International Journal on Advanced Science Engineering Information Technology

Vol.14 (2024) No. 1
ISSN: 2088-5334

Design and Implementation of Mobile Application for CNN-Based EEG Identification of Autism Spectrum Disorder

Melinda Melinda [a,*], Fitri Arnia [a], Al Yafi [a], Nur Afny Catur Andryani [b], I Ketut A. Enriko [c]

[a] *Department of Electrical and Computer Engineering, Engineering, Syiah Kuala University, Banda Aceh, Indonesia*
[b] *Computer Science Program, School of Computer Science, Bina Nusantara University, Jakarta, Indonesia*
[c] *Department of Telecommunication Engineering, Faculty, Institut Teknologi Telkom Purwokerto, Purwokerto, Indonesia*
*Corresponding author: *melinda@usk.ac.id*

Abstract*—Autism spectrum disorder (ASD) is a disorder of the nervous system from birth and during infancy. This disorder affects children's development, making it difficult for nerve function to develop, and causes the child concerned to have difficulty in fostering social relationships. Early detection of children with ASD is needed so that treatment is fast and on target. Currently, facilities and research on early diagnosis of ASD patients through EEG signals are still very few, requiring much cost and more effort to analyze EEG signals in examinations related to ASD detection cases. This study proposes a mobile phone application that can distinguish people living with ASD and normal data signals based on asynchronous EEG brain signals. This research also produces a preprocessing algorithm and BCI2000 EEG data signal so that it can be automated using Python. This research also produces an output model, namely the Deep Learning Convolutional Neural Network, which is deployed using Python-Flask so that the diagnosis of EEG signals with ASD and normal patients can be used on various platforms through restAPI. This research is also expected to help the community and support the diagnosis of ASD sufferers so that they can be handled appropriately. Data for ASD sufferers and normal data were correctly classified into the appropriate class. Handling this disease requires close and integrated cooperation, so this ASD classification will be very helpful for patients and can make a diagnosis in a faster time, enabling patients to receive targeted treatment and therapy.

***Keywords*— BCI2000; signal to image; OpenCV; convolutional neural network; mobile apps.**

Manuscript received 6 Oct. 2023; revised 8 Nov. 2023; accepted 17 Dec. 2023. Date of publication 29 Feb. 2024.

I. INTRODUCTION

Autism spectrum disorders (ASD) are disorders of the nervous system that exist from birth to childhood. This disorder interferes with the child's development, making it difficult for the child to develop nerve function and difficult for the affected child to form social relationships [1], [2]. ASD reflects a decrease in brain function, this is because autistic children have a stronger spectrum at higher frequencies than normal children. The difference between the two groups was 2 points on the crown and 2 points on the occipital region [3]. Autistic people have communication and behavior problems, such as eye contact and lack of facial expressions. The effects of autism on the brain affect the control of motor functions.

Diagnosis of ASD sufferers in children generally only depends on several parameters of physical examination, behavior, and examination of the child's growth and development. This examination includes the main complaints such as delays in language development, regression of language skills, delays in communication skills, abnormal behavior, physical form, behavior, and so on [4], [5]. This examination model will take a long time because it must pay attention to the child's growth and development and must require more effort to ensure that a child has ASD or not.

Currently, technology can check ASD patients through electroencephalography (EEG) signals, namely by diagnosing ASD patients by paying attention to signal patterns that have 4 to 32 channels, manually or automatically [6]. However, until now, there is still very little technology that can analyze EEG signals and diagnose ASD sufferers automatically, so it requires a lot of money and time and makes many children with ASD not realize this disease early and have a bad impact on their development.

This study proposes a mobile application that can distinguish ASD and normal patients based on asynchronous EEG brain signal data. This research also produces a preprocessing algorithm and BCI2000 EEG signal data using the BCI2K python library and turns the signal into an image using OpenCV so that it can be automated using Python. It also produces the output of a Deep Learning trained model using the convolutional neural network (CNN) method, which is deployed using Python-Flask so that the diagnosis of ASD and normal patients based on EEG signals can be used on various platforms through restAPI, but this research is only focused on using the platform. Android uses a multi-platform framework, namely Flutter. This research is also expected to help the community so as to accelerate the diagnosis of ASD sufferers so that they can be handled properly. Data for ASD sufferers and normal data were correctly classified into the appropriate class. Handling this disease requires close and integrated cooperation, so this ASD classification will be very helpful for patients and can make a diagnosis in a faster time, enabling patients to receive targeted treatment and therapy.

