ISSN: 2945-4565 VOL 05, ISSUE 01, 2026

AI Powered Debugger: Intelligent AI-Based Debugging Framework for Block-Based Music Programming

Miss. Anushka Warhekar[1], Miss. Vaishnavi Rathod[2], Mr. Meher Pathe[3], Mr. Atharva Kale[4], Mr. Yash Matkar[5], Mr. Samyak Mujmule[6] and Prof. Sunil Panjabrao Chinte[7]

[1,2,3,4,5,6]B.E Students, Department of Computer Engineering, Jagadambha College of Engineering and Technology, Yavatmal , India.
[7]Prof, Department of Computer Engineering, Jagadambha College of Engineering and Technology, Yavatmal, India.

Abstract

Block-based programming platforms like Music Blocks help beginners learn coding through music, but debugging errors in such programs is often difficult for learners. This project proposes an AI-powered debugger for Music Blocks that automatically identifies errors, explains their causes, and suggests possible solutions in simple language. The system uses artificial intelligence techniques to analyze block-based programs and provide real-time feedback to users. By integrating AI with Music Blocks, the proposed debugger reduces debugging time, improves learning efficiency, and enhances creativity in music programming. Experimental evaluation was carried out in a real educational environment that showed the proposed approach significantly reduces debugging time, program correctness, and improving the engagement of learners. It seems that AI-aided debugging would very effectively support music education, strengthening computational thinking and deepening programming concepts in inexperienced learners.

Keywords: Android, Mobility, Technology, Safety.

1. INTRODUCTION

Music Blocks is an educational visual programming tool that integrates music studies with computational thinking, making it easier to learn for beginners as well. But students might encounter problems in pointing out and fixing errors in programs, hindering them from learning faster.

For this purpose, there arises an imperative need to use this AI-based educational debugging tool in the Music Blocks system. The proposed tool uses AI technology to automatically identify errors in programs that learners might make and assist them with relevant hints to enhance problem-solving abilities among learners. track of their child's progress.

The AI-Powered Music Blocks Debugger is an intelligent assistant designed to help children debug their Music Blocks projects. It combines the power of modern AI with educational pedagogy to provide:

Kid-friendly explanations of coding errors and music theory concepts
Interactive debugging sessions that encourage learning through play
Context-aware suggestions based on Music Blocks documentation and examples
Real-time analysis of Music Blocks project JSON code

Beginner learners can now easily understand programming concepts thanks to Music Blocks, an effective visual programming platform that combines computational thinking with music education. However, debugging visual music programs can be difficult for many students, which frequently results in confusion and decreased engagement. Through intelligent feedback

ISSN: 2945-4565 VOL 05, ISSUE 01, 2026

and individualized learning support, recent developments in artificial intelligence (AI) present new opportunities to improve educational tools.
This project presents an AI-driven educational debugger for Music Blocks that automatically identifies typical mistakes and gives students context-aware advice. Without taking the place of teacher intervention, the suggested system seeks to increase learner engagement, decrease debugging time, and improve problem-solving abilities.

Purpose
Music Blocks is a learning visual programming application that combines music learning with computer programming. As a result, it is easier for students to learn. However, students may face difficulties in identifying and correcting errors in a program. Thus, students are unable to learn faster. In this context, there emerges a great need for the use of this AI learning-debugging tool in the Music Blocks system. The learning-debugging solution developed applies AI technology to detect any error in a program that students may commit. It then helps students with guidance.

Scope
The scope of this project revolves around the design and implementation of an AI-powered educational debugger that will enable the Music Blocks visual programming environment. It should target beginner learners by detecting common programming errors and giving intelligent, context-aware feedback that supports learning. This project further focuses on strengthening computational thinking, problem-solving skills, and increasing learner engagement in music-based programming activities. The system is intended strictly for educational use and is envisioned to assist learners rather than replace instructor support. In addition, the research lays the base for potential future enhancements regarding adaptive feedback mechanisms, advanced AI techniques, and possible integrations with other educational programming platforms.

