or cloud services, ensuring cost predictability, privacy, and accessibility in educational environments.

Regarding additional system technologies, Google Speech Recognition relies on deep learning models trained on thousands of hours of multilingual audio. Comparative studies show that commercial Automatic Speech Recognition (ASR) systems such as Google's achieve word error rates (WER) between 4.9% and 5.5% on controlled bench-

marks such as Switchboard [Microsoft Research, 2017, Faria et al., 2022]. However, in real-world scenarios with diverse audio conditions, WER values typically range from 20% to 22% [Ferraro et al., 2023], depending on factors such as accent, background noise, and domain specificity.

Human–robot interaction (HRI) involves multiple factors that influence engagement, attention, and learning outcomes. A systematic review by Belpaeme *et al.* [Belpaeme et al., 2018], published in *Science Robotics*, demonstrates that social robots can achieve learning results comparable to human tutors in specific tasks. The physical presence of the robot, combined with facial expressions, gestures, and responsive behavior, enhances students' emotional and cognitive engagement.

Conversational latency is a critical element for natural interaction. Studies on human dialogue timing indicate that turn-taking typically occurs within 200 to 300 ms after the end of the previous utterance [Bögels and Torreira, 2015]. This short interval is particularly notable considering that human vocal production often requires more than 600 ms for planning and execution [Levinson and Torreira, 2015]. Computational systems introducing delays longer than one or two seconds are frequently perceived as slow and unnatural. User-experience analyses show that pauses above 1.5 seconds tend to reduce the perceived fluency of the interaction [Cresta, 2024].

To mitigate the perception of slowness, the use of visual feedback is essential. The alternating images synchronized with speech, implemented in this study, communicate to the user that the system remains active during processing. This reduces uncertainty regarding system state and maintains engagement, even when local model inference requires longer times than those observed in natural human conversation.

3. Methodology

This study adopted an experimental development approach combined with quantitative performance analysis. The proposed methodology consisted of designing and implementing a functional prototype of a conversational robotic system, conducting real-world tests with automatic collection of temporal metrics, and performing descriptive statistical analysis of the collected data. The purpose of these procedures was to enable performance characterization and the identification of computational bottlenecks. The system was developed using Python version 3.10, selected for its extensive ecosystem of specialized libraries for audio processing, graphics, and API communication. The full implementation source code is available at: `https://github.com/vitor-souza-ime/roboSLM`.

The overall architecture follows a sequential pipeline (Figure 1). Each complete user interaction progresses through four main stages. The first stage is speech recognition, where audio from the microphone is converted into text. The second stage is natural language processing, in which the recognized text is sent as a prompt to the Phi-3 Mini model running locally via Ollama, generating the corresponding textual response. The third stage is speech synthesis, converting the generated text into audio and playing it back to the user. The fourth stage consists of animated visual feedback, displaying alternating images of the robot with open and closed mouth synchronized with the audio playback. The pipeline was instrumented with precise timestamp measurements at the beginning and end of each stage using Python's standard `time.time()` function, allowing the

exact computation of latency introduced by each individual component.

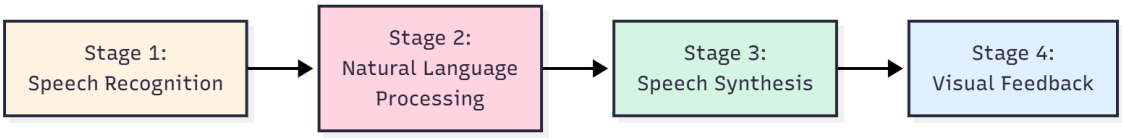

Figure 1. Project pipeline

The hardware used was a desktop computer equipped with an Intel Core i5-3570 third-generation processor (Ivy Bridge, 2012) operating at a base frequency of 3.40 GHz with four physical cores. The machine had 16 GB of DDR3-1600 RAM and no dedicated GPU, relying solely on Intel HD Graphics 2500 integrated graphics. This configuration is entirely CPU-based and reflects computer systems commonly found in educational institutions with limited budgets, where informatics laboratories often contain outdated machines with insufficient memory, progressively becoming incompatible with contemporary educational demands [Souza and Silva, 2019]. The operating system was Windows 10 Pro version 21H2.

