An intelligent system for speech analysis and control using customized criteria

Zoriana Rybchak*,†, Olha Kulyna*,†, Leslav Kobyliukh*,†

Lviv Polytechnic National University, S. Bandery street 12, Lviv, 79000, Ukraine

Abstract

This article presents an intelligent system designed to analyze and control speech based on user-defined criteria, with the objective of enhancing communication skills through insightful data analysis. Leveraging Python libraries such as PyAudio, Vosk, Pandas, and Plotly, the system enables audio recording, speech-to-text conversion, data management, and visualization of speech patterns.

The study explores effective speech recognition methods and algorithms for audio processing and text analysis, including keyword detection and segment analysis. Visualizations generated by the system offer users a clear understanding of their speech dynamics over time. The software features an intuitive interface to ensure widespread usability. Key functionalities include speech recording, processing, unwanted word management, audio playback, and chart creation. This research contributes a comprehensive speech analysis application utilizing modern techniques to provide actionable insights for improving spoken language proficiency.

Keywords

Speech analysis, intelligent software, speech recognition, data visualization, audio data, Speech-to-Text

1. Introduction

The impact of the Internet and social media on the state of the Ukrainian language has been profound. These platforms offer a unique opportunity for communication and professional activities in Ukrainian. However, despite efforts to promote the language, the unconscious use ofRussian remains prevalent, particularly in eastern and southern regions. Additionally, issues persist in sectors such as advertising and online content, hindering the full adoption of Ukrainian.

MoDaST-2024: 6th International Workshop on Modern Data Science Technologies, May, 31 - June, 1, 2024, Lviv-Shatsk, Ukraine

* Corresponding author.

† These authors contributed equally.

zoriana.l.rybchak@lpnu.ua (Z. Rybchak); olha.v.kulyna@lpnu.ua (O. Kulyna); leslav.b.kobyliukh@lpnu.ua (L. Kobylyukh)

0000-0002-5986-4618 (Z. Rybchak); 0000-0002-2334-0660 (O. Kulyna); 0009-0006-5653-4943 (L. Kobylyukh)

Between 2014 and 2022, Ukraine underwent significant changes and reforms aimed at elevating the role and status of the Ukrainian language within society. Nevertheless, language- related challenges persist and require ongoing attention. The Ukrainian-speaking population is estimated at around 42 million, including both residents of Ukraine (approximately 37 million) and members of the global Ukrainian diaspora.
The Ukrainian language encompasses various forms, including Literary Ukrainian—used in literature, science, media, and official contexts—vernacular Ukrainian for everyday speech, regional dialects with unique vocabulary and grammar, and Surzhyk, a blend of Ukrainian and Russian spoken in specific areas.

Despite the linguistic diversity, Ukrainian remains the predominant language of daily communication for over 80% of Ukraine's population. Beyond Ukraine's borders, the total number of Ukrainian speakers reaches 45-50 million [1]. By other sources this date drops down to 67 % [2]. Notably, certain regions, like Donetsk and Luhansk oblasts, have sizable Russian- speaking communities. This study aims to develop an intelligent system for analyzing and monitoring speech basedon user-defined criteria.

Key objectives of the article include:

1. Assessing the current state of the Ukrainian language using online sources and literature.
2. Identifying linguistic challenges and justifying their significance.
3. Developing conceptual models employing UML methods.
4. Selecting appropriate tools and methods to address these challenges.

The research focuses on the Ukrainian language as a medium of communication and a vital part of Ukraine's cultural heritage. Specific topics of investigation encompass grammatical and lexical variations across regions, language proficiency among Ukrainians, and the cultural and educational impact of linguistic developments.

The study's scientific novelty lies in presenting the Ukrainian language as a target for controlled analysis, adaptable to user-defined criteria. The outcomes will inform new methods of language analysis and control, tailored to individual user needs. Identifying prevalent speech errors will facilitate the formulation of recommendations to enhance communication quality in Ukrainian.

2. Research methods and materials

To enhance the implementation of our language analysis and control intelligent system based on user-defined criteria, we employed a selection of tools including Visual Studio Code, Python, SQLite, Vosk Models, and Excel. Here's a concise overview of their roles and considerations for future alternatives [3; 4; 5; 6].

Visual Studio Code (VSCode) was chosen as the primary development environment due to its cross-platform compatibility, extensibility, integrated terminal, version control integration, and customizability. Alternative options like PyCharm, Sublime Text, and Jupyter Notebooks offer specific advantages but were not as suitable for this project's needs.