Existing System
Currently, several block-based programming platforms such as Scratch, Blockly, and Music Blocks are widely used in educational environments to help children and beginners learn coding concepts in a simple, visual, and interactive way. These systems allow users to create programs by dragging and dropping visual “blocks” that represent code instructions, instead of typing traditional text-based code.
The existing facility for debugging within the Music Blocks programming environment is incomplete and merely supports simple error notifications and trial-and-error approaches by learners themselves. They are expected to find logical or structural mistakes independently, without further details or hints. Debugging depends either on instructor or peer support, not always available. Such an approach may lead to frustration among learners, lengthening debugging time and reducing engagement, especially for beginners. Finally, the current system lacks in providing intelligent, personalized feedback as an important basis for supporting the learning process.

Proposed System
The new system comes with the implementation of an AI-based educational debugger for the Music Blocks coding environment. This system would efficiently examine the learners’ code for detecting syntax, logical, and structural errors. The system would make the most out of artificial intelligence approaches like pattern recognition and rule-based reasoning to point out mistakes and give learners intelligent and context-aware feedback. Rather than pointing out

ISSN: 2945-4565 VOL 05, ISSUE 01, 2026

mistakes, the system would give learners guidance and explanations that would help them critically analyze and debug the code on their own. It would keep adaptable to learners and provide personalized help to beginners. Rather than relying on tutors and trial-and-error debugging, the new system would seek to improve learner engagement and increase the accuracy and competence of learners in computational thinking and music-based programming.

2. LITERATURE SURVEY

Mitchel Resnick et al. (2009) – "Scratch: Programming for All"
This study introduces Scratch, a visual programming environment created at MIT to make programming accessible to children and beginners. By using drag-and-drop blocks instead of traditional text-based syntax, Scratch removes the complexity of coding and encourages creativity. The research demonstrates how such block-based environments foster logical thinking, problem-solving, and computational creativity.This work serves as a strong foundation for developing systems like the AI Powered Debugger for Music Block, where learners can explore coding concepts through creative expression.

Google Developers (2012) – "Blockly: A Client-Side Library for Visual Programming"
Google's Blockly is an open-source, client-side JavaScript library that allows developers to create block-based programming interfaces. The library translates visual code blocks into various programming languages, making it an adaptable tool for educational and professional use. The study emphasizes the significance of user-friendly interfaces and how visual abstraction helps users focus on logic instead of syntax. In the context of the proposed project, Blockly's architecture serves as an inspiration for the drag-and-drop interface in Music Blocks and its AI-driven debugging.

McFee, B. et al. (2015) – "Librosa: Audio and Music Signal Analysis in Python"
This research presents Librosa, an open-source Python library for analyzing and visualizing musical and audio signals. It provides functionalities for extracting important features such as tempo, pitch, rhythm, chroma, and harmonic structures. The study highlights its applications in music information retrieval and signal processing.

Tzanetakis, G., & Cook, P. (2002) – "Musical Genre Classification of Audio Signals"
This research introduces methods for automatically classifying music genres based on timbral, rhythmic, and pitch-related features. The authors propose feature extraction techniques that capture the unique patterns of different musical styles. The study's importance lies in its combination of signal processing and machine learning to understand musical content. In the proposed project, similar analytical concepts are used to evaluate the musical correctness and harmony of compositions created through Music Blocks. By analyzing these features, the AI debugger can suggest improvements that align with basic music theory and aesthetic principles.

VanLehn, K. (2011) – "The Relative Effectiveness of Intelligent Tutoring Systems"
This paper evaluates the effectiveness of Intelligent Tutoring Systems (ITS) and compares them to human tutoring. The results show that AI-driven tutoring systems can produce learning outcomes close to those achieved through human instruction. The research emphasizes the importance of real-time, context-aware feedback in improving learner understanding and motivation. The findings strongly support the educational goal of the AI Powered Debugger, which provides child-friendly, interactive explanations that help students grasp coding and musical concepts.

ISSN: 2945-4565 VOL 05, ISSUE 01, 2026

Yang, Shulin & Hu, Jieping (2014) – "Research and Implementation of Web Services in Android Network Communication Framework Volley"
This paper focuses on using web services and network communication frameworks for efficient data handling and scalability.

3. METHODOLOGY

AI Debugger Module:
This is the core component of the system. It analyzes the Music Blocks project JSON code in real-time and identifies both coding and musical errors. Using natural language processing and machine learning models, the debugger provides kid- friendly explanations of errors instead of displaying complex technical messages. It also offers context-aware suggestions based on the user's current project and the Music Blocks documentation.
For example, if a user's rhythm loop is incorrect or a melody block is misplaced, the AI will explain the issue in simple terms and guide the student step-by-step to fix it. This module enhances the learning experience by turning debugging into an interactive teaching moment.