For speech recognition, the SpeechRecognition library version 3.10 developed by Zhang [Zhang, 2017] was employed. This library provides a unified interface for multiple speech-recognition backends, including the Google Speech API. It was configured with dynamic energy threshold adjustment (`dynamic_energy_threshold`) to adapt to varying ambient noise levels. The pause duration (`pause_threshold`) was set to 0.8 s, representing a balance between responsiveness and robustness against false positives. The language was set to American English (en-US). Before each capture, the library executed `adjust_for_ambient_noise` for 0.5 s to calibrate speech detection based on background noise, following recommendations from the official documentation suggesting at least 0.5 s for improved accuracy [Amos, 2023].

For natural language processing, the study used the Ollama framework version 0.13.1 running the Phi-3 Mini model in its 4-bit quantized variant (`phi3:mini`), optimized for CPU execution. Communication with Ollama occurred via HTTP POST requests to the `/api/generate` endpoint, with parameters including a temperature value of 0.7 to balance creativity and coherence. The prompt was formatted with explicit instructions to ensure concise responses in English.

For speech synthesis, the `pyttsx3` library version 2.99 was used, a cross-platform offline text-to-speech solution for Python that requires no internet connection [Bhat, 2025]. The system was configured to use native SAPI5 voices through the Microsoft Speech API. The `pyttsx3` library provides control over speech rate, volume, and voice selection, operating across different operating systems via dedicated engines such as SAPI5 on Windows, NSSpeechSynthesizer on macOS, and eSpeak on Linux [GeeksforGeeks, 2025]. The speech rate was configured to 155 words per minute.

For the visual interface, the Pygame library version 2.6.1 was used to render the graphical window and implement synchronized animation during speech synthesis. Two images were alternated at 6 Hz to represent open and closed mouth states, as shown in Figure 2.

Automatic metric collection was implemented through a global data struc-

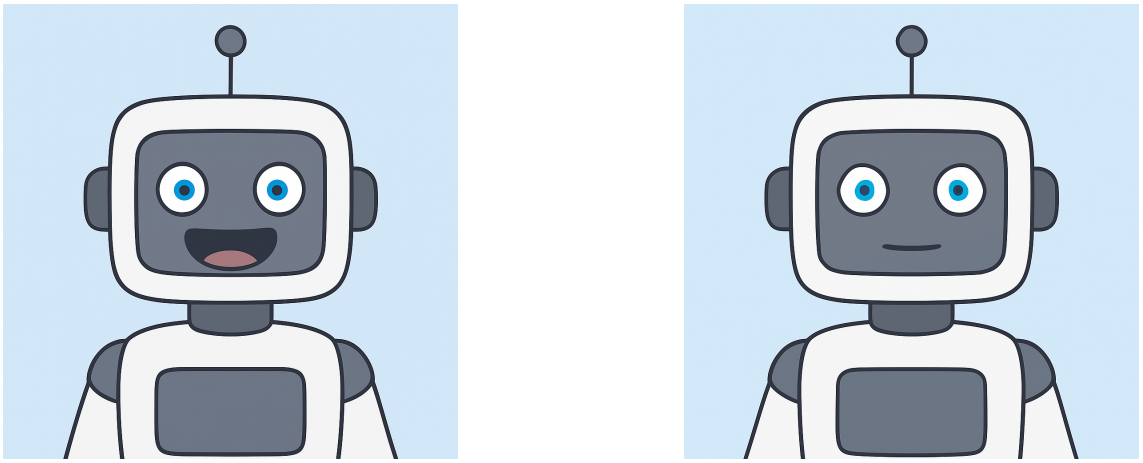

Figure 2. Images used in the project to simulate speech.

Table 1. Experimental results of five test interactions

Interaction	Input	Input Words	Response (truncated)	Output Words	T_Recog (s)	T_SLM (s)	T_Synth (s)
1	what is machine learning	4	Machine learning is a subset of artificial intelligence that involves...	32	5.99	6.61	12.40
2	what is a neural network	5	A neural network is a series of algorithms that endeavors...	42	6.41	8.76	16.23
3	what is the capital of Japan	6	Tokyo	1	8.46	1.55	1.54
4	Define artificial intelligence in one sentence	6	Artificial intelligence is the simulation of human intelligence processes...	52	7.44	12.82	28.88
5	explain Newton's second law	4	Newton's Second Law of Motion states that the acceleration...	46	5.71	10.56	17.44
Average	-	5.00	-	34.60	6.80	8.06	15.30
Std. Dev.	-	0.90	-	18.01	1.02	3.84	8.80

ture containing counters for `total_interactions` and accumulators for `total_recognition_time`, `total_llm_time`, and `total_speech_time`. Error counters (`recognition_errors` and `llm_errors`) were also included. For each interaction, timestamps were recorded immediately before and after each operation, and the differences were added to the corresponding accumulators. Metrics were automatically saved to a JSON file at the end of execution to enable post-processing analysis and ensure data preservation.