To summarize, the contributions of this paper are listed as follows:

- Development of a mobile application for diagnosing children with ASD based on EEG signals.
- Implementation of asynchronous methods for efficiency in using EEG data.
- Focus on the Android platform with Flutter for ease of use on Android devices.

II. Materials and Methods

A. Dataset

This study utilizes an EEG dataset obtained from King Abdulaziz University (KAU), which comprises recordings from 17 subjects, including 4 with normal brain activity and 13 with autism spectrum disorder (ASD). These EEG recordings typically span between 1 to 2 hours and are stored in a ".dat" format [7]. It is divided into normal and ASD signal folders, which can be visualized using Python and BCI2000 viewer. It is essential to clarify that the data in the ".dat" format represents preprocessed EEG data organized in a matrix-like structure rather than a raw waveform. This format is specifically chosen to facilitate data handling and analysis in the context of this research.

By using the ".dat" format, we refer to the structured representation of EEG signals as matrices rather than the original waveform. This choice enables compatibility with the tools and techniques employed in our analysis. Therefore, it's not a conversion of waveforms to ".dat" but rather an acknowledgment that the data is already stored in this matrix format, allowing for efficient processing and visualization. This format can be readily visualized and manipulated using Python and BCI2000 viewer, making it suitable for the analysis and experiments conducted in this study.

B. Preprocessing

By re-implementing existing computational techniques for signal processing, Utrecht computer scientists have managed to speed up the technique by one hundred, with no loss in quality. This enables significant improvements in countless applications that work with signals or data streams, from sensors, MRI scanners, and systems that predict earthquakes. Lukas Arts and Egon van den Broek published their results today in Nature Computational Science [8].

Dataset preprocessing is carried out in several stages: inputting EEG data, cleaning EEG data using DWT, windowing, EEG to Spectrogram, and splitting the dataset with a ratio of 80% training data and 20% test data. The overall process can be seen in Fig. 1, which can be explained as follows.

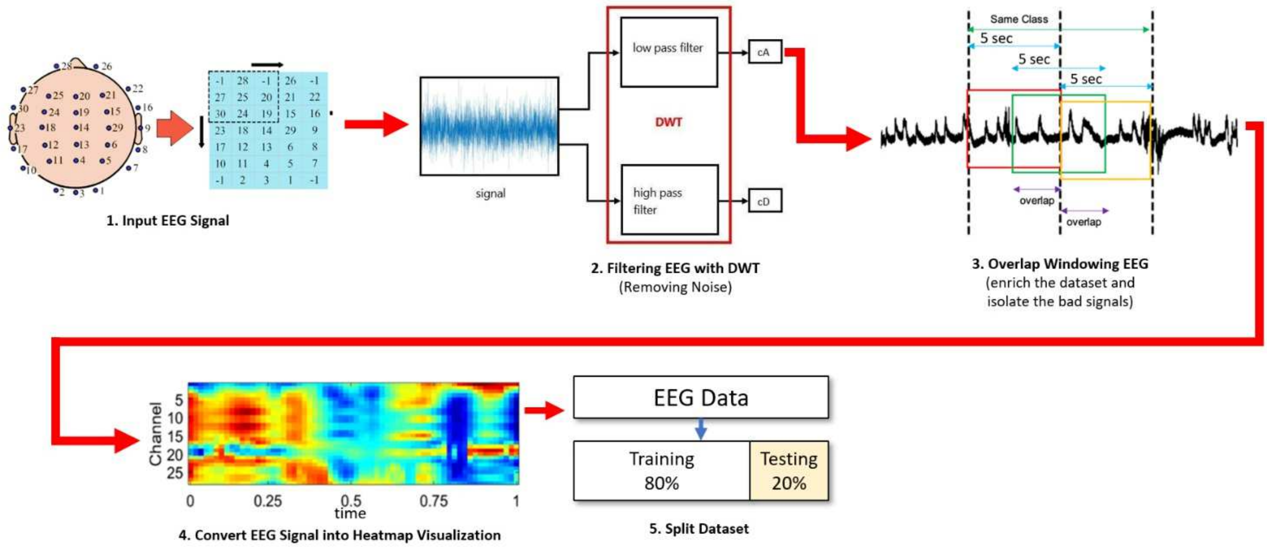


Fig. 1 Preprocessing processes

1) Input Data: The BCI2KReader library, which must be installed and imported into the program, is used to enter EEG data.