A versatile and widely popular programming language, Python was chosen for its simplicity, readability, and extensive library ecosystem. It's ideal for the language analysis and control system due to its clarity, vast library support, compatibility across platforms, active community, and integration with advanced technologies like machine learning. Although other languages like Java and Swift offer benefits for specific platforms (Android and iOS), Python was selected for the initial Windows version of the program.

SQLite is selected for its self-contained and lightweight nature and serves as the database for user data storage. While alternatives such as PostgreSQL, MySQL, or MongoDB offer enhanced features and scalability, SQLite meets the current requirements efficiently.

Vosk Model, an automatic speech recognition (ASR) library, is integrated for speech-to-text conversion, real-time speech analysis, and word recognition within the system. Its deep learning capabilities enable accurate transcriptions.

Excel:Initially used for storing user speech data due to its ease of use, compatibility, and built-in functionalities for data manipulation and analysis. Future considerations may involve transitioning to relational databases, cloud storage platforms, or specialized data analysis tools for enhanced productivity and scalability.

Cloud platforms for data storage and analysis, such as Google Sheets or Microsoft OneDrive, offer real-time collaboration and remote access to data, which can be beneficial for distributed teams and projects [7; 8].

Specialized software tools for data analysis, like R or Python with pandas, provide advanced capabilities for data processing, analysis, and visualization.

These tools can handle large datasets and offer extensive libraries for various data processing tasks, making them suitable for more complex analyses.

In the future, depending on project requirements and scale, transitioning from Excel to a more robust and flexible data management solution could be beneficial to enhance productivity, security, and analytical capabilities of the system.

The language analysis and control system aims to detect and prevent the use of undesirable words in user speech, enhancing communication quality and fostering a healthy linguistic environment.

The system can be implemented in various settings such as educational institutions, workplaces, social media, and internet platforms, and can integrate with devices like smartphones, computers, and tablets.

The system is developed to improve communication quality and reduce the negative impact of unwanted words. It promotes positive linguistic culture and enhances awareness of language norms and etiquette.

While our current toolset aligns well with the project's initial objectives, ongoing enhancements may involve transitioning to more advanced solutions to accommodate evolving needs and scalability. This approach ensures continual optimization and adaptation of our language analysis and control intelligent system.

3. System analysis and justification of the problematic situation

Using a goal tree is a crucial step in creating and designing a system for several reasons [9]:

1. Clarity and Structure: A goal tree helps break down a complex project into smaller, manageable parts, facilitating a better understanding of tasks and related requirements. This creates a clear structure for the project.
2. Priority Setting: With a goal tree, priorities for different tasks and resources can be established, enabling efficient allocation of attention and resources during system development.
3. Communication and Coordination: The goal tree serves as a common basis for discussion and coordination among team members, facilitating the exchange of ideas and mutual understanding among project participants.
4. Progress Tracking: A goal tree can be used to track project progress at different development stages, allowing the team to monitor the achievement of individual sub-goals and make necessary planning adjustments if needed.
5. Risk and Opportunity Assessment: Analyzing the goal tree helps identify potential risks and opportunities that may arise during project implementation.

Furthermore, the sub-goals of the goal tree will be outlined, aiding in breaking down the main goal into smaller and more specific objectives, thus providing a better understanding of the necessary steps to achieve the project's main goal.

The third step involves establishing criteria. Goal tree criteria are specific metrics that help measure the degree of achievement of each sub-goal.

Criteria can be qualitative (e.g., customer satisfaction) or quantitative (e.g., reduction in time, costs, or errors).

The importance of sub-goals and criteria in the goal tree lies in their ability to thoroughly plan and detail the project, making it manageable and controllable. Here are the user-defined criteria of a goal tree for a language analysis and monitoring system : Clarity (C1), Accessibility (C2), Realism (C3), Relevance (C4), Flexibility (C5), Effectiveness (C6).

To achieve this goal, the following sub-goals will be pursued: Audience definition; Determination of the system's primary objectives; Selection of methodology and technologies for system implementation; Analysis of articles, scientific publications, and other sources; Examination of key issues and identification of optimal problem-solving methodologies;

Development of models using UML methodologies; Programming internal components of the software; Designing the database; Developing the external interface; Testing the developed system; Providing system support and updates.

The next step is to specify and select appropriate criteria for the sub-goals.

Clarity was identified as the primary and critical criterion for initiating the system's implementation.

Accessibility was chosen as the criterion for selecting methodologies and technologies, emphasizing the importance of general availability and ease of use.