The experimental protocol consisted of five test interactions designed to cover different question types and response complexities. Included were explanations of technical concepts in computing and artificial intelligence ("what is machine learning" and "what is a neural network"), simple factual questions ("what is the capital of Japan"), a concise definition request ("define artificial intelligence in one sentence"), and a scientific principle explanation ("explain Newton's second law"). This diversity enables evaluation of the system's behavior for both short and long responses as well as factual and explanatory prompts. All questions were formulated clearly in English by a proficient speaker to minimize recognition errors. Tests took place in a laboratory environment with moderate background noise typical of real educational settings.

4. Results and Discussion

The experimental results collected over five test interactions are summarized in Table 1, which presents for each interaction the recognized input text, number of input words, generated response (truncated due to space limitations), approximate number of output words, and the measured times in seconds for speech recognition, SLM processing, and speech synthesis. The analysis of these data reveals important patterns regarding the system's behavior and performance bottlenecks when running on conventional CPU hard-

ware without dedicated acceleration.

The average speech recognition time was 6.80 seconds with a standard deviation of 1.02 seconds, showing relative consistency across interactions. This time includes audio capture until a pause indicating the end of the utterance, as well as acoustic and linguistic processing. The observed minimum and maximum times (5.99 to 8.46 seconds) partly correlate with the number of words in the utterance, with longer utterances tending to require more time. These values exceed typical times reported in the literature for modern local recognition systems such as Whisper, which can process audio near real-time with a Real-Time Factor (RTF) close to 0.3s.

Comparatively, the 6-8 seconds of recognition latency, although high for natural human conversation where responses typically begin approximately 200 milliseconds after turn completion [Stivers et al., 2009,Roberts et al., 2015], remain below the 10-second threshold identified by Nielsen [Nielsen, 2024] as critical for maintaining user attention in computational tasks. Studies on latency in e-learning indicate that latencies above 300-420 milliseconds begin to negatively impact pedagogical effectiveness [Bush et al., 2008], suggesting that while the observed values in the present system demonstrate technical feasibility, substantial optimization is needed for large-scale educational applications.

Figure 3 shows the relationship between the number of words in the utterance and the speech recognition time, illustrating the trend of increasing latency with longer input length.

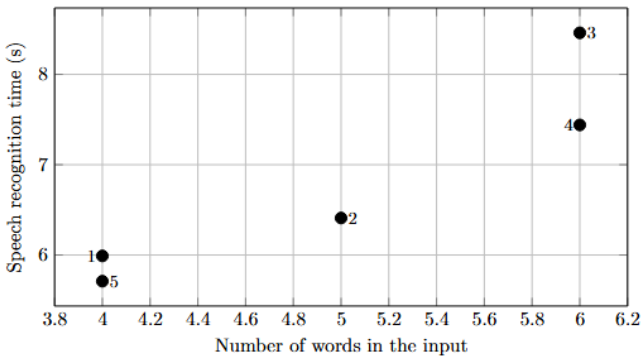

Figure 3. Relationship between utterance word count and speech recognition time

The SLM processing time exhibited the greatest variability among all components, with an average of 8.06 seconds and a standard deviation of 3.84 seconds, reflecting a strong dependence between response complexity and length and processing latency. Interaction 3, which requested simply the capital of Japan, resulted in a one-word response (“Tokyo”) and a minimal processing time of 1.55 seconds, demonstrating that for extremely short factual responses the model can operate with relatively low latency even on CPU.

In contrast, interaction 4, which requested the definition of artificial intelligence in one sentence, produced the longest response with 52 words and the maximum processing time of 12.82 seconds, indicating that the model did not strictly follow the “one sentence” instruction but generated multiple explanatory sentences—typical behavior of

SLMs prioritizing completeness and informativeness over strict brevity constraints.

