VAHCINET: Voice Activated Human Computer Interaction using Convolutional Neural Networks Algorithm

Regina Coellie U. Escobar
Computer Engineering Department
Colegio de Muntinlupa
Muntinlupa City, Philippines
escobar.reginacoellie.u.109@cdm.edu.com

Ivan Webster J. Razo
Computer Engineering Department
Colegio de Muntinlupa
Muntinlupa City, Philippines
razo.ivanwebster.j.102@cdm.edu.com

Francis Ivan D. Tabusao
Computer Engineering Department
Colegio de Muntinlupa
Muntinlupa City, Philippines
tabusao.francisivan.d.641@cdm.edu.com

***Abstract*— Motor impairments has a diverse condition affecting movements and coordination that presented difficulties to perform everyday tasks. Over the years, technologies had been developed to reduce the challenges faced by motor impaired persons (MIPs), particularly Assistive technologies (AT). These ATs such as automatic speech recognition (ASR) systems were used to perform computer tasks but was inconsistent due to noise problems. Then, multiple active noise cancelling techniques were implemented to address these noise problems from digital signal processing to the use of deep learning (DL) algorithms. The noise problems were minimized but, despite those intensive researches for noise cancellation, ASR systems lacks Human-Computer Interaction (HCI) systems for computer navigation. As such, with previous studies lacking for HCI-based voice commands for computer navigation. The researchers aimed to develop a novel system with custom datasets by classifying 34 HCI-based voice commands to perform computer navigations for MIPs that utilized convolutional neural network (CNN) algorithm along with a feature for manual computer navigation through eye sight and eye blinking. After the prototype testing, the one of the major findings of the study was 17653 custom datasets constructed for the CNN architecture. Then, the best trained CNN architecture produced an accuracy of 99.3%, 0.07 loss with a performance metrics of 0.98 for all F1-score value, precision value, and recall value. However, VAHCINET had a mean average percentage error of 29.08% during prototype testing which resulted its actual accuracy at 70.92% without speaker characteristic factors. Lastly, the performance of eye tracker was evaluated and reported that 11-13 seconds were needed to perform each manual computer navigation while the highest time for one task was 13 seconds.**

Keywords—assistive technologies, automatic speech recognition, convolutional neural network, human-computer interaction, motor impaired persons

I. Introduction

Motor impairments encompass a diverse range of conditions affecting movements and coordination. These detrimental effects were common among motor-impaired persons (MIP) when performing everyday tasks, such as difficulty using both hands simultaneously [1]. Government Programs such as the Magna Carta for Persons with Disabilities in the Philippines guarantee individuals with disabilities access to quality education, including necessary support systems and assistive technology (AT) [2].

With the recent developments and the rising demand for technologies nowadays, the educational system for learning opted in using computers has become relevant. A conducted a study in Brazil that investigated the impact of motor disabilities on computer use and identified the main challenges faced by 100 MIPs [3]. 80% of respondents reported difficulty using standard input devices. 60% of respondents reported feeling frustrated when using computers. 40% of respondents reported avoiding using computers altogether due to accessibility barriers. The findings of this study highlighted the need for improved accessibility of computer technology for MIPs.

Over the years, technologies, ATs became more accessible for MIPs who independently handled a more comprehensive range of activities. These ATs, such as automatic speech recognition (ASR) systems, assisted MIPs in reading and writing documents, communicating with others, and searching for information online. However, ASR systems had difficulties recognizing the user's voice inputs due to noises captured by the system leading to the need for noise-canceling features.

Multiple active noise canceling techniques were implemented such as spectral subtraction and adaptive filters from least mean square (LMS), Jaya, and particle swarm optimization (PSO) that proved more effective than traditional noise canceling techniques [4][5]. Aside from that, deep learning (DL) algorithms were also integrated into ASR systems, such as deep neural network-based noise suppression algorithm to systems that used convolutional neural network (CNN) and recurrent neural network (RNN) to find the most suitable audio classification model for noises [6][7]. Despite intensive research dedicated to noise cancellation, it lacks Human-Computer Interaction (HCI) systems for computer navigation.