Realism is a vital criterion for studying key issues and finding optimal problem-solving methodologies, as more realistic methodologies yield better final solutions.

Correspondence was chosen for the sub-goal of creating models using UML methodologies.

Flexibility and efficiency were selected for developing internal software components and ensuring system support and updates, respectively.

Next, the Analytic Hierarchy Process (AHP) method will be applied, considering the goal tree and selected criteria for the sub-goals, involves defining criteria, such as Clarity (C1), Accessibility (C2), Realism (C3), Correspondence (C4), Flexibility (C5), and Efficiency (C6), and matching them with valid information systems types, including Analytical (A1), Knowledge Management (A2), Decision Support (A3), and Search (A4) systems, based on their functionalities and relevance to language control [10].

To create a pairwise comparison matrix of criteria, a list of criteria to be compared was defined. Next, for each criterion pair, it is necessary to determine which criterion has a higher priority based on a given assessment.

This will be done using a scale from 1 to 9, where 1 indicates that the first criterion has lower priority and 9 indicates that the first criterion has higher priority (Table 1).

Table 1
Importance of priorities

Value	Qualitative characteristic
1	Uniformity
2	Very weak
3	Weak
4	Somewhat weak
5	Neutral or average
6	Somewhat strong
7	Strong
8	Very strong
9	Absolute or very high

Now, a matrix is constructed to display expert assessments (Table 2). The column and row names will correspond to the criteria mentioned earlier. Additionally, columns will be added to calculate eigenvalues and eigenvectors.

Table 2
The matrix of expert assessment

	C1	C2	C3	C4	C5	C6
C1	1	7	2	3	2	3
C2	6	1	6	4	5	2
C3	2	6	1	8	2	2
C4	3	5	8	1	5	3
C5	2	5	2	5	1	8
C6	3	2	2	3	8	1

Let's calculate the eigenvalues using the given formula (2.1):

$$EN(C1) = \sqrt[6]{1 * 7 * 2 * 3 * 2 * 3} = 2{,}51$$
$$EN(C2) = \sqrt[6]{6 * 1 * 6 * 4 * 5 * 2} = 3{,}36$$
$$EN(C3) = \sqrt[6]{2 * 6 * 1 * 8 * 2 * 2} = 2{,}70$$
$$EN(C4) = \sqrt[6]{3 * 5 * 8 * 1 * 5 * 3} = 3{,}49$$
$$EN(C5) = \sqrt[6]{2 * 5 * 2 * 5 * 1 * 8} = 3{,}05$$
$$EN(C6) = \sqrt[6]{3 * 2 * 2 * 3 * 8 * 1} = 2{,}57$$

As mentioned above, it is also necessary to calculate the values of the eigenvectors. For this, we will use the following formula:

$$EV = \frac{w_i}{\sum_{i=1}^{n} w_i}, \tag{2.1}$$

We calculate the denominator as follows:

2,51 + 3,36 + 2,70 + 3,49 + 3,05 + 2,57 = 17,67

Now, to calculate each vector, we use the following calculations:

$$EV(C1) = \frac{2{,}51}{17{,}67} = 0{,}14; EV(C2) = \frac{3{,}36}{17{,}67} = 0{,}19;$$

$$EV(C3) = \frac{2{,}70}{17{,}67} = 0{,}15; EV(C4) = \frac{3{,}49}{17{,}67} = 0{,}20;$$

$$EV(C5) = \frac{3{,}05}{17{,}67} = 0{,}17; EV(C6) = \frac{2{,}57}{17{,}67} = 0{,}15.$$

In the next step, we will conduct a similar manipulation for alternatives. Accordingly, the size of the matrix will be 4*4 and two additional columns of vectors. Table 3 shows a matrix of comparisons for alternatives according to the clarity criterion.

Table 3
Alternatives C1 (Clarity)

	A1	A2	A3	A4	EN	EV
A1	1	2	5	7	2,89	0,29
A2	2	1	2	5	2,11	0,21
A3	5	2	1	2	2,11	0,21
A4	7	5	2	1	2,89	0,29

Table 4
Alternatives C2 (Accessibility)

	A1	A2	A3	A4	EN	EV
A1	1	3	5	8	3,31	0,31
A2	3	1	3	4	2,45	0,23
A3	4	3	1	2	2,21	0,20
A4	8	4	2	1	2,83	0,26

We will now calculate the third matrix for the criterion of realism and record the results in Table 5

Table 5
Alternatives C3 (Realism)

