IntelliCode: A speech-based programming environment

Dr. Manoj Kumar*, Hitesh Yadav*, Jayant Yadav*, Mayank Jha*

* Department of Computer Science and Engineering, Delhi Technological University, New Delhi, 110042, India

***Abstract*— Studies show that nearly 2 out of 100 software developers have visual impairment or blindness yet they still yearn to do programming. Moreover, motor injuries such as repetitive stress injuries can make programming a challenging task. A wide variety of technologies have been created to help people with disabilities get over obstacles, particularly those connected to computers and technology. The term "assistive technology" pertains to software and hardware products that aid in activities such as academic paper reading and writing, interpersonal interaction, and online research. With the proliferation of assistive technology options and increased accessibility to mainstream technologies, students and workers with disabilities can now perform a broader range of activities independently. Nevertheless, people with disabilities still struggle to learn to code, as writing code remains a difficult task even after mastering the semantics and features of a programming language. To address this challenge, IntelliCode, the software proposed in this study, eliminates this barrier and assists individuals in achieving their programming goals.**

***Index Terms* – speech recognition, speech to code, code conversion, natural language processing**

I. INTRODUCTION

Surveys conducted in 2020 by World Health Organization indicate that there are an estimated 18.7 million visually impaired and blind individuals in India [1], and at least 2.2 billion worldwide [2]. Furthermore, Stack Overflow, a website used by nearly every programmer or developer all around the world, conducted a survey of over 65,000 developers and discovered that nearly 2 out of 100 developers have a visual impairment or some other disability [3], yet still desire to program on a regular basis. However, these individuals face significant challenges in doing so.

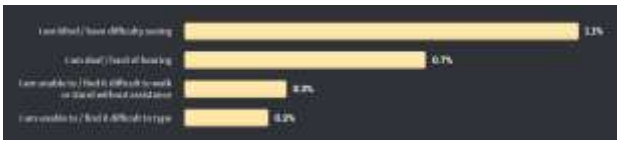

Fig. 1. Results of Stack Overflow developer survey, 2020

Numerous technologies have been developed to reduce or eliminate obstacles experienced by individuals with impairments, particularly those related to computers and technology. Assistive technology encompasses hardware and software solutions that can assist individuals with tasks like reading, writing, communication, and online information search. With the increasing availability of assistive technology and the growing accessibility of mainstream technology, individuals with impairments, whether they are students or workers, are now able to carry out a wider range of activities with greater independence. However, people with disabilities continue to face challenges when using computers, including the hurdle of learning how to code. Even after mastering the syntax and functionality of a programming language, writing code can remain a challenge. Our proposed software aims to help remove this barrier and assist individuals in achieving their programming objectives.

IntelliCode is an online coding platform designed to meet the needs of individuals with disabilities while also providing an exceptional user experience for everyone. The platform integrates speech recognition, natural language processing (NLP), code editing and code conversion to revolutionize the coding experience for individuals with visual impairments, physical disabilities, and others.

A. Background

HTML

Hyper Text Markup Language (HTML) is a vital aspect of the Internet, which serves to establish the organization and purpose of online content. It is a markup language that enables the creation of web pages that can be interpreted by web browsers. In combination with other technologies, such as Cascading Style Sheets (CSS) and scripting languages like JavaScript, HTML can be utilized to improve the functionality of web pages.

When HTML documents are retrieved from web servers or local storage, web browsers display them as multimedia web pages. HTML was originally designed to include semantic definitions of the structure of a web page as well as guidelines for how the content should be presented.

CSS

Cascading Style Sheets (CSS) is a type of style sheet language that serves to define the formatting of documents that have been written in markup languages, such as HTML or XML. CSS is one of the primary technologies utilized on the World Wide Web, along with HTML and JavaScript.

CSS allows the separation of a document's text content from its presentation, which includes elements like layout, colours, and fonts. This separation can result in improvements in content accessibility, providing greater flexibility and control over the presentation of a document. Furthermore, multiple web pages can share a single CSS file, reducing redundant coding and enabling the file to be cached for quicker page loading speeds.

JavaScript

JavaScript (JS) is a popular programming language that is widely used on the World Wide Web, alongside HTML and CSS. It is estimated that by 2022, JavaScript will be used by 98% of websites for controlling client-side functionality, often utilizing third-party libraries. All major web browsers incorporate a JavaScript engine for executing JavaScript code on users' devices. JavaScript is a high-level language that is often interpreted during runtime.