HCI is an emerging technology with many algorithms, approaches, and techniques to enhance interaction [8]. A standard computer requires peripherals, such as a mouse and keyboard, that can be controlled using muscle movements but are primarily unsuitable for MIPs [9]. As such, numerous HCI-based systems were developed to address the gaps between PWDs and computers. One of these studies was conducted to explore DL algorithms for gesture with voice command systems for computer navigation [10]. Then, HCI systems, particularly voice commands, bring about studies related to using the CNN model for speech recognition and natural language processing with ten possible voice commands for computer navigation [11]. In a recent study using the CNN model, 12 voice commands were used not only for computer navigation but also to find the effect of dataset size on the system's accuracy [12].

The multiple CNN architectures of previous studies had been trained and tested to determine which can yield the highest accuracy. To further understand the differences made by each CNN architecture, it was presented by Table I.

TABLE I. STATE OF THE ART OF CNN ALGORITHMS

Architecture	Number of Datasets	Number of Classes	Accuracy (%)
CNN-DICVA1	18000	10	81
CNN-DICVA2	18000	10	89
CNN-DICVGG	18000	10	89
CNN-Native 50	600	12	64.81
CNN-Native 100	1200	12	77.23
CNN-Native 250	3000	12	82.12
CNN-Native 500	6000	12	89.64
CNN-Native 1000	12000	12	92.58
CNN-Native 2500	28136	12	92.11
CNN-Native Max	41682	12	94.64

The difference in the number of datasets significantly contributed to the overall accuracy between each architecture since DL algorithms required larger datasets to increase the accuracy of the CNN architecture, mainly since the datasets were recorded voice inputs that consisted of several continuous signal data. Consequently, the CNN-DICV architectures had different configurations of CNN layers, while all the CNN-Native architectures had the exact configuration of CNN layers. This also showed that the CNN architectures used were based on a trial-and-error approach to achieve the highest possible accuracy.

As such, according to previously mentioned studies, there is a need for more HCI-based voice commands, particularly for computer navigation. This led to this paper, wherein the researchers aimed to develop a novel system with custom datasets by classifying HCI-based voice commands to perform computer navigation for MIPs that utilized the CNN algorithm along with a feature for manual computer navigation through eyesight and eye blinking.

For the next section, the paper introduced the methodology of this study.

II. METHODOLOGY

A. Audio Pre-Processing Techniques

Frequency masking was applied to the datasets (voice input) to eliminate louder sounds and generate a clearer dataset. Then, each dataset was further processed through downsampling as decreased the amount of data within a signal. Lastly, stereo-to-mono channel conversion was done not only due to signal requirements in the DL platform but also for data augmentation, as it only accepts mono channels.

B. Data Augmentation

VAHCINET was integrated with the highest accuracy of a novel CNN architecture with an original number of custom datasets of 3,078 waveform (WAV) audio file format. Then, through data augmentation of audio pitch, audio time shift, audio speed and noise injection using Google Colab, the total number of datasets was increased to 15,622 total augmented datasets plus 2031 noise WAV files from ARCA23K dataset [13]. Together, the total custom dataset reached 17,653 WAV files as shown by Fig. 1.

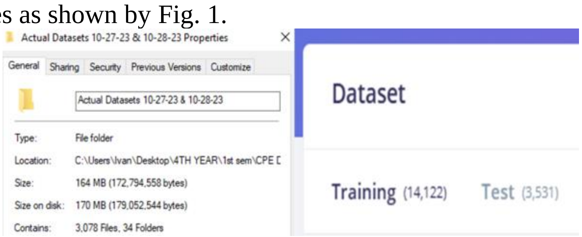

Fig. 1 Comparison of the Original Dataset and Augmented Dataset of VAHCINET

C. Feature Extraction

After data augmentation and obtaining the sufficient number of datasets for VAHCINET, each WAV files of respective voice commands undergone feature extraction as shown by Fig. 2 [14].