Student Learning Module:
In this module, students interact with the system through a drag-and-drop block interface, where they can create, modify, and debug their music projects. The system continuously tracks their progress and provides interactive hints, quizzes, and guided corrections. Students not only learn programming logic but also gain a deeper understanding of musical theory concepts such as rhythm, melody, tempo, and harmony.
The AI provides real-time suggestions and explanations that encourage self-learning and experimentation, making the process fun and educational. This module bridges the gap between coding and creativity, motivating children to explore STEAM (Science, Technology, Engineering, Arts, and Mathematics) learning.

Educator/Administrator Module:
This module allows teachers or mentors to monitor student progress and performance. Educators can access activity reports, track the types of errors students frequently encounter, and assess their understanding of both coding and music concepts. The system's analytics dashboard provides valuable insights into how students are interacting with the AI debugger, helping educators adjust their teaching strategies accordingly.
Additionally, teachers can add custom examples, exercises, and lesson plans into the system, which the AI can later reference while guiding students during.

AI Model and Knowledge Base Integration:
The AI model is trained on a dataset containing Music Blocks examples, documentation, and real debugging cases. It uses pattern recognition and natural language understanding to interpret both code and music theory relationships. The system's knowledge base evolves continuously by learning from user interactions and improving the accuracy of its feedback. It combines rule-based logic for syntax correction with machine learning-based reasoning for musical analysis, ensuring a comprehensive debugging experience.

User Interaction Module:
This module focuses on creating a child-friendly, interactive interface that promotes engagement. The system communicates through animated feedback, guided dialogues, and

ISSN: 2945-4565 VOL 05, ISSUE 01, 2026

multiple-choice prompts, turning complex debugging into a fun learning activity. It ensures that the environment remains encouraging and curiosity- driven, rather than error-focused, so that children view mistakes as learning opportunities**.**

Proposed Plan of Work

The proposed system aims to design and develop an AI-Powered Debugger for Music Block, an intelligent tool capable of automatically detecting, analyzing, and reporting musical and technical errors in digital audio files. The project begins with problem identification and requirement analysis, where the challenges in manual music debugging are studied, and the functional as well as non-functional requirements of the system are defined.

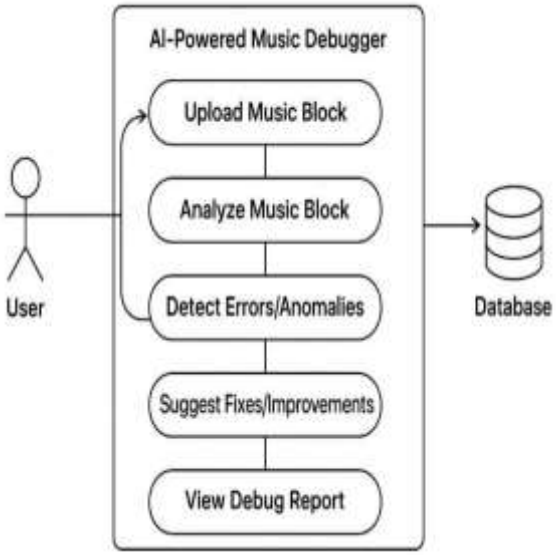
In the system design phase, the architecture of the

AI debugger is planned, and UML diagrams such as use case, data flow, and activity diagrams are prepared to illustrate system flow and interactions. After designing, the dataset collection and preprocessing stage involves gathering a diverse set of music and audio samples. These datasets are processed to remove noise, normalize signals, and extract key audio features such as MFCC, pitch, and tempo for further analysis.

The testing and evaluation phase ensures that the system performs accurately and efficiently. Multiple audio samples are tested to verify error detection precision, speed, and reliability. Based on the testing outcomes, necessary improvements are made to enhance system performance. The final stages involve documentation and reporting, where detailed records of design, implementation, and testing are prepared for submission and presentation.

Finally, provisions for future enhancement are outlined, including the integration of real-time debugging features, advanced AI models for improved accuracy, and cloud-based deployment for scalability. This structured plan ensures a systematic and efficient approach to developing a robust, intelligent, and user-friendly AI-powered music debugging system.