Figure 4 depicts the relationship between response length and SLM processing time, highlighting the strong correlation between these variables. Very short responses are dominated by fixed inference initialization overhead, whereas longer responses incur progressively higher token-by-token generation costs.

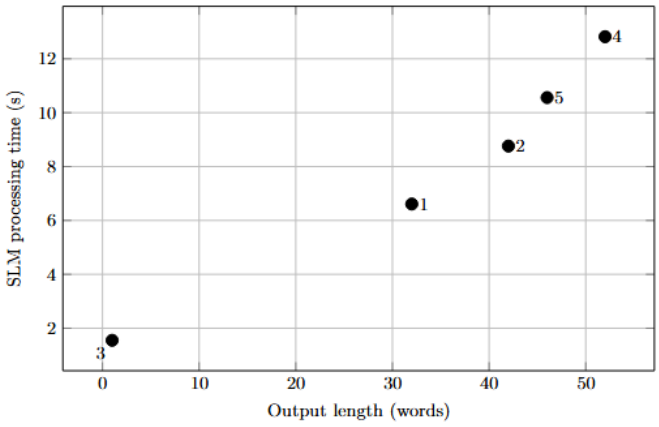

Figure 4. Relationship between response length and SLM processing time

Comparative analysis of the three main pipeline components reveals that speech synthesis was the dominant bottleneck, consuming on average 50.73% of total interaction time, followed by SLM processing with 26.72%, and speech recognition with 22.55%. This latency distribution suggests that optimization efforts should primarily focus on replacing the speech synthesis engine with a more efficient alternative, followed by accelerating SLM inference.

Figure 5 illustrates the percentage distribution of total interaction time among the three main pipeline components, emphasizing the disproportionate latency caused by speech synthesis.

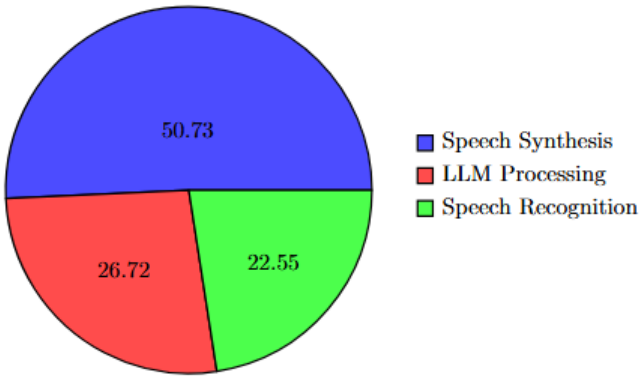

Figure 5. Latency distribution across main pipeline components

The average total latency per interaction of 30.16 seconds with a standard deviation of 12.22 seconds exceeds the thresholds established by Nielsen [Nielsen, 2024] for human-computer interaction: 1 second to maintain the flow of thought without interruption, and 10 seconds as the maximum limit to maintain user attention. Above 10 seconds,

users typically lose context and need to reorient themselves when returning to the interface.

Commercial virtual assistants such as Alexa and Google Assistant respond within 2-5 seconds total, including recognition, processing, and synthesis.

However, it is important to contextualize these results considering the hardware used is over a decade old (2012) and represents the lower spectrum of hardware available in 2024-2025. Modern consumer processors like the Intel Core i5-14600K (14th generation, 2024) with 14 cores (6P+8E) and frequencies up to 5.3 GHz, or AMD Ryzen 5 9600X (Zen 5 architecture, August 2024) with 6 cores operating up to 5.4 GHz and AVX-512 support, are expected to achieve substantial inference performance improvements. CPU optimization studies demonstrate that AVX-512 support can accelerate prompt evaluation by up to 10x compared to AVX2 instructions [Mozilla Ocho, 2024], while Intel processors with Advanced Matrix Extensions (AMX) reach 7-10x latency improvements on quantized models [Lenovo and Intel, 2024]. Considering improved memory architectures, higher clock speeds, and advanced SIMD instructions in current CPUs, conservative estimates suggest 3-5x speedups compared to the Core i5-3570, potentially reducing total latency to the 6-10 second range per interaction.