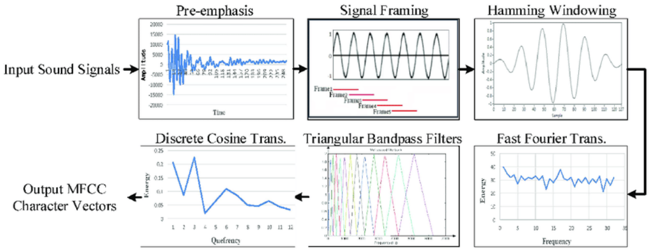

Fig. 2 Feature Extraction to MFCC

Input Sound Signals

Voice inputs were captured using devices such as microphone and was mapped to its respective amplitude vs time graph in the time domain.

Pre-emphasis

Then, pre-emphasis, as the first step of the MFCC adaptation process was used to balance the frequency spectrum, as higher frequencies usually had smaller magnitudes and avoided numerical problems during the Fourier transform (FT). The pre-emphasis filter can be applied using the first-order filter, as expressed in Fig. 3 [15].

$$y(t) = x(t) - \alpha x(t-1) \quad (1)$$

Fig. 3 First Order Filter Formula

Where: y(t) = first order filter
x(t) = voice input signal
α = filter coefficient
t = time

Signal Framing and Hamming Windowing

The signal framing broke down the process signal into shorter frames typically 20-30 milliseconds (ms) long. However, vowel sound has frames of 40-80 ms. Then, FTs was done on 20 millisecond frames, with 10 ms overlaps between frames for good approximation of the frequency contours [15].

After frames were obtained, window functions including Hanning and Hamming windows were applied to take finite sets of data from the processed signal and minimize spectral leakage during Fourier analysis [15].

Fast Fourier Transform

A power spectrum shows how the power of a signal is distributed across different frequencies. To determine the power spectrum of each signal frame, we use the following equation, as shown in Fig. 4 [14]:

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{-\frac{2\pi j n K}{N}} \quad k = 1, 2, 3 \ldots N-1 \quad (2)$$

Fig. 4 Power Spectrum Formula

Where: X(k) = Power spectrum

x(n) = 1-th frame of signal
N = N-point on each frame
t = time

Then, the processed signal of each signal frame was converted to an energy vs. frequency graph using the Short-Time Fourier transform (STFT).

Triangular Bandpass Filters

The Mel filter bank applied triangular filters on the Mel scale to the power spectrum and extracted the features from the processed signals after the Fourier transform that is non-linearly related to frequency, similar to how humans perceive speech sounds. The conversion between Hertz (f) and Mel (m) is shown in Fig. 5:

$$m = 2595 \log_{10}\left(1 + \frac{f}{700}\right)$$
$$f = 700\left(10^{\frac{m}{2595}} - 1\right) \quad (3)$$

Fig. 5 Hertz and Mel-scale Conversion [15]

Where: m = Mel-scake
f = Hertz

Discrete Cosine Transforms (DCT)

Discrete Cosine Transform (DCT) is applied to the outputs of a Mel scale filter bank to select the most informative coefficients or to decorrelate the log mel spectral energies. The DCT is calculated using the following equation as shown by Fig. 6 [14]:

$$X\,(k) = \sum_{n=0}^{N-1} x_n * COS\left(2\pi jnk/_N\right), \quad k = 1, 2, 3 \ldots N-1 \quad (4)$$

Fig. 6 Discrete Cosine Transforms Formula

Where: X (k) = DCT
N = Mel-frequency cepstrum coefficients

The DCT helps reduce correlation between adjacent filters in the filter bank. It allows to retain most of the important auditory information with just the first few coefficients.

Output MFCC Character Vectors

Following digital signal processing (DSP) and feature extraction, spectrogram values were generated and input to VAHCINET's CNN architectures for training and testing.

D. Convolutional Neural Network

The paper implemented 4 versions of CNN Architecture to identify the optimal configuration that yielded the highest accuracy as shown by Fig. 5.

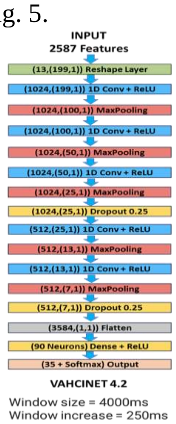

Fig. 5 VAHCINET 4.2 CNN Architecture

VAHCINET version 4.2 had a window size of 4000ms and window increase of 250ms, with 1937 and 2587 input features. It utilized various 1D convolutional layers with ReLU activation, MaxPooling layers, and flatten elements. The flatten output was extracted by dense layers of 90 neurons. Across architectures, the output contained 35 voice command and noise classifications with Softmax activation.