2) Discrete Wavelet Transform (DWT): In this process, we use DWT-2 with the PyWavelets module in Python for EEG data cleaning. Remarkably, this process reduces noise and artifacts from the EEG signal, creating frequency sub-fields ready for further analysis or classification [9]. The DWT2 approach with "haar" transformation and low-pass filtering (LPF) effectively cleans the EEG signal, producing purer and more relevant data.

3) Overlap Windowing: In this phase, a windowing technique known as "overlap windowing" is employed to segment the EEG signal into smaller, manageable slices. The process commences by establishing the initial and final boundaries of these slices, each set to the desired length as specified in the reference [10]. These initial and final slice boundaries are strategically determined to optimize the signal segmentation. Notably, the key feature of this windowing approach is that these slice boundaries are configured to overlap by 50%. This deliberate overlap is incorporated with the specific intention of incorporating information from previously windowed signals. Doing so aims to provide valuable context and continuity when developing the Convolutional Neural Network (CNN) model. This overlap facilitates the seamless transition between adjacent slices and enhances the CNN's ability to extract meaningful patterns and relationships from the EEG data.

4) EEG to Heatmap: EEG to heatmap is the process of converting EEG data into a visual representation in the form of a heat map, where different colors or intensities indicate the level of brain activity at different locations [10], [11]. This facilitates EEG data analysis that is more intuitive and easier to understand for researchers and health professionals [12]. The goal is to prepare data as input to a CNN model [13].

C. EEG Classification using Convolutional Neural Network (CNN)

CNN is a deep learning method commonly used for classification, be it figs, signals, writing, etc. CNN is not much different from other neural networks. CNN consists of neural weights, biases, and functions and has layers arranged so that they are shaped like filters [14]–[16].

CNN/ConvNet is part of a deep neural network, an artificial neural network widely used in the recognition and processing of fig. This algorithm was created primarily to handle pixel data and visual figures. The neurons of the CNN algorithm are intended to function similarly to the frontal lobes, especially regions of the visual cortex, in human and animal brains. The visual cortex is the area of the brain responsible for processing information in the form of visual input. This is what distinguishes CNN from other neural network techniques in image processing [17]. Examples include facial recognition, fig categorization, and other applications of CNN in computer vision. Like a simple neural network, CNN contains parameters that can be learned, like a neural network, namely weights, biases, etc [18].

EEG signal classification using CNN is carried out in three main stages, which play an important role in data processing and classification predictions, which can be seen in Fig. 2. The first stage is input, where EEG data representing brain signals at various times is converted into a heatmap image [19]. Subsequently, the heatmap image becomes input data for CNN.

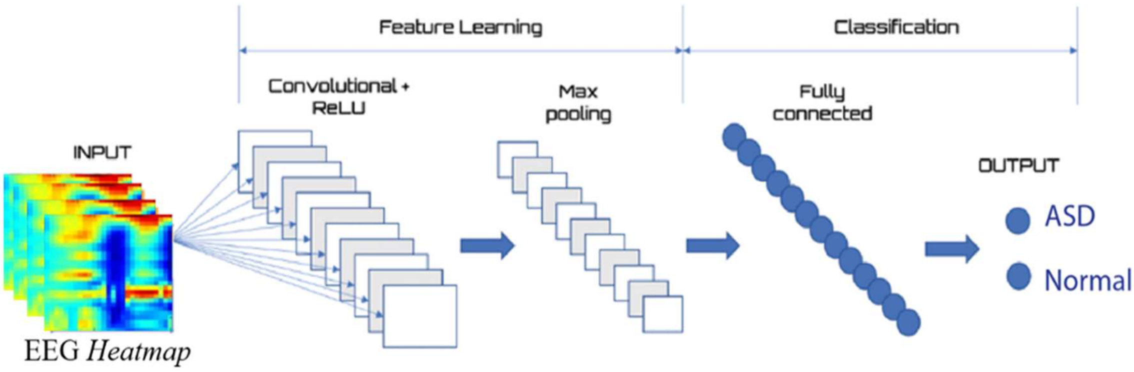


Fig. 2 Classification process using the Res-Net101 model

D. Model Deployment using Flask

Flask is a web framework from the Python programming language. Flask is used as the core and framework of the application. By using Flask and the Python language, developers can create APIs that are structured easily and do not require a particular tool or library to use [20]. This study used Python Flask to preprocess EEG data and run a trained deep-learning model for classifying EEG signals in ASD and normal patients. Flask can also return data as restAPI, which can be accessed on various platforms [21], [22]. In this study, the platform used was Android, which was built with the Flutter framework [23].