Moreover, GPU use, even in mid-range models like the NVIDIA RTX 4060, can provide approximately an order of magnitude acceleration relative to CPU-only execution, achieving token generation rates of 50-80 tokens per second for similarly quantized models. In such configurations, total latency per interaction could fall to 5-10 seconds, approaching commercial virtual assistant performance and providing a significantly smoother user experience for continuous educational applications.

Error analysis showed no speech recognition errors (recognition_errors = 0) and no SLM communication errors (llm_errors = 0) during the five test interactions, indicating robust operation under controlled conditions with a proficient speaker, moderate noise environment, and locally running Ollama server. The quality of Phi-3 Mini responses was qualitatively adequate for educational purposes, with factually correct, linguistically coherent, and appropriately detailed answers for all test questions.

Limitations of this study include:

1. Small sample size: only five interactions tested, sufficient for technical feasibility demonstration but inadequate for robust statistical generalization or prolonged use analysis.
2. Controlled environment: tests conducted in an office with a single proficient speaker do not capture variability in real educational settings with multiple students, variable noise, diverse accents, especially in Brazilian contexts where English is a second language with heterogeneous proficiency.
3. Lack of subjective evaluation: absence of user Quality of Experience (QoE) metrics to complement latency data with perceptions of naturalness and educational effectiveness.
4. No direct benchmarking: absence of side-by-side comparisons with alternative systems (GPU, cloud services) to quantify performance vs. cost and accessibility trade-offs.

Future work could address these limitations by expanding sample sizes, deploying

and evaluating in real educational environments with diverse student populations, collecting qualitative user experience data, and systematically benchmarking various hardware and software configurations.

To facilitate such investigations, the complete source code and reproduction instructions are publicly available in a GitHub repository.

5. Conclusions

This work presented the development, implementation, and performance analysis of a complete conversational robotic system for education, operating exclusively on conventional CPU hardware that is accessible and representative of the infrastructure available in many educational institutions with limited resources. The experimental results demonstrate the technical feasibility of integrating speech recognition, natural language processing via a local Small Language Model (SLM), and speech synthesis in a functional pipeline without reliance on dedicated GPUs or cloud services, achieving an average total latency of approximately 30 seconds per complete interaction on a third-generation Intel Core i5-3570 processor with 16 GB of RAM.

The main contributions of this work to educational robotics and conversational systems can be organized into five axes. The first contribution is the real-world demonstration that SLM-based systems can operate entirely on accessible CPU hardware. The second is the provision of reproducible benchmarks for each component of the conversational pipeline, enabling direct comparisons in future research. The third is the identification of the main performance bottlenecks, along with quantification of their relative contribution to total latency, which enables precise guidance for optimizations. The fourth is the validation of the Phi-3 Mini Small Language Model as a suitable solution for interactive educational applications, indicating that compact models can meet many demands without the need for larger and more costly architectures. The fifth contribution is the availability of a complete and open system with automatic metric collection that can serve as a basis for replication and extension by other researchers.

The current trend toward more efficient language models suggests that CPU performance will continue to improve in the coming years. Thus, the CPU-GPU performance gap tends to narrow, making fully local solutions even more viable for educational applications. The characterization presented here provides a foundation for future research and the development of inclusive educational technologies that benefit students in diverse social and economic contexts.

References

J. Wang et al., "Large language models for robotics: Opportunities, challenges, and perspectives," *Journal of Automation and Intelligence*, vol. 4, no. 1, pp. 52–64, Mar. 2025.

M. Abdin et al., "Phi-3 Technical Report: A Highly Capable Language Model Locally on Your Phone," *Microsoft Research*, 2024. Available at: https://arxiv.org/abs/2404.14219.

D. Gouaillier et al., "The NAO humanoid: a combination of performance and affordability," *arXiv preprint arXiv:0807.3223*, 2008.

A. C. R. Ribeiro, D. A. C. Barone and L. E. P. Mizusaki, "ROBO+ EDU: Project and Implementation of Educational Robotics in Brazilian Public Schools," in *Robot Intelligence Technology and Applications 3*, Cham: Springer, 2015, pp. 495–503.

AI FOR GOOD, "The future of educational robotics: enhancing education, bridging the digital divide, and supporting diverse learners," ITU AI for Good, Dec. 16, 2024. Available at: https://aiforgood.itu.int/the-future-of-educational-robotics-enhancing-education-bridging-the-digital-divide-and-supporting-diverse-learners/.