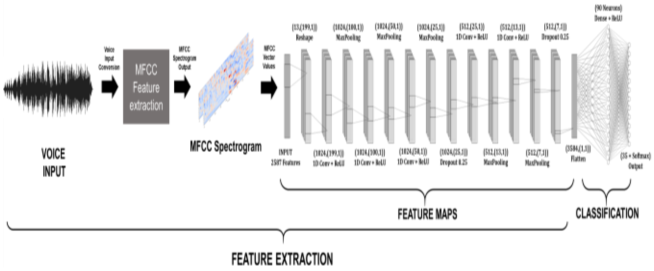

Fig. 6 VAHCINET 4.2 Architecture Schematic

VAHCINET version 4.2 has a two-part architecture: feature extraction and classification. Feature extraction uses MFCC to obtain values from time-series datasets. After flattening, these values pass through a multi-class neural network, resulting in 35 classifications.

E. System Overview of VAHCINET

In this section, it proposes the project design and system overview of the VAHCINET as shown by Fig. 7 and functions through the following operations:

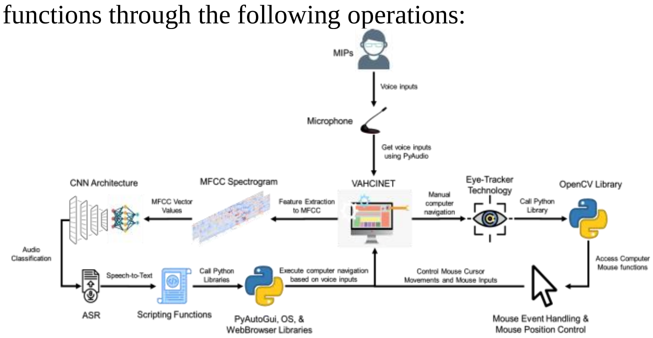

Fig. 7 System Overview of the VAHCINET

The operations of VAHCINET have two modes of computer navigation: HCI-based voice commands and mouse event handling and position control using eye-tracker technology as discussed below.

VAHCINET Operation: HCI-based Voice Command

Step 1 – MIPs: Voice inputs from MIPs are captured.

Step 2 – Microphone: Captured voice inputs are transmitted to VAHCINET using PyAudio.

Step 3 – VAHCINET: The system prepares the voice input for feature extraction.

Step 4 – MFCC Spectrogram: After feature extraction, the spectrogram will be generated wherein MFCC vector values can acquired in an array data type.

Step 5 – CNN Architecture: Using the MFCC vector values, the novel CNN architecture with the highest accuracy will perform audio classification among the 34 HCI-based voice commands.

Step 6 – ASR: The classified HCI-based voice command will be converted from speech to text using ASR system of VAHCINET.

Step 7 – Scripting Functions: The text from the ASR will undergo numerous conditional statements that contains the syntaxes of python libraries for computer navigation.

Step 8 – PyAutoGui, OS & WebBrowser Libraries: Each HCI-based voice commands have different line of codes as PyAutoGui and OS library for desktop computer navigations while WebBrowser library for browser computer navigations.

VAHCINET Operation: Mouse Event Handling and Position Control

Step 1 – Eye-Tracker Technology: Choosing manual computer navigation from the system activates the Eye tracker of VAHCINET using OpenCV.

Step 2 – OpenCV Library: OpenCV libraries accesses the mouse functions and supports the following functions based on Table II.

TABLE II. EYE AND HEAD TRACKER ACTION AND FUNCTION TABLE

Eye and Head Action		Function
Left Eye	Right Eye	
		Right Mouse Button Click
		Left Mouse Button Click
		No function and Print "Both Eyes Blinked"
		Mouse Cursor Movements

Step 3 – Mouse Event Handling and Mouse Position Control: Using the following functions, mouse cursor movements and mouse inputs can be controlled manually to navigate the computer and use the available functions of VAHCINET:

VAHCINET Functional Buttons

1. Search Button: Imitates the Windows Search button.

2. Upload Button: Opens the Default Browser and will navigate to Google Drive.

3. LMS Button: Opens up the CDM LMS Blackboard using Default Browser.

4. Files Button: Opens the File Manager and automatically clicks the search panel.

5. Browser Button: Opens the Default Browser and will navigate to Google.

6. Eye Tracker Mouse Button: Activates the ET Mouse feature that allows manual control of the mouse cursor using users eye and head movements.