The trained TensorFlow model is deployed via Flask, a flexible Python framework suitable for small to medium models [24]. The deployment process involves configuring Flask, creating endpoints, and custom functions that download EEG data, perform preprocessing, and automatic classification. Testing uses model training data, and once successfully deployed, the API will be integrated with a mobile application for ASD diagnosis based on EEG data.

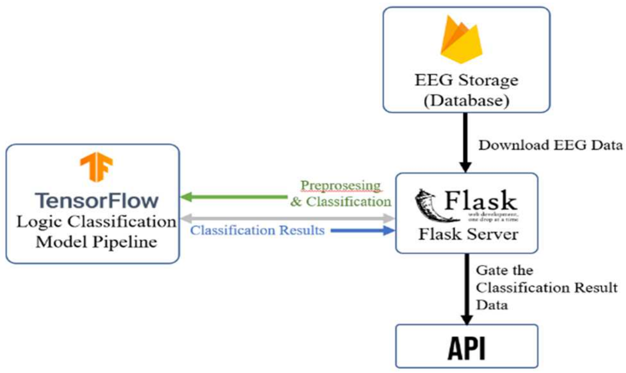


Fig. 3 Proposed backend mobile app structure

This Mobile Apps development uses the Agile method with an Extreme Programming (XP) approach for flexibility and continuous testing. Fig. 3 shows an ASD classification application with a front-end for uploading data and visualization of diagnosis results, as well as a back-end using Firebase for data storage and Flask for diagnosing EEG data in the Firebase database. This combination is expected to increase speed, quality, and user satisfaction.

E. Confusion Matrix

Confusion Matrix is a matrix that displays actual and predicted classification data. The confusion matrix is n x n in size, where n is the number of different classes [25]. Cross-validation was used to evaluate the performance of the proposed method [26]. Classification performance has been evaluated as parameters such as receiver operating characteristics (ROC), true positive (TP), true negative (TN), false positive (FP), and false negative (FN). Sensitivity, specificity, F1 score, and overall accuracy using the formula:

$$\text{Sensitivity} = \frac{TP}{TP+FN} \times 100 \quad (1)$$

$$\text{Specificity} = \frac{TN}{TN+FP} \times 100 \quad (2)$$

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+Fp+FN} \times 100 \quad (3)$$

$$\text{F1 Score} = 2\frac{TP}{2TP++FP+FN} \quad (4)$$

In the given equations, these terms are defined as follows:

- TP is the sum of the data from the actual class, true and correctly classified as a positive prediction class.
- TN is the amount of data from the actual true class but is classified as a negative prediction class.
- FP is the amount of data from the false class but is classified as a positive prediction class.
- FP is the sum of the data from the False class and is correctly classified as a Negative prediction class.

A. Flutter

Flutter is an open-source framework created by Google to build beautiful, native, and multi-platform applications such as Android, iOS, web, and desktop with only one codebase [14]. Flutter has two components: the software development kit (SDK), which is a useful tool for running applications on various platforms, and the UI Framework, which displays components such as buttons, text, and others according to customized needs. Another advantage of Flutter is that it has less complicated customizations so it can save development time [27], [28]. In this study, flutter will be integrated with Firebase Cloud Services to build a high-performance mobile application that allows users to easily input EEG data files and quickly get immediate diagnosis results.

B. Firebase Service

Firebase is a service owned by Google, with several features that can help developers develop the core functions of an application. With this service, application development can be more focused on the display section [29]–[32]. Firebase has several services that are very useful for mobile application development. Services on Firebase, such as authentication [32], apply more security for users. Not only that, Firebase also provides Cloud Firestore and Cloud Storage for storing databases and also allows users to upload all types of files to this cloud-based storage [28]. With this, mobile applications integrated with Firebase will continue to run at high performance without eating up additional local storage.