It follows the ECMAScript standard and features a prototype-based object-oriented model, first-class functions, and dynamic typing. It is a language that can accommodate multiple programming paradigms, including imperative, functional, and event-driven programming. Additionally, it offers APIs for managing common data structures, such as regular expressions, dates, and the Document Object Model (DOM).

Node.js

Node.js is a cross-platform, open-source server environment that can operate on Windows, Linux, Unix, and macOS. It provides a platform for using JavaScript on the server-side, utilizing the V8 JavaScript Engine to execute JavaScript code outside of a web browser. JavaScript, a scripting language, can be used by programmers to create server-side scripts and command-line tools for Node.js applications. By using the ability to execute server-side scripts, dynamic content can be generated before a user's web browser receives the page.

Node.js adopts a "JavaScript everywhere" approach to web application development, which allows developers to use a single programming language for both client-side and server-side scripts, instead of using different languages. Node.js is designed with an event-driven architecture that offers asynchronous I/O, enabling it to be a suitable choice for developing real-time web applications, as well as other applications that require fast and flexible input/output processing, such as browser games and real-time communication programs.

Express.js

Express.js is a web development framework for the back end of Node.js, available as open-source software under the MIT License. It enables the creation of RESTful APIs and is specifically built for developing web applications and APIs. Express.js is widely regarded as the most widely used server framework for Node.js.Express is a Sinatra-inspired server that is known for being easy to use and highly customizable through the use of plugins, according to its original inventor, TJ Holowaychuk.

It is commonly used in several development stacks, for example, Mongo-Express-React-Node (MERN), Mongo-Express-Angular-Node (MEAN), Mongo-Express-Vue-Node (MEVN), etc.

Fig. 2. Client – Server architecture

Neural Networks

One type of machine learning algorithm, referred to as neural network, emulates the human brain's structure and function. Neural networks are made up of interconnected processing nodes, also known as neurons, which collaborate to recognize patterns and draw inferences or deductions based on the input data. Neural networks are often built in layers, with each layer processing a particular component of the incoming input before transferring the findings to the layer below. Because of its hierarchical structure, neural networks are more able to understand complex patterns and make nuanced predictions than simpler, single-layer algorithms.

Deep Learning

Deep learning is a subset of machine learning that employs artificial neural networks and representation learning. Deep learning can operate in unsupervised, semi-supervised, or supervised learning modes.

The term "deep" is used because deep learning neural networks consist of multiple layers. Early research has shown that a linear perceptron cannot be a universal classifier, but a neural network with one hidden layer of infinite size and a nonpolynomial activation function can be. Deep learning, which is a more recent approach, focuses on using multiple layers of defined sizes, which allows for fast implementation and practical application while still maintaining theoretical universality under certain conditions.

OpenAI

OpenAI is a research institution focused on advancing artificial intelligence (AI) technology. AI-enabled software and machine learning algorithms now have the ability to perform a variety of tasks, such as generating images from text and designing a robotic hand that can solve Rubik's Cubes.

Their most recent project, OpenAI Codex, promises to make programming tools and apps more accessible to the general public while cutting down on the time and effort expert programmers must expend while writing code.

The GPT-3 language generation model from OpenAI was used to create the AI coding tool known as Codex, which functions as a translator between humans and computers. In early demos, users were able to create simple webpages and games without utilising a particular programming language by using plain English or natural language.

GPT-3

Generative Pre-trained Transformer 3 (GPT-3) is an

autoregressive model that employs deep learning to generate writing that is similar to human writing. By giving an initial text as input, GPT-3 can produce output text that continues the given prompt.

GPT-3 has both advantages and downsides. It creates text of such high quality that it can be difficult to identify whether it was created by a person. GPT-3 was hailed as "one of the most exciting and significant AI systems ever constructed" by Australian philosopher David Chalmers. According to a review that appeared in The New York Times in April 2022, GPT-3 is capable of writing creative language with a level of fluency that is equivalent to that of a human.