7. Speak Button: Activates the ASR system that allows computer navigation using voice inputs.

8. Macro Button: Opens a popup that contains the list of available voice commands that the application can perform.

9. MS Office Button: Opens a pop-up menu of the Microsoft Word, Excel, and Powerpoint application.

10. Music Button: Opens the default media player of the computer.

F. Data Validation

The study implemented confusion matrix which assessed the performance of all CNN architectures used during the training and testing phase using True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN) parameters.

After determining that parameters, Precision can be calculated to know the correctly predicted classes that are positives predictions denoted by [16]:

$$Precision = \frac{TP}{TP + FP} \quad (5)$$

Fig.8 Confusion Matrix Formula for Precision

Then, Recall can be calculated to know the correctly predicted classes out of all the predictions in the confusion matrix denoted by [16]:

$$Recall = \frac{TP}{TP + FN} \quad (6)$$

Fig.9 Confusion Matrix Formula for Recall

Lastly, accuracy was calculated using previous parameters to determine the CNN architecture accuracy [16]:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \quad (7)$$

Fig.10 Confusion Matrix Formula for Accuracy

In contrast, to determine the number of the percentage error of each HCI-based voice commands, within the 10 trials, correct execution of the voice commands was counted for the rest of the 45 respondents.

$$\text{Percentage Error} = \frac{\left|\text{Actual number of correct Voice command executed} - \text{Expected correct number of Voice command executed}\right|}{\text{Expected correct number of Voice command executed}} \times 100$$

$$\text{Mean Average Percentage Error} = \frac{\text{Sum of All Percentage Errors}}{\text{Total number of HCI-based Voice Commands}} \quad (8)$$

Fig. 11 Mean Average Percentage Error Computation

The average number of correct voice commands executed per respondent was calculated. The percentage error was determined using the actual and expected number of correct commands. Finally, the MAPE was found by summing all percentage errors and dividing by 34 voice commands.

Separately, MAPE was used to determine VAHCINET's accuracy regarding speaker characteristics. This included testing with voice changer software, as well as variations in gender, loudness, speech rate, and pitch.

III. RESULTS AND DISCUSSIONS

A. Construction of the Custom Dataset

The experimental results from all the proposed architectures were included with all the necessary metrics.

After the data augmentations were executed, it generated a total 17653 datasets and was distributed as shown by Fig. 12.

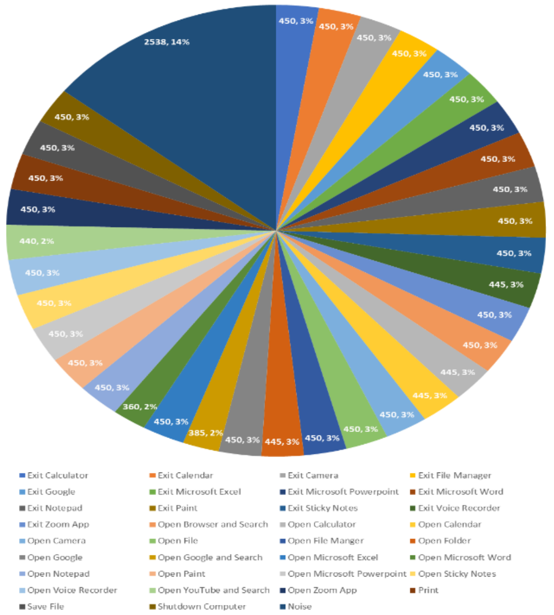

Fig. 12 Distribution of Recorded WAV Files Among 35 Classes

The graph showed that class balancing for each voice command differed by 2-3%, except for the noise class. This percentage range allowed the CNN architecture to avoid biases among voice commands during training and testing. However, the number of WAV files was not equal among some of the 35 HCI-based voice commands, so the CNN architecture could still be prone to improper audio classifications.