III. Results and Discussion

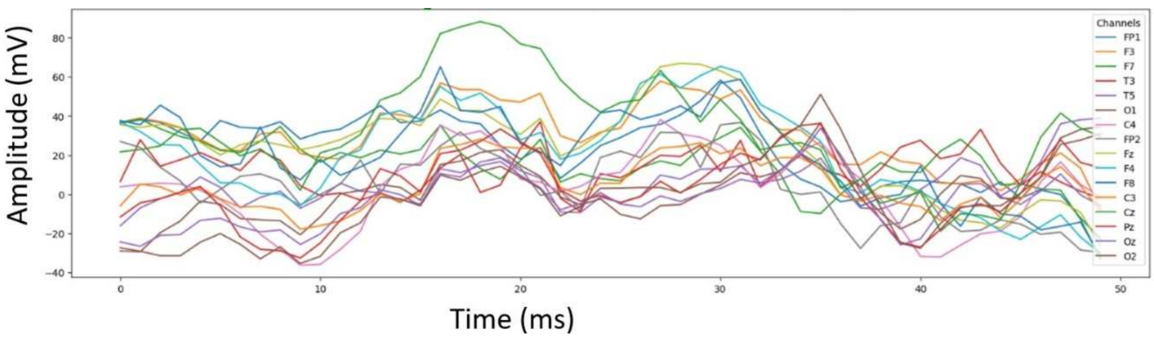
A. EEG Reading Result

Reading EEG data with the BCI2KReader produces a 2D array representing brain activity. EEG data in .dat format can only be read by this library in Python. Fig. 4 displays 16 channels of EEG signals over a period of 0 ms to 100 ms, with various colors representing each signal. This visualization provides a clear picture of brain activity during data collection, greatly assisting in analyzing brain patterns and exploring important findings in this study.

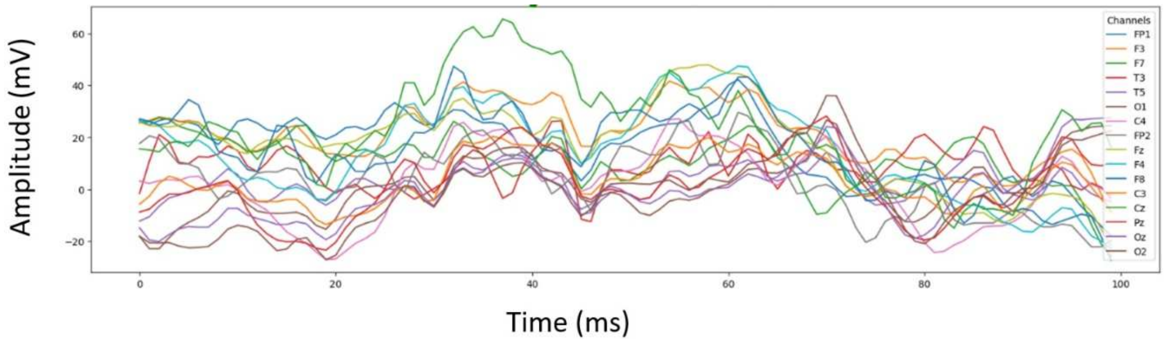
B. Dataset Preprocessing Result

1) Results of DWT Filter: The EEG signal has been cleaned from interference and noise using the DWT method. Although the visual changes are not significant, DWT at least cuts unwanted frequency components, such as electromagnetic noise or artifacts. The result is a cleaner and purer EEG signal, improving data quality for additional analysis.

2) Results of Windowing of EEG Signals: This section focuses on analyzing EEG signals with slicing techniques to expand the dataset. The initial EEG signal of varying lengths is cut into 10,000 ms long segments with an overlap of 5,000 ms, which is depicted in Fig. 5. The main goal is to increase the number of data samples for further analysis, enabling the identification of hidden patterns in the EEG signal. Table II contains details of the number of datasets generated from the initial EEG signal through this slicing technique, which is expected to improve the performance of the analysis model for recognizing important patterns and events in the EEG signal.