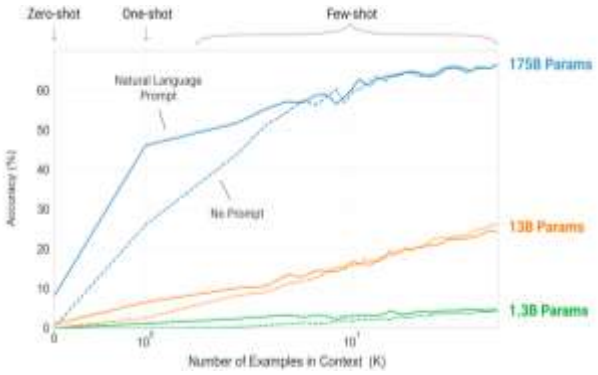

Fig. 3. GPT-3 parameter curve

B. Related works

In recent times, modern technologies such as machine learning, deep learning, and natural language processing have sparked interest in integrating human voice into programmable code. However, using conventional natural language processing methods, automated speech recognition has faced challenges in effectively performing programming tasks. Text editors usually come with vendor-provided editing commands that cater to word processing and permit style alterations and clipboard access.

According to sources, Jeff Gray integrated speech capabilities into the Eclipse programming environment, although not the editor [4]. VoiceCode [5] is a tool that has shown impressive results with finite-state command grammars that are compatible with Python and C++. However, it has not undergone a thorough evaluation yet. NaturalJava [6] accepts programmers' natural language descriptions of the Java constructs they want to create instead of coding them directly. NaturalJava only caters to code authoring and not editing or navigation. SPEED [7] is a significant contribution to the field of speech-based coding, but it only supports Java and Eclipse programming IDE.

Our study on IntelliCode is a first-of-its-kind examination voice-based programming in a practical situation.

C. Contributions

The key functional contributions of the proposed software are as follows:

- Users can simply specify the programming steps in plain English and create complete code in the language of their choice using the speech-to-code capabilities. People with vision problems or hand deformities will be able to code faster and be more productive because of this.
- Users have the option to select a pre-defined algorithm, such as sorting algorithms, binary search, and tree traversal, which the software can implement in either Python or JavaScript.
- Additionally, the software has the capability to debug, make typing corrections, and ensure proper termination of the program.
- The code conversion feature allows for quick translation of code from one language to another (primarily from JavaScript to python and vice versa), saving users a significant amount of time and money.
- Finally, we intend to build IntelliCode as a completely online collaborative coding platform that will create a welcoming and inclusive atmosphere for differently abled persons while also allowing them to connect effectively with their co-workers.

D. Roadmap

The remainder of the paper is divided into three sections. Section II details the methodology employed in the proposed work, while Section III assesses and analyzes the results. The paper concludes with Section IV, which expounds upon the conclusions drawn from the research and identifies future areas of work.

II. METHODOLOGY

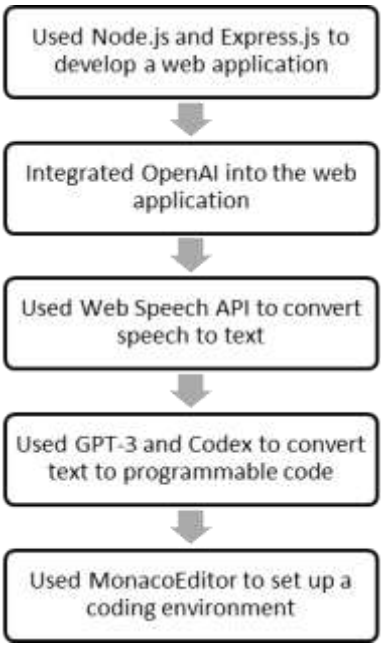

Fig. 4. Overview of methodology

A. Server setup

Setting up a server with Express.js is a very easy task. Node provides support to write JavaScript on the backend side of the application and Express.js provides the support to route the incoming requests to the Node server and take appropriate action.

Node.js comes with a node package manager, short for npm, and is used to download and install all the external dependencies that one might need for a node application. Below table shows the dependencies required in this application along with their versions.

TABLE I. DEPENDENCIES USED IN THE NODE SERVER

Dependencies	Version
dotenv	16.0.0
express	4.17.3
openai	2.0.5

B. OpenAI Integration

A developer may use OpenAI to access GPT-3, which can carry out a number of tasks using natural language, and Codex, which converts natural language into code. Modern language models that have been taught by OpenAI are excellent at comprehending and producing text. These models are accessible through OpenAI API, which may be used to do almost any task involving language processing.

The OpenAI npm library must be downloaded in order to use the OpenAI API. Once created, an OpenAI object must be linked with an API key so that it may be used to query the OpenAI server for API endpoints and utilise the capability for tasks involving natural language processing.