B. Performance Metrics of CNN Architectures

After the training phase of all CNN architectures from 40 to 80 epochs, each accuracy, F1-score, precision, and recall value was collected, as presented by Fig. 13.

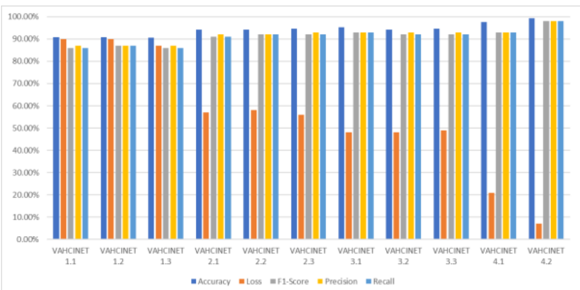

Fig. 13 Average Performance Metrics of All Novel CNN Architecture

Based on the figure, VAHCINET 4.2 achieved the highest accuracy of over 99.3% with the lowest loss of 0.07. Its F1-score, precision and recall were 0.98, demonstrating the best performance for accurate audio classification among 35 classes. To understand how VAHCINET 4.2 achieved this top metric performance, Figure 14 shows its confusion matrix.

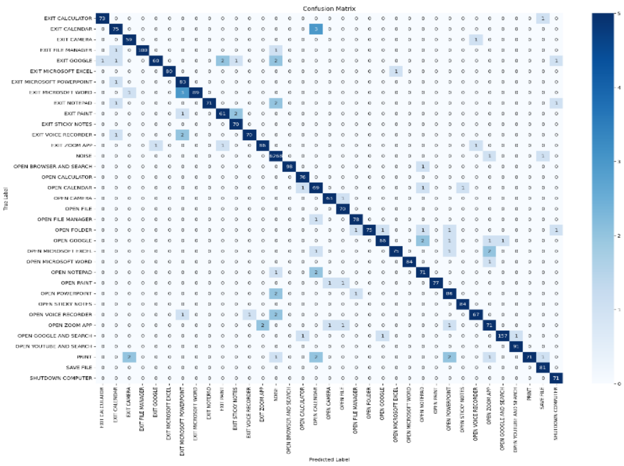

Fig. 14 Confusion Matrix of VAHCINET 4.2 CNN Architecture

The confusion matrix color legend ranges from 0-5, indicating the number of true positives, true negatives, false positives, and false negatives in VAHCINET 4.2's voice command confusion matrices. Table III presents the average counted TP, TN, FP, and FN to understand the confusion matrix results.

TABLE III. AVERAGE TP, FP, FN, TN PARAMETERS OF VAHCINET 4.2

Voice Commands	TP	FP	FN	TN
Exit Calculator	73	1	0	8938
Exit Calendar	75	3	0	8934
Exit Camera	59	1	0	8952
Exit File Manager	100	1	1	8910
Exit Google	60	6	2	8944
Exit Microsoft Excel	80	1	1	8930
Exit Microsoft Powerpoint	83	0	4	8925
Exit Microsoft Word	89	0	1	8922
Exit Notepad	71	3	1	8937
Exit Paint	61	2	0	8949
Exit Sticky Notes	70	0	3	8939
Exit Voice Recorder	70	0	2	8940
Exit Zoom App	86	1	0	8925
Noise	6268	2	0	2742
Open Browser and Search	98	1	0	8913
Open Calculator	76	0	0	8936
Open Calendar	69	2	1	8940
Open Camera	63	1	0	8948
Open File	70	0	0	8942
Open File Manager	78	0	1	8933
Open Folder	75	4	1	8932
Open Google	66	5	0	8941
Open Google and Search	75	3	1	8933
Open Microsoft Excel	84	1	0	8927
Open Microsoft Word	71	0	3	8938
Open Notepad	77	0	2	8933
Open Paint	86	0	3	8923
Open Microsoft Powerpoint	84	0	0	8928
Open Sticky Notes	67	0	4	8941
Open Voice Recorder	71	0	5	8936
Open YouTube and Search	157	1	2	8852
Open Zoom App	91	0	0	8921
Print	71	1	8	8932
Save File	81	0	0	8931
Shutdown Computer	71	0	0	8941
Average TP			255.03	
Average FP			1.14	
Average FN			1.31	
Average TN			8754.51	