	A1	A2	A3	A4	EN	EV
A1	1	5	2	3	2,34	0,23
A2	5	1	3	3	2,59	0,25
A3	2	3	1	7	2,55	0,25
A4	3	3	7	1	2,82	0,27

For the criterion of relevance, we will compute everything analogously. We will also determine the eigenvectors and eigenvalues. The results are shown in Table 6

Table 6
Alternatives C4 (Relevance)

	A1	A2	A3	A4	EN	EV
A1	1	6	3	9	3,57	0,30
A2	6	1	2	4	2,63	0,22
A3	3	2	1	4	2,21	0,19
A4	9	4	4	1	3,46	0,29

For the penultimate criterion, the computation flexibility is analogous Table 7.

Table 7
Alternatives C5 (Flexibility)

	A1	A2	A3	A4	EN	EV
A1	1	6	3	9	3,57	0,30
A2	6	1	2	4	2,63	0,22
A3	3	2	1	4	2,21	0,19
A4	9	4	4	1	3,46	0,29

The last criterion for which the matrix will be computed is efficiency. The result is presented in Table 8.

Table 8
Alternatives C6 (Efficiency)

	A1	A2	A3	A4	EN	EV
A1	1	8	5	6	3,94	0,27
A2	8	1	7	4	3,87	0,26
A3	5	7	1	5	3,64	0,25

A4	6	4	5	1	3,31	0,22

The final step involves creating comparisons among alternatives based on the previously computed values of each alternative relative to each criterion.

For this purpose, a matrix is used where the alternatives are indexed as rows and the criteria as columns, respectively. The first row is filled with the data of each alternative concerning the corresponding criteria computed in the previous stage. Each column corresponds to the data of each criterion, which was also computed in the previous stage using the eigenvalues.

Therefore, to calculate this, we will use the following formula (2.3) and record the result in Table 9.

Formula:

$$E\sum_{j=i}^{n} a_{ij} * k_j, \tag{2.3}$$

Table 9
Alternatives (A) and Criteria (C)

	C1	C2	C3	C4	C5	C6	Summarised priorities
	0,14	0,19	0,15	0,20	0,17	0,15	
A1	0,29	0,31	0,23	0,30	0,31	0,27	0,286255828
A2	0,21	0,23	0,25	0,22	0,12	0,26	0,21430797
A3	0,21	0,20	0,25	0,19	0,26	0,25	0,224890991
A4	0,29	0,26	0,27	0,29	0,30	0,22	0,274545212

After analyzing the results and using the hierarchy method to select the best type of system for the project, very similar results were obtained for all four alternatives. However, the first alternative, which has an analytical nature, proved to be the best. Therefore, an information- analytical system will be designed.

Next, the conceptual model construction using UML methods typically starts with defining key concepts and relationships among them. This is done by analyzing the system and identifying its main elements. Now, appropriate UML diagrams need to be selected to visualize these elements and their relationships. For instance, use case and activity diagrams can be used for the analysis stage, class, sequence, collaboration, and state diagrams for the design stage, and component and deployment diagrams for the development stage. It's important to build the conceptual model using the relevant UML diagrams and tools, validating and checking the model considering user needs and system requirements. The final step involves documenting the conceptual model and using it for further system design or process [11; 12; 13].

Additionally, actors and use cases can be defined to model the functionality of the system from a user's perspective. Use Case Diagrams, for example, help depict interactions between the system and its users or external systems within specific interaction scenarios. They consist of actors, use cases, and relationships illustrating dependencies and interactions between actors and use cases. This diagram aids in understanding what functions the system should perform to meet user needs and identifies interactions between different parts of the system and external systems.

A State Diagram visually models system or object behavior based on current states. It's useful for complex systems where objects have stable states and respond to state changes. The diagram includes states, transitions (state changes triggered by events), and events (stimuli causing state transitions).

Users start from an initial state and move to "Choose Option" to select system actions. Choosing "Parameters" leads to "Await Initial Parameters" where necessary settings are specified. Then, it moves to "Await Voice Element" and "Recognize Voice Data" after a voice sample. Data goes to the Sublimator, then "Sublimate Data" before processing in the database ("Process Z-Words"). After setting parameters, it moves to "Await Confirmation."

Choosing "Analysis" goes to "Record Voice," then "Authenticate and Recognize Speech." Data moves to the Sublimator, then "Sublimate Data," and processes in the database ("Compare New Data with Database"). A successful word match goes to "Confirm Z-Word Usage" and signals the user, returning to "Record Voice" before ending. If no match, it returns directly to the final state.