C. Web speech API Integration

Using JavaScript and the built-in Online Voice API offered by the current Web API, a user may include speech data into web programmes. The Web Speech API consists of two components: SpeechRecognition and SpeechSynthesis (Asynchronous Speech Recognition).

Utilizing the device's default speech recognition service, the SpeechRecognition interface allows users to access speech recognition, which enables them to comprehend the context of a voice input and reply appropriately.

D. Text to Code Conversion

The OpenAI API offers a feature referred to as "completion". There are numerous tasks that may be performed using the "completions" endpoint. It offers any OpenAI model a straightforward yet effective interface. As a prompt, we can enter some text, and the model will provide a text completion that tries to fit the context or pattern we provided.

Using completion API, we can accurately translate a spoken utterance into an English statement.

E. Code conversion

The "completion" endpoint of OpenAI Codex can be used to access the Codex and convert code between languages. Since the endpoint is developed in Python, it needs a few PyPI (Python Package Index) libraries, which may be downloaded by using the "pip" command. Below is a list of those packages in more depth.

TABLE II. DEPENDENCIES USED IN THE OPENAI APPLICATION

Dependencies	Version
certifi	2020.6.20
charset-normalizer	2.0.9
colorama	0.4.4
decorator	4.4.2
idna	3.3
imageio	2.13.1
imageio-ffmpeg	0.4.5
moviepy	1.0.3
numpy	1.19.5
Pillow	8.4.0
proglog	0.1.9
requests	2.26.0
SpeechRecognition	3.8.1
tqdm	4.62.3
urllib3	1.26.7
Wincertstore	0.2

III. RESULTS AND DISCUSSION

The participants in the study were experienced developers and programmers having profound knowledge of python and JavaScript. Most of them had never used any speech-to-code software before. None of them had any disabilities which would make typing difficult. The study comprised 8 programmers who were allotted 15 minutes each to develop a binary search program for finding an element in an array. Any challenges faced during the coding process were attributed to the usage of IntelliCode rather than the program creation itself.

The study was conducted over three sessions that were identical except for the voice recognition technology used. The first session utilized IntelliCode, which provided an accurate natural language to code conversion model. The second session involved the use of SPEED software developed by A. Begel and S. L. Graham, while in the third session, a non-programmer human typed in the code.

The third session demonstrated how IntelliCode could be leveraged for its near-perfect accuracy in natural language to code conversion.

Various metrics, as shown in Table III, were used to analyse the programming sessions of the users. A single viewer watched the screen recording while listening to the audio transmission to code the metrics. The results were then compared with those of the SPEED software.

Speed and accuracy

During the three sessions, the programmers had varying experiences. Some participants struggled due to slow speed and low accuracy, resulting in only partial completion of the code creation task. Others were able to finish their work with time to spare. The group that used a human voice recognizer was able to complete more of the task in less time.

However, there were some issues with the recognizer, such as incorrect word recognition and occasional recognition of unintended words. To address this, some programmers muted their microphones while dictating to themselves.

Despite these challenges, the SPEED software demonstrated a recognition delay of around 0.7 seconds, while IntelliCode had a recognition delay of approximately 1 second. Additionally,

IntelliCode had a higher recognition accuracy of approximately 70%, compared to SPEED's overall accuracy of 60%.

Speaking code

The users had a relatively low rate of command errors caused by user mistakes. These errors occurred mainly when users were unsure of what to say or how to express themselves. As these errors were primarily the users' own, rather than resulting from misrecognition, they were deemed more acceptable to the users. The voice recognizer malfunction sometimes caused programmers to speak in short bursts, and correcting incorrect words was time-consuming.

Participants were initially hesitant to use natural language when dictating code, but not when using identifier names. They believed that a combination of voice and keyboard would be more effective than either alone.

Subjective evaluation

When questioned about the possibility of using voice recognition for programming, the majority of participants expressed reservations. They cited concerns such as noise pollution, cognitive interference, and the inability to use their hands while speaking. However, some participants mentioned that they might consider using it at home if they worked alone and didn't disturb anyone else.

Ultimately, the participants concluded that they would not prefer to use this programming environment for daily coding. However, they all agreed that they would consider using the software if they had an injury that limited their use of hands or eyes or if they were working in a hands-free environment.