Usecase Diagram



Benefits and Advantages

Flexibility: The system is designed to be flexible and adaptable to different learning scenarios. It can support various levels of learners, from beginners to intermediate users, by adjusting the complexity of feedback provided. The debugging framework can also be extended to accommodate different types of music programs and block combinations within the Music Blocks environment.

ISSN: 2945-4565 VOL 05, ISSUE 01, 2026

Efficiency: By automatically identifying errors and suggesting corrective guidance, the system significantly reduces debugging time. Learners can focus more on understanding concepts rather than spending excessive time locating mistakes, leading to improved learning efficiency and productivity. Infrastructure enables real-time or near real-time responses, allowing users to experience fast and smooth performance even for computationally intensive tasks.

Adaptability: The debugger adapts to learner behavior by analyzing previous mistakes and interaction patterns. Based on learner performance, the system can modify the feedback style and difficulty level, providing a personalized learning experience. This adaptability helps learners progress at their own pace and improves long-term understanding.

Web-Based Accessibility: The proposed AI-powered debugger is implemented as a web-based system, allowing learners to access it through standard web browsers without the need for specialized software installation. This ensures platform independence and enables usage across different devices such as desktops, laptops, and tablets. Web-based deployment also supports remote and collaborative learning, making the system easily accessible in both classroom and online educational environments.

Learner-Centric Design: The project emphasizes a learner-centric approach by providing guided hints instead of direct solutions. This encourages active problem-solving and critical thinking, which are essential skills in both programming and music education.

Reduced Instructor Dependency: The system minimizes reliance on instructors for routine debugging tasks. This allows educators to focus on higher-level conceptual teaching and creative guidance, making the learning environment more effective and scalable

Research and Educational Impact: From a research perspective, the project demonstrates the effective application of artificial intelligence in music-based visual programming education. It opens new research directions in intelligent tutoring systems, adaptive learning, and AI-assisted music education.

4. CHALLENGES AND LIMITATIONS

Limited Error Coverage:
The system primarily focuses on predefined and common error patterns in Music Blocks, which may limit its ability to detect rare or highly complex logical errors.

Dependency on Training Data and Rules:
The accuracy of the AI-powered debugger depends on the quality of error rules and training data, and incomplete datasets may affect feedback precision.

Complex Musical Semantics:
Interpreting musical logic and creative intent is challenging, and the system may not fully capture subjective or artistic musical errors.

Performance Overhead:
Real-time analysis and feedback generation may introduce slight latency, especially when handling complex programs or multiple users simultaneously.

Limited Personalization:
Although adaptive feedback is supported, the level of personalization may be restricted without extensive learner data over time.

Web Dependency:

ISSN: 2945-4565 VOL 05, ISSUE 01, 2026

As a web-based system, continuous internet connectivity is required, which may limit accessibility in low-connectivity environments.

Evaluation Constraints:
The effectiveness of the system is validated within controlled educational settings, and results may vary across different learner groups or institutions.

5. FUTURE SCOPE

The future scope of this project is to extend this system by applying more sophisticated approaches to machine learning and deep learning to further improve its accuracy. This system could be further enhanced to cater to fully personalized learning scenarios based on learner behavior and performance records. In this regard, future extensions include enabling real-time collaborative debugging to allow learners to debug and collaborate with other learners with this system.

Several enhancements can be implemented to make the AI-powered debugger for Music Blocks more convenient and user-friendly. Simplifying the language used in feedback and providing clear visual indicators for erroneous blocks can help learners identify and understand mistakes more easily. Introducing a step-by-step guided debugging mode and optional one-click hint suggestions can reduce cognitive load and improve usability for beginners. Personalizing feedback based on learner proficiency levels can further enhance learning effectiveness. Additional features such as audio-based guidance, multilingual support, and progress-tracking dashboards can improve accessibility and learner engagement. Optimizing system performance for faster feedback and enabling partial offline functionality can also increase the practicality of the system in diverse educational environments.
Furthermore, it is conceivable that this system could be integrated with other existing visual programming platforms like Scratch and Blockly to extend its application to cover other learning domains besides music. In other dimensions, this system is extendible to develop a full intelligent tutoring system. In fact, further research directions could include making this system applicable to places with limited or no internet connection.