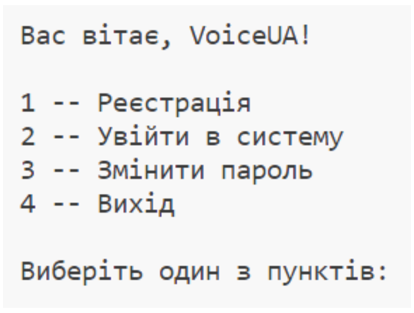
4. Results and Discussions

The software is developed using the Python programming language and includes the following main modules:

1. User authentication and registration module
2. Audio recording and analysis module
3. Live speech analysis module
4. Word list correction module
5. Audio file playback module
6. Data analysis creation and display module

Let's delve into implementing each module, considering the involvement of auxiliary programs and system components.

Upon encountering the system, the user is presented with a welcome window offering four possible options:



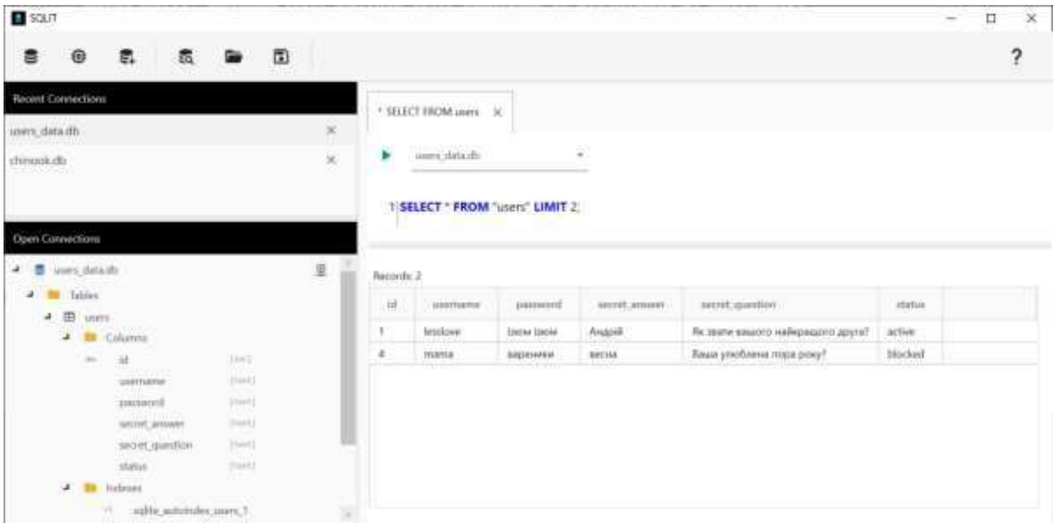
```
Вас вітає, VoiceUA!

1 -- Реєстрація
2 -- Увійти в систему
3 -- Змінити пароль
4 -- Вихід

Виберіть один з пунктів:
```

Figure 1: Initial window of the intelligent system.

Registration initiates a complete registration process where the user creates a profile. The register() function initializes speech recognition using sr.Recognizer() and establishes a connection with an SQLite database using sqlite3.connect(dbpath). The user enters their login, and if the login is unique, continues with security questions and password setup. The system processes the password by recognizing the spoken input, saving it securely in the database, and recording user details (username, password, security question, and default status: active) in the user table. Additionally, the system creates a user folder for recordings and an Excel file for texts and words-Z, notifying the user upon successful registration. If the password isn't recognized, the user can retry [3; 14].



```
1 SELECT * FROM "users" LIMIT 2;
```

Records: 2

id	username	password	secret_answer	secret_question	status
1	lesslove	[illegible]	Андрій	Як звати вашого найкращого друга?	active
4	mama	вареники	весна	Ваша улюблена пора року?	blocked

Figure 2: Users' data.

The program automatically creates a folder for user records and an Excel file for their texts and words-Z, initializing it with a template file, and notifies the user of successful registration. If the password is not recognized, an error message is displayed, and the user is prompted to retry. Immediately after registration, the system calls the login() function to direct the user to the login procedure.