For the TP, VAHCINET 4.2 had an average of 255.03 correctly classified voice commands when comparing the predicted label to the true label in the confusion matrix, implying its architecture was more effective for audio classification of 35 classes. The FP and FN averages of 1.14 and 1.31 falsely classified commands, lower than the TP, implying VAHCINET 4.2 can correctly classify audio rather than falsely, resulting in higher accuracy. Lastly, the TN average of 8754.51 indicates the sum of TP, FP, and FN for

34 classes excluding one, implying proficiency in distinguishing classes and not incorrectly assigning labels.

C. Prototype Testing of VAHCINET HCI System

The prototype testing of the HCI system was performed with the respondents to determine the percentage errors of each voice command using MAPE. Each command was tested over 10 trials, and the average correct executions were used to calculate percentage errors. The overall MAPE was 29.08%, implying an accuracy of 70.92% for the system when integrated with the best CNN architecture. While the model achieved 99.30% in testing, the actual accuracy declined to 70.92% during computer navigation scenarios. Aside from that, tests using a voice changer yielded errors from 74.12% to 98.82%, indicating performance issues for speaker characteristics factors.

D. Prototype Testing of VAHCINET Eye Tracker System

The eye tracker prototype testing was done after the HCI system testing. However, only 20 of 45 respondents agreed to participate due to personal reasons. Some tasks recorded zero times during prototype testing due to eye tracking functionality limitations. Many tasks required holding the left mouse button and scrolling, implying the eye tracker can only perform some navigation as it currently lacks more advanced functions. Completed tasks took 11-13 seconds using current eye tracker functions, with "Open any application" taking longer, as respondents had difficulty adapting to eye tracking for navigation during testing.

IV. CONCLUSION AND RECOMMENDATION

VAHCINET is an HCI system that uses voice commands for computer navigation by persons with disabilities, specifically motor-impaired persons (MIPs). Researchers acquired, preprocessed, and augmented WAV files to increase custom datasets to 17,653 for training and testing novel CNN architectures to improve audio classification performance metrics. The CNN architecture VAHCINET 4.2 proved best for audio classification due to training and testing performance metrics. However, prototype testing showed decreased performance for unseen data, since custom datasets excluded MIP respondents. New data variations like speaker characteristics inconsistently affected audio classification, requiring more consideration during data acquisition.

While the eye-tracker offered potential for manual navigation, it remained limited to current functions, implying the need for optimization and expanded functions. Overall VAHCINET improved on prior HCI systems but requires further refinement, beginning with data acquisition accounting for voice variations to enable correct audio classification under different voices. Prototype testing in an open area introduced inconsistencies from environmental noises and other voices. Using noise-canceling earphones/headsets instead of laptop microphones could address this. Researchers recommend exploring capabilities beyond navigation like dictation and text editing. Improving existing commands and adding functionality like file/program access from any storage location were also proposed. A tutorial and startup procedure were proposed for users. Future systems should employ adaptive DL algorithms enabling adding custom commands/words on the fly for flexibility and ease of access. Researchers recommend expanding the eye-tracker's role and accounting for conditions affecting pupil detection and interface interaction. Further refinement is needed to optimize VAHCINET for individuals with motor impairments.

ACKNOWLEDGMENT

Foremost, the researchers would like to express sincere gratitude to their research adviser, Engr. Cyd Laurence B. Santos, research professor, Engr. Ricrey E. Marquez and on behalf of the Government of Barangay Bayanan, City of Muntinlupa for the continuous support and guidance throughout the research study.

REFERENCES

[1] "Fine Motor Difficulties," *SBCC*, Sep. 07, 2023. https://www.sbcc.sg/fine-motor-difficulties/

[2] Official Gazette, "Republic Act No. 7277 | GOVPH," *Official Gazette of the Republic of the Philippines*, Mar. 24, 1992. https://www.officialgazette.gov.ph/1992/03/24/republic-act-no-7277/

[3] Oliveira, M. E. C., et al., "The Impact of Motor Disabilities on Computer Use: A Survey of Users," *Disability and Rehabilitation*, 2020.