6. CONCLUSION

In this project, we explored an AI-powered debugger for block-based music coding, where programs are created by dragging and dropping visual blocks instead of typing text. The system makes coding easier, faster, and less error-prone, especially for beginners. Students and creators can focus on logic and musical creativity rather than syntax errors, while the AI debugger provides instant feedback and guidance. This approach encourages experimentation, learning, and collaboration, making it an effective tool for both education and creative music development.

The AI-powered debugger for Music Blocks provides an effective and intelligent solution to support beginner learners in music-based visual programming. By automatically detecting common programming and musical logic errors and offering context-aware feedback, the system promotes independent learning, enhances problem-solving skills, and strengthens computational thinking.
The project reduces reliance on instructors, minimizes debugging time, and increases learner engagement through an interactive and user-friendly interface. Furthermore, its scalable and

ISSN: 2945-4565 VOL 05, ISSUE 01, 2026

adaptable design ensures that it can be extended to different learning levels and future visual programming platforms. Overall, the project demonstrates the potential of integrating artificial intelligence into educational tools, contributing to improved learning outcomes and advancing research in music-based programming education.

Acknowledgment

The authors would like to express their sincere gratitude to the Department of Computer Engineering, Jagadambha College of Engineering and Technology, Yavatmal, for providing the necessary facilities and support to carry out this research work. We are thankful to our project guide and faculty members for their continuous guidance, encouragement, and valuable suggestions throughout the development of this research paper.

We also extend our appreciation to all those who directly or indirectly contributed to the successful completion of this work. The support and cooperation received during the course of this project are gratefully acknowledged.

7. REFERENCES

1. J. P. Jayavardhini et al., "AI Study Partner: Development of an LLM and Gen AI-Enhanced Study Assistant Tool," *International Journal of Scientific Research in Engineering and Management*, 2024.
2. M. Resnick et al., "Scratch: Programming for All," Communications of the ACM, vol. 52, no. 11, pp. 60–67, 2009.
3. W. Yan et al., *"How Do Elementary Students Apply Debugging Strategies in a Block-Based Programming Environment?,"* Educ. Sci., 2025 — Explores debugging strategies and challenges in block-based programming, useful for educational context of music blocks.
4. Babu, D. N., Varshini, P. K., Supriya, P. L., & Prasanna, K. L. (2025). AI-powered code debugging assistant. *International Journal of Advanced Research and Review*, 10(6), 441–447. AI tools provide real-time feedback, error explanations, and code improvement suggestions.
5. Garg, U. (2020). Exploring the use of artificial intelligence for software testing and debugging. *International Journal of Electrical Engineering and Technology*. AI techniques improve automated testing and debugging efficiency.
6. G. Fraser, K. Götz, and P. Feldmeier, "Automated test generation for Scratch programs," *Empirical Software Engineering*, vol. 27, no. 5, pp. 1–35, 2022.
7. A. Deiner and G. Fraser, "NuzzleBug: Debugging block-based programs in Scratch," in *Proc. Int. Conf. on Software Engineering (ICSE)*, 2023, pp. 1–6.
8. K. Kim, J. Yuan, and R. Hill, "Debugging during block-based programming: Common errors and strategies," *Instructional Science*, vol. 46, no. 5, pp. 1–22, 2018.
9. S. M. H. Amiri and M. M. Islam, "Enhancing programming education with an AI-powered code helper," *arXiv preprint arXiv:2509.20518*, 2025.
10. Y. Si, K. Qi, D. Li, and X. Wang, "Stitch: Step-by-step LLM-guided tutoring for Scratch programming," *arXiv preprint arXiv:2510.26634*, 2025.
11. A. Bhuekar, "Automated computational thinking assessment from visual programming artifacts using AI," *Preprints.org*, 2025.
12. S. Brennan and M. Resnick, "Computational practices in programming education," *Journal of Educational Technology*, vol. 10, no. 3, pp. 1–15, 2012.
13. S. Babu, P. Varshini, and K. Prasanna, "AI-powered code debugging assistant," *International Journal of Advanced Research and Review*, vol. 10, no. 6, pp. 441–447, 2025.

ISSN: 2945-4565 VOL 05, ISSUE 01, 2026

14. J. Kim et al., “Computational thinking and debugging knowledge in block-based programming,” *Early Childhood Research Quarterly*, vol. 65, pp. 1–12, 2023.
15. M. Hill and L. Vasconcelos, “Block-based programming environments and debugging support,” *Computer Science Education*, vol. 30, no. 4, pp. 1–20, 2020.