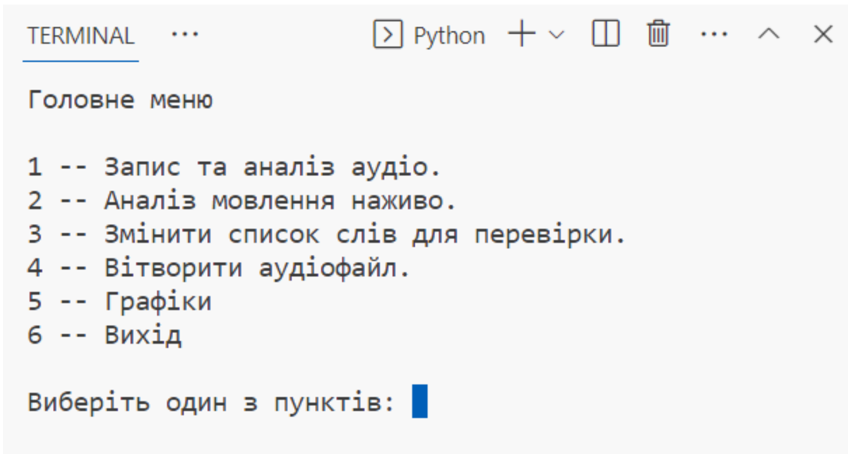
Upon selecting this step or completing the registration, the user is redirected to the login window. A speech recognition object is again created, initializing the number of allowed authentication attempts to 3. The current path is then set, and a connection to the SQLite database is established. The system prompts the user to enter their username (login). The program retrieves user data from the database corresponding to the provided username (login). If the user is found in the database, the system checks the user's status:

If the user is blocked, the system informs the user and exits the loop.

If the user is not blocked, the system prompts the user to speak their password. Next, the system uses speech recognition to listen to the audio and convert it to text using the Google Speech Recognition API. If the recognized password matches the stored password in the database, the user is notified of successful authentication and their username is returned. If the password does not match, the number of attempts is decreased, and the user is informed of an incorrect password. If the user is not found in the database, the number of attempts is decreased, and the user is informed that the user was not found.

Change Password allows the user to update their password after authentication via a security question and speech-based password input. If the answer matches the registered response, the password is updated, and the user is notified of the successful change.

After successful login, the system presents the main menu with five usage options, each employing specific tools and techniques.



```
Головне меню

1 -- Запис та аналіз аудіо.
2 -- Аналіз мовлення наживо.
3 -- Змінити список слів для перевірки.
4 -- Вітворити аудіофайл.
5 -- Графіки
6 -- Вихід

Виберіть один з пунктів:
```

Figure 3: The main menu of the intelligent language analysis and control system variesaccording to the criteria specified by the user.

Module for Audio Recording and Analysis describes the implementation of an audio recording and analysis option (option1) that records and analyzes audio input provided by the user through speech. The function performs the following actions:

1. Retrieve the current file path and set a directory for storing recorded audio files.
2. Start a loop that continues until the user decides to stop.
3. Generate a file name based on the current date and time of recording.
4. Record the user's audio input using the record_audio function, which saves the audio in the specified directory.
5. Convert the recorded audio to text using the audio_to_text function.
6. Display the transcribed text in the console.
7. Obtain a list of words for analysis from the user's Excel file using the check_word_list function.
8. Search for specified words in the transcribed text using the analyze_text function, which returns a list of found words and their count.
9. Save the transcribed text and analysis results to the user's Excel file using the save_to_excel function.
10. Provide the user with two options: return to the main menu or start the analysis again.

The loop continues based on the user's choice [15].

The module for audio recording and analysis utilizes speech recognition, text analysis, and Excel file manipulation to allow users to record their speech, transcribe it into text, search for specific words in the text, and save the results to an Excel file. This function is part of a program that offers various user interactions and data processing capabilities.

Real-time Speech Analysis Module analyzes a user's speech in real-time. If the user's speech contains certain predefined words, the program notifies the user about the use of unwanted words. The process_speech function, responsible for this implementation in the code, is designed to perform real-time speech recognition and detect unwanted words in a specified list [16; 17].

Module for Word List Correction manages function (option 3). It implements the process of correcting word lists (Z). To delve into the algorithm of this procedure, the modify_excel_column function is described for interactive editing of a specific column in an Excel file. The function accepts three parameters: file_name: the name of the Excel file;

sheet_name: the name of the sheet within the Excel file; column_name: the name of the column to be edited.

The audio playback and analysis module is defined by the play_audio_file function, enabling users to play audio files from a specific folder based on their recording date. The function utilizes the pygame library for audio playback and the keyboard library to detect keyboard keypresses [18].

First, the pygame.mixer.init() function initializes the pygame mixer module for audio playback. Then, the data directory is defined using os.path.dirname(os.path.abspath(file)) to obtain the script directory path and os.path.join() to create the path to the user's recordings folder. A list of audio files (in .wav format) in the user's recordings folder is created using the list method.