(a)



(b)

Fig. 4 (a) Unfiltered EEG, (b) Filtered EEG with DWT

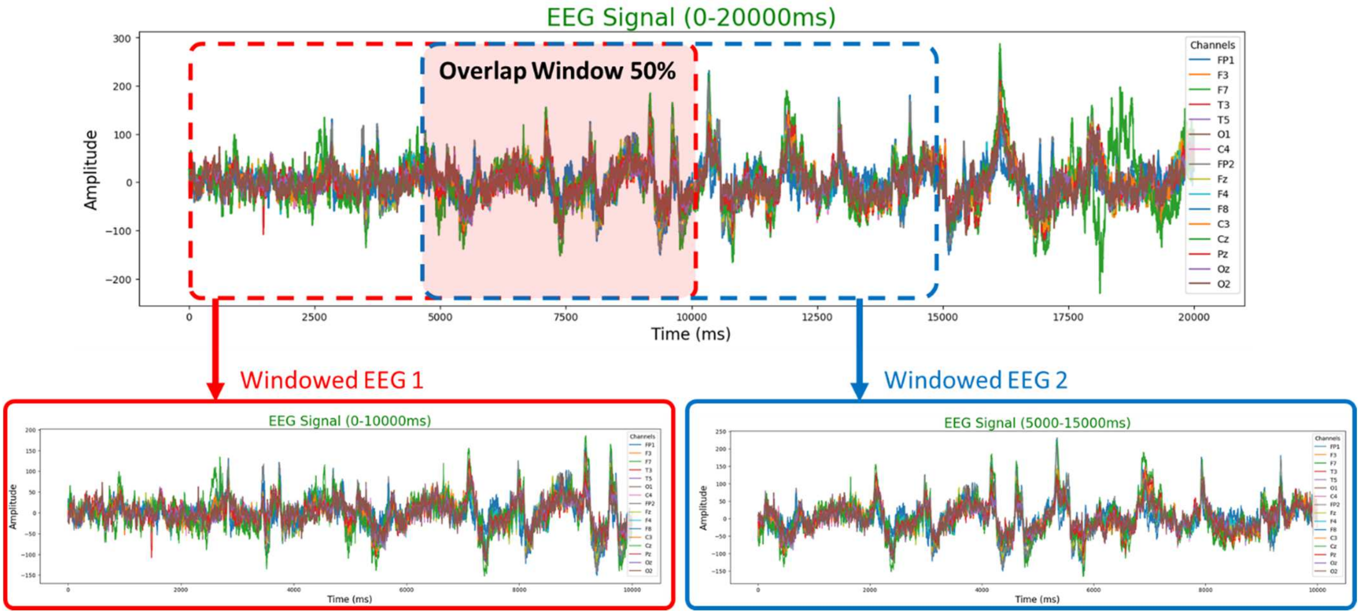


Fig. 5 Architectural schematic of a mobile application for classifying ASD sufferers using EEG signals

3) Implementation of EEG Data into Heatmap: Conversion of EEG signals to heatmap produces visualization of relative activity on the surface of the subject's head. The goal is to prepare data for the CNN model in subsequent analysis. Fig. 6 is the result of heatmap mapping, which visually depicts brain activity based on EEG recording data at a time position of 40 ms. Fig. 7 is the result of a heatmap visualization depicting brain activity during the complete recording period. This visualization allows analysis at a more comprehensive and in-depth time domain level.

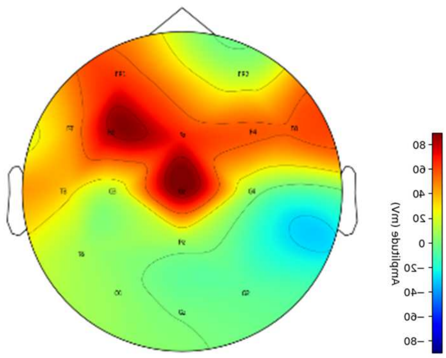
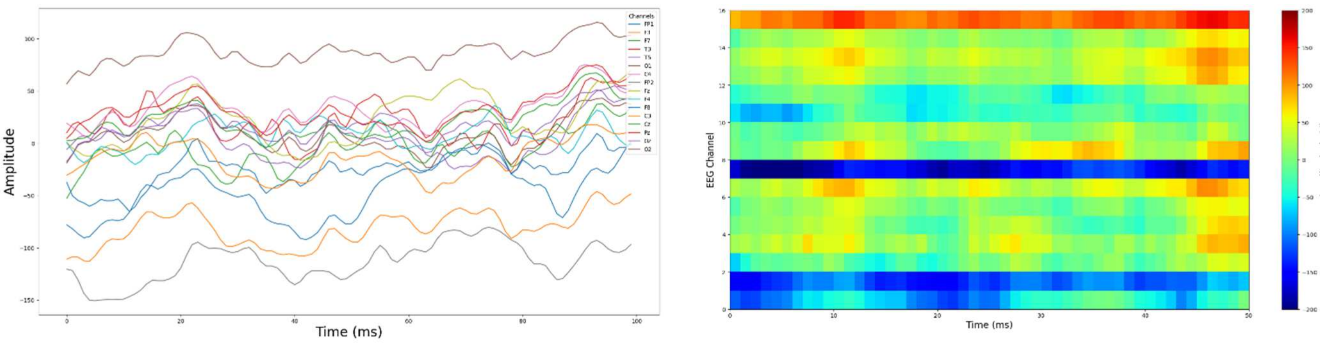