[4] J. Lin *et al.*, "Speech Enhancement Using Forked Generative Adversarial Networks with Spectral Subtraction," *Proceedings of Interspeech 2019*, Sep. 2019, doi: https://doi.org/10.21437/Interspeech.2019-2954.

[5] M. Venkata Sudhakar, M. Prabhu Charan, G. Naga Pranai, L. Harika, and P. Yamini, "Audio signal noise cancellation with adaptive filter techniques," *Materials Today: Proceedings*, vol. 80, pp. 2214–7853, Jul. 2021, doi: https://doi.org/10.1016/j.matpr.2021.07.080.

[6] Zhang, X.-L, Yang, K.-F., and Xu, Q.-L, "Deep Neural Network-based Noise Suppression for Robust Speech Recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 858–869, 2021, doi: https://doi.org/10.1109/TASLP.2021.3064949.

[7] M. S. Imran, A. F. Rahman, S. Tanvir, H. H. Kadir, J. Iqbal, and M. Mostakim, "An Analysis of Audio Classification Techniques using Deep Learning Architectures," *2021 6th International Conference on Inventive Computation Technologies (ICICT)*, pp. 1–6, Jan. 2021, doi: https://doi.org/10.1109/icict50816.2021.9358774.

[8] P. Chakraborty, D. Roy, Z. Zahid, and S. Rahman, "Eye Gaze Controlled Virtual Keyboard," *International Journal of Recent Technology and Engineering (IJRTE)*, vol. 8, no. 4, pp. 3264–3269, Oct. 2019, doi: https://doi.org/10.35940/ijrte.D8049.118419.

[9] B. Šumak, M. Špindler, M. Debeljak, M. Heričko, and M. Pušnik, "An empirical evaluation of a hands-free computer interaction for users with motor disabilities," *Journal of Biomedical Informatics*, vol. 96, p. 103249, Aug. 2019, doi: https://doi.org/10.1016/j.jbi.2019.103249.

[10] T. Nithya Shree and N. Alagu Sundari, "A Virtual Assistor for Impaired People by using Gestures and Voice," in *2023 5th International Conference on Inventive Research in Computing Applications (ICIRCA)*, pp. 546–550. doi: https://doi.org/10.1109/ICIRCA57980.2023.10220605.

[11] H. A. Rusan and B. Mocanu, "HumanComputer Interaction Through Voice Commands Recognition," in *2022 International Symposium on Electronics and Telecommunications (ISETC)*, pp. 1–4. doi: https://doi.org/10.1109/ISETC56213.2022.10010253.

[12] A. N. Çayır and T. S. Navruz, "Effect of Dataset Size on Deep Learning in Voice Recognition," in *2021 3rd International Congress on HumanComputer Interaction, Optimization and Robotic Applications (HORA)*, pp. 1–5. doi: https://doi.org/10.1109/HORA52670.2021.9461395.

[13] T. Iqbal, Y. Cao, A. Bailey, M. D. Plumbley, and W. Wang, "ARCA23K," *Zenodo (CERN European Organization for Nuclear Research)*, Sep. 2021, doi: https://doi.org/10.5281/zenodo.5117901.

[14] W. Tsai, Y. Shih, and N. Huang, "HardwareAccelerated, ShortTerm Processing Voice and Nonvoice Sound Recognitions for Electric Equipment Control," *Electronics*, vol. 8, no. 9, 2019, doi: https://doi.org/10.3390/electronics8090924.

[15] Haytham Fayek, "Speech Processing for Machine Learning: Filter banks, Mel-Frequency Cepstral Coefficients (MFCCs) and What's In-Between," *Haytham Fayek*, Apr. 21, 2016. https://haythamfayek.com/2016/04/21/speech-processing-for-machine-learning.html

[16] A. Bhandari, "Confusion matrix for machine learning," *Analytics Vidhya*, Apr. 17, 2020. https://www.analyticsvidhya.com/blog/2020/04/confusion-matrix-machine-learning/