The function implements a nested loop structure to create a menu-based interface for the user. The outer loop handles date selection, while the inner loop handles audio file selection. This structure allows users to navigate between two menus until they choose to exit. An important aspect of this system is that the function recognizes audio files by extracting and analyzing their names (assuming the date is in the first 8 characters of the file name). Therefore, the files created with option1 are named based on the recording date. The dates are then sorted in ascending order.

The function displays a list of unique recording dates and prompts the user to select a date by entering a number. If the user enters "0", the function exits the loop and clears the screen. For the selected date, the function filters audio files whose names start with the selected date and displays the list of audio files for that date.

Next, the system prompts the user to select an audio file for playback by entering its number. If the user enters "0" again, the function exits the loop and clears the screen. If the user chooses a valid file number, the function plays the selected audio file using pygame.mixer.music. It loads the file with pygame.mixer.music.load() and starts playback with pygame.mixer.music.play(). To stop playback, the function checks if the "esc" key is pressed using keyboard.is_pressed(). If the key is pressed, the function stops audio playback with pygame.mixer.music.stop().

In case of errors, the function provides feedback on incorrect actions during the date and file selection process. The user is prompted to retry if an incorrect value is entered. In the future, this function can be enhanced in various ways to improve its functionality, usability, and performance.

The data analysis and visualization module is implemented by the create_charts function. It generates two interactive charts using Plotly based on data stored in an Excel file. The Excel file contains three sheets: "Sheet1" containing words (one per row) and the time they were spoken, "Sheet2" for unwanted words (Z-words), and "Sheet3" for general text data that is not segmented into individual words.

The function begins by setting paths to the Excel file, which is named {username}_text.xlsx, and reads the three sheets into three separate Pandas DataFrames: data_word_time, data_overall_text, and data_unwanted_words. The data_word_time DataFrame is grouped by words and computes the word count for each word, storing the result in a new DataFrame named word_counts.

A comprehensive intelligent language analysis and control system that processes and analyzes user speech is based on criteria specified by the user. It identifies unwanted words and provides various visualization methods to help users better understand their language issues. The application utilizes Python libraries such as PyAudio, Vosk, Pandas, and Plotly for audio recording and processing, speech recognition, data management, and visualization creation. Additionally, detailed descriptions of various functions handling different aspects of the program are provided, including speech recording, speech processing, managing unwanted words in Excel, audio playback, and creating diagrams. Furthermore, alternative implementation methods for some of these functions, potential future improvements, and additional visualization and analysis methods that can be applied to provide users with more insights into their data are explored. These include Word Clouds, time series decomposition, sentiment analysis, topic modeling, n-gram analysis, part-of-speech tagging, and lexical diversity [19].

By leveraging these additional features and enhancements, the program can become an even more powerful tool for users who want to analyze their speech, identify patterns and trends, and work on improving their communication skills. The combination of speech recognition, data analysis, and visualization makes this program a versatile solution for users seeking deeper insights into their speech habits and progress . The final section provides a comprehensive overview and demonstration of the program's full functionality that users will encounter when using it.

5. Conclusions

An intelligent speech analysis and monitoring system has been developed to aid in identifying deficiencies and inaccuracies in the use of the Ukrainian language. The system considers individual user requirements, adapting its approach to analysis and monitoring for different contexts and situations. An analysis of the current state of the Ukrainian language revealed problematic aspects related to insufficient language proficiency, regional linguistic differences, and the influence of other languages on Ukrainian. These findings highlight the need for greater support and promotion of the Ukrainian language.

The application of systems analysis and UML methods helped identify key problem areas and develop conceptual models for speech analysis and monitoring. Diagrams (deployment, activity, state, classes, and others) were created, using the MAI method to investigate the type of intelligent system and determine priority actions for improving speech quality and proposing language monitoring methods.

The developed system allows adapting the process of speech analysis and monitoring to various user-established criteria, contributing to an increased overall level of proficiency in the Ukrainian language. Specifically, the system can be used to check speech compliance with language standards, detect and correct grammar and vocabulary errors subsequently. The research results may serve as a basis for the development of new methods of speech analysis and monitoring that consider individual user needs. Methodologies for identifying and correcting common speech errors, tools for automatic text analysis, and algorithms for assessing proficiency in the Ukrainian language can be developed.

Future research efforts could focus on enhancing the developed speech analysis and monitoring system, including expanding its functionality, integrating machine learning algorithms to improve analytical capabilities, and integrating with other linguistic tools and services. Moreover, the system could be adapted for use in various fields where a high level of Ukrainian language proficiency is required, such as education, science, culture, quality assurance of translations, and communication technologies.