CRESTA, "Engineering for Real-Time Voice Agent Latency," Cresta Blog, 2024. Available at: https://cresta.com/blog/engineering-for-real-time-voice-agent-latency.

BENTOML, "Exploring the World of Open-Source Text-to-Speech Models," BentoML Blog, 2024. Available at: https://www.bentoml.com/blog/exploring-the-world-of-open-source-text-to-speech-models/.

X. Li et al., "Cm-tts: Enhancing real-time text-to-speech synthesis efficiency through weighted samplers and consistency models," in *Findings of the Association for Computational Linguistics: NAACL 2024*, 2024, pp. 3777–3794.

A. Vaswani et al., "Attention Is All You Need," in *Advances in Neural Information Processing Systems 30 (NIPS 2017)*, 2017. Available at: https://arxiv.org/abs/1706.03762.

A. Faria et al., "Toward zero oracle word error rate on the switchboard benchmark," *arXiv preprint arXiv:2206.06192*, 2022.

MICROSOFT RESEARCH, "Microsoft researchers achieve new conversational speech recognition milestone," Microsoft Research Blog, Jun. 13, 2017. Available at: https://www.microsoft.com/en-us/research/blog/microsoft-researchers-achieve-new-conversational-speech-recognition-milestone/.

A. Ferraro et al., "Benchmarking open source and paid services for speech to text: an analysis of quality and input variety," *Frontiers in Big Data*, vol. 6, 2023.

T. Belpaeme et al., "Social robots for education: A review," *Science Robotics*, vol. 3, no. 21, p. eaat5954, 2018.

S. Bögels and F. Torreira, "Listeners use intonational phrase boundaries to project turn ends in spoken interaction," *Journal of Phonetics*, vol. 52, pp. 46–57, Sept. 2015.

S. C. Levinson and F. Torreira, "Timing in turn-taking and its implications for processing models of language," *Frontiers in Psychology*, vol. 6, p. 731, 2015.

D. J. Souza and M. A. Silva, "CEMAO OS: A GNU/Linux distribution for obsolete computer labs," Undergraduate Thesis — IF Baiano, 2019. Available at: https://www.ifbaiano.edu.br/unidades/bonfim/files/2023/04/TCC-Jeferson-Matos.pdf.

A. Zhang, "SpeechRecognition," PyPI, version 3.11, 2017. Available at: https://pypi.org/project/SpeechRecognition/.

D. Amos, "The Ultimate Guide To Speech Recognition With Python," Real Python, Sept. 7, 2023. Available at: https://realpython.com/python-speech-recognition/.

N. M. Bhat, "pyttsx3: Text to Speech (TTS) library for Python 3," PyPI, version 2.99, Jul. 8, 2025. Available at: https://pypi.org/project/pyttsx3/.

GEEKSFORGEEKS, “Text to Speech by using pyttsx3 - Python,” Apr. 14, 2025. Available at: https://www.geeksforgeeks.org/python/python-text-to-speech-by-using-pyttsx3/.

T. Stivers et al., “Universals and cultural variation in turn-taking in conversation,” *PNAS*, vol. 106, no. 26, pp. 10587–10592, 2009.

S. G. Roberts, F. Torreira and S. C. Levinson, “The effects of processing and sequence organization on the timing of turn taking: a corpus study,” *Frontiers in Psychology*, vol. 6, p. 509, 2015.

J. Nielsen, “Response Time Limits,” Nielsen Norman Group, 2024. Available at: https://www.nngroup.com/articles/response-times-3-important-limits/.

H. F. Bush et al., “An Investigation of the Effect of Network Latency on Pedagogic Efficacy: A Comparison of Disciplines,” *Contemporary Issues in Education Research*, vol. 1, no. 4, pp. 11–26, 2008.

MOZILLA OCHO, “Llamafile 0.7 release notes: AVX-512 support,” GitHub, Apr. 1, 2024. Available at: https://github.com/Mozilla-Ocho/llamafile/releases/tag/0.7.

LENOVO and INTEL, “AI Inferencing on Intel CPU-Powered Lenovo Servers: Accelerating LLM Performance with Intel Technology,” Lenovo Press, 2024. Available at: https://lenovopress.lenovo.com/lp2167.pdf.