TABLE III. DATA RECORDED FROM INTELLICODE USER STUDY FROM ALL PARTICIPANTS

		All Utterances	Correctly Recognised	Error Rate
IntelliCode	***U1***	68	47	31%
	U2	61	44	27%
	U3	73	47	36%
SPEED	***U4***	80	60	27%
	U5	103	51	50%
	U6	89	50	43%
Human Voice Recognition	***U7***	51	43	15%
	U8	55	42	23%

IV. CONCLUSION

IntelliCode is a software application designed to assist programming through the use of voice input. It accepts a user's spoken commands and generates code in response. The software can create code snippets or macros based on the voice input, and it can also facilitate code conversion between different programming languages, primarily Python and JavaScript. Through a user study, we evaluated the software and discovered that programmers can learn to program using voice commands with minimal practice. However, they face significant challenges when the speech recognizer misunderstands their speech.

The platform integrates speech recognition, natural language processing (NLP), code editing, and code conversion to enhance the coding experience for individuals with visual impairments, physical disabilities, and others. Programming-by-voice can also enable software engineers with motor impairments to program, although their efficiency may be lower compared to an unimpaired programmer. With additional research, improved user interface designs, and better analysis, speech-based programming could become a competitive alternative for software developers in the future.

A. Future Scope

The progress in machine learning algorithms that can efficiently process natural language and text-to-code is likely to play a significant role in the future of speech-to-code applications.

Moreover, a major area of focus will be to support new programming languages such as C/C++, Java, Rust, and Go. However, real-time programming, which involves complex coding techniques, poses a challenge as the application can only perform basic speech recognition with good accuracy currently. As the complexity of speech and code increases, so does the error rate of the application. The software supports speech-to-code functionality, but users still have to use built-in buttons to convert speech to code and then execute it. Integrating AI with this feature could allow users to use voice commands to convert and run code.

A computer program can be prone to errors as programming involves both syntactic and semantic typing. The potential for errors increases when using AI to translate voice to code. In the future, we plan to incorporate a feature that enables the AI to identify errors and warnings and suggest fixes. We also aim to introduce a debugging mode that allows users to specify the line number and desired modifications, which the software can execute.

As artificial intelligence and natural language processing continue to evolve, they will significantly impact the functionality and accuracy of our speech-to-code application.

ACKNOWLEDGMENT

Our academic achievements and the success of our project were made possible by the contributions of many people. We would like to express our gratitude to all those who helped us. Specifically, we are indebted to Dr Manoj Kumar for his invaluable supervision and guidance throughout the project. His leadership and inspiration have been instrumental in our progress. We also want to extend our thanks to the department faculty members for their continuous support and encouragement. Lastly, we appreciate the assistance provided by our parents during these challenging times.

REFERENCES

[1] Dandona L, Dandona R, John RK. Estimation of blindness in India from 2000 through 2020: implications for the blindness control policy. Natl Med J India. 2001 Nov-Dec;14(6):327-34. PMID: 11804362.

[2] WHO Blindness and vision impairment: https://www.who.int/news-room/fact-sheets/detail/blindness-and-visual-impairment

[3] Stack overflow developer survey:

https://insights.stackoverflow.com/survey/2020#developer-profile-developer-type-united-states-unweighted

[4] S. Shaik, R. Corvin, R. Sudarsan, F. Javed, Q. Ijaz, S. Roy-choudhury, et al., "SpeechClipse: an Eclipse speech plug-in", Eclipse Technology eXchange Workshop, pp. 84-88, 2003

[5] A. Desilets, D. C. Fox and S. Norton, "Voicecode: An innovative speech interface for programming-by-voice", Proceedings of ACM CHI 06 Conference on Human Factors in Computing Systems, April 2006.

[6] Price, E. Rillof, J. Zachary, and B. Harvey. Natural Java: A natural language interface for programming in Java. In IUI, Short Paper/Poster/Demonstration, pages 207–211, 2000.

[7] A. Begel and S. L. Graham, "An Assessment of a Speech-Based Programming Environment," Visual Languages and Human-Centric Computing (VL/HCC'06), 2006, pp. 116-120, doi: 10.1109/VLHCC

AUTHORS

First Author – Dr. Manoj Kumar, Professor, Delhi Technological University, Delhi, India, 110042,
Second Author – Hitesh Yadav, Student, Delhi Technological University, Delhi, India, 110042,
Third Author – Jayant Yadav, Student, Delhi Technological University, Delhi, India, 110042,
Fourth Author – Mayank Jha, Student, Delhi Technological University, Delhi, India, 110042,

Correspondence Author – Hitesh Yadav, Student, Delhi Technological University, Delhi, India, 110042,