Detailed discussions of various functions responsible for performing different program tasks, such as speech recording, speech processing, managing unacceptable words using Excel, listening to audio files, and creating diagrams, have been considered. Alternative approaches to implementing some of these functions, potential improvements, and additional visualization and analysis methods that can help users better understand their data have been discussed.

Thanks to these additional capabilities and improvements, the program can become an even more powerful tool for users seeking to analyze their speech, identify patterns and trends, and develop their communication skills. The application of speech recognition, data analysis, and visualization techniques makes this program a universal solution for users looking to delve deeper into their speech habits and track their progress in developing language skills. Therefore, during the execution of the master's qualification work, an intelligent speech analysis and monitoring system was developed that meets the stated goal and helps address the set tasks. The research results have scientific novelty and can be used for further improvement of analytical and control methods in the field of the Ukrainian language.

References

[1] Languages of Ukraine. URL: Languages of Ukraine - Wikipedia.

[2] Language data for Uraine. URL: Language data for Ukraine - Translators without Borders.

[3] Li, X., and Wu, X. A survey of speech-to-text techniques. International Journal of Speech Technology, 20(2), (2017). 303–313.

[4] N. Sethiya, Ch. K. Maurya, End-to-end speech-to-text translation: a survey, 2023. URL: https://arxiv.org/html/2312.01053v1.

[5] M. Malik, Automatic speech recognition: a survey / Maltimedia tools and applications, 80 (3):1-47, 2021. doi: 10.1007/s11042-020-10073-7.

[6] H. Kheddar, M. Hemis, Y. Himeur, Automatic speech recognition using advanced deep learning approaches: a survey / Information Fusion, V. 109, 2024. doi: https://doi.org/10.1016/j.inffus.2024.102422.

[7] C. Deuerlein, M. Langer, J. Sebner, Human-robot-interaction using cloud-based speech recognition systems / Procedia CIRP, V. 97, 2021, p. 130-135. doi: 10.1016/j.procir.2020.05.214.

[8] P. Daniels, K. Iwago, The suitability of cloudbased speech recognition engines for language learning, 2017. doi: 10.29140/jaltcall.v13n3.220.

[9] A. Jalashgar, Goal oriented systems modelling: justification of the approach and overview ofthe metods / Reliability Engineering and System Safity, V. 64, 1999, p. 271-278. doi: 10.1016/S0951-8320(98)00067-2.

[10] R. de F.S.M. Russo, R. Camanho, Criteria in AHP: a systematic review of literature / ProcediaComputer Science, V. 55, 2015. doi. 10.1016/j/procs.2015.07.081.

[11] UML Diagram Types | Learn About All 14 Types of UML Diagrams. Creately Blog. URL: https://creately.com/blog/diagrams/uml-diagram-types-examples/.

[12] What is Unified Modeling Language (UML)?. Ideal Modeling & Diagramming Tool for Agile Team Collaboration. URL: https://www.visual-paradigm.com/guide/uml-unified- modeling-language/what-is-uml/ (date of access: 26.11.2022).

[13] S. Ahmed, A. Ahmed, N. U. Eisty, Automatic Transformation of Natural to Unified Modeling Language:A Systematic Review,2022.URL:scholarworks.boisestate.edu/ cgi/viewcontent.cgi?article=1356&context=cs_facpubs.

[14] A. Rista, A. Kadriu, Automatic speech recognition: A comprehensive survey / SEEU Review, V. 15, Issue 2. doi: 10.2478/seeur-2020-0019.

[15] S.W. Chin, K.P. Seng, L.-M. Ang, Audio-visual speech processing for human computer interaction. doi: 10.1007/978-3-642-23363-0_6.

[16] J. Wu, English Real-time Speech Recognition Based on Hidden Markov and Edge ComputingModel, 2021. doi: 10.1109/ICIRCA51532.2021.9544571.

[17] S. Giriajan, A. Pandian, Real-Time Speech Enhancement Based on Convolutional Recurrent Neural Network, 2023. doi: 10.32604/iasc.2023.028090.

[18] A. Stergiou, D. Damen, Play it back: Iterative Attention For Audio Recognition, 2023. URL: http://2023.ieeeicassp.org.

[19] K. Padmanandam, A speech recognized dynamic word cloud visualization for text summarization, 2021. doi: 10.1109/ICICT50816.2021.9358693.