Fig. 6 Heatmap mapping results taken from EEG recording at 40 ms



(a) (b)

Fig. 7 Heatmap visualization of EEG

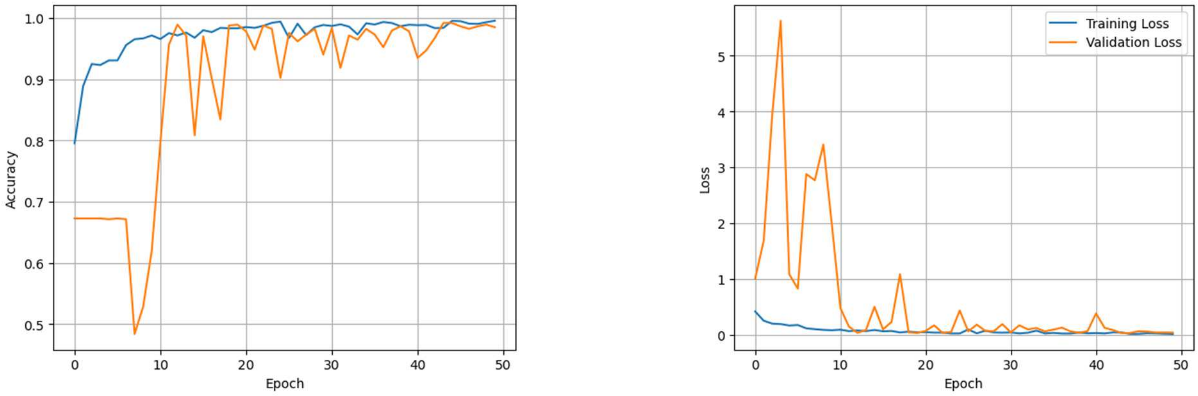
4) Training Parameters: Table I displays the classification model parameters used during training, with the cross-entropy loss function as a metric to measure the model's predictions.

TABLE I
CLASSIFICATION MODEL TRAINING PARAMETERS

Parameters	**CNN Model**
Epoch	50
Batch Size	32
Loss Function	Cross-entropy Optimizer
Optimizer	RMSProp
Learning Rate Optimizer	0.001
Epoch	50
Batch Size	32

The RMSProp optimizer is used with a learning rate of 0.001 to adjust the model weights and biases so that convergence toward optimal results occurs efficiently. The training results optimize the parameters of the classification model, allowing the CNN model to understand important patterns related to the target class and improve prediction accuracy through parameter adjustment by the RMSProp optimizer.

5) Classification Model Training: After training using Res-Net101 CNN technology, the model can identify complex patterns in the data. The analysis is carried out by examining the accuracy and loss graphs because the study focuses on implementing the model into a Mobile Application. The results in Fig. 8 showed that training accuracy reached 99.3%, while validation accuracy reached 98.9%. The two differences show that the model does not suffer from overfitting. This consistency indicates that the model has understood important patterns from the training data without sacrificing performance on the test data. With this level of accuracy, the trained CNN model has shown excellent performance in data classification.



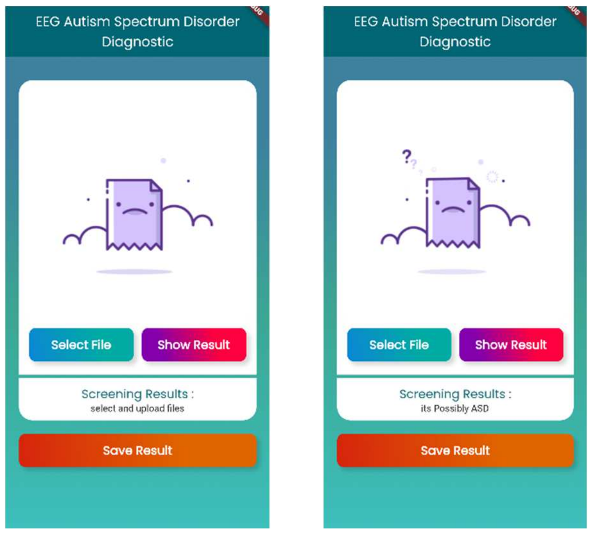
(a) (b)

Fig. 8 Results of the learning curve; (a) Accuracy and validation accuracy (b) Loss and validation loss

The loss and validation loss graphs from CNN model training showed a consistent decrease over several epochs. The training loss reached 0.0135, indicating an improvement in the model's predictions. A validation loss of 0.039 shows the model's ability to generalize well on test data that has never been seen before. The small difference between these two losses shows that the model is not experiencing overfitting. These results confirm that the Res-Net101 model has successfully understood important patterns from the training data, supporting various applications such as object recognition in images, medical data analysis, or anomaly detection in EEG signals for medical diagnosis and treatment.

C. Results of Mobile Apps and Integration with Trained Models for ASD and Normal EEG Classification

EEG data reading using BCI2KReader to understand human brain activity better. EEG data from KAU has the extension '.dat' and can only be read by the BCI2KReader library in Python programming. The reading results are a 2-dimensional Array where the x-axis represents the signal length, and y represents the signal amplitude. Meanwhile, the color of each signal represents the channel of each signal.



(a) (b)

Fig. 10 (a) Views before diagnosis, (b) Display after screening

Fig. 10 shows the results of the mobile application in this study, which has 2 main features, the first feature in Fig. 10 (a) is the ability to upload EEG data. Users can quickly and easily upload their EEG data to the app via a simple and user-friendly interface. After the data is uploaded, the application will automatically process and analyze the data to provide classification prediction results based on the pre-trained CNN model. Fig. 10 also shows another feature, namely the output of ASD or Normal diagnosis results, which are displayed in different displays for each category. Users will receive prediction results presented in an easy-to-understand format. The prediction results will show whether the EEG data falls into the ASD or normal category, with an intuitive and informative visual display.

Fig. 10 (b) shows that if the prediction results show that the data falls into the ASD category, the user will see a special display that provides additional information and useful resources related to autism spectrum disorder. Meanwhile, Fig. 10 (a) shows that if the prediction results show that the data falls into the normal category, the user will see a display that briefly explains the results.

1) Results of Mobile Apps that have been integrated with the CNN Model: This EEG classification application was created through the UI/UX design stage in Figma and implemented using the Flutter framework. All components are connected with Firebase and Flask integration to ensure the EEG classification function runs smoothly. The app has two main features: users can upload their EEG data to be analyzed using a CNN model. The prediction results for the ASD or Normal category are displayed clearly. If the data is classified as ASD, the user will get additional information about ASD; if the data is Normal, the user will receive a brief explanation of the results.

2) Mobile Apps Performance Results: As shown in Table II, the results of testing the EEG classification application system using Flutter and Flask highlight significant differences in the size of the EEG data files, with the largest reaching 12,794 Kb. Despite varying file sizes, EEG data upload times (E/DGL_Emulation) were relatively consistent, ranging from 71 ms to 133 ms. The performance of the classification model varied between files, with the lowest time per step being around 118 ms in the file 'Bader_Autism_24_11_2011S001R01.dat', and a better time of around 88 ms in the file 'Omran_Normal_5_5_2011S001R01.dat'. These results indicate that the characteristics of the processed EEG data influence the performance of the EEG classification application. Overall, these tests provide insight into the relationship between file size, upload time, and classification model performance in the context of Flutter and Flask. Further optimization is required to handle larger EEG data or obtain faster responses.

TABLE II
WINDOWING RESULT, FINAL DATASET NUMBER, AND APPLICATION PERFORMANCE

Dataset	Class	Data Size (Kb)	signal length (ms)	Final Number of Datasets	E/DGL_Emulation		Model Performance (ms/step)
					Upload EEG Data (ms)	Request Results Data API (ms)	
Amer_Normal_5_5_2011S001R01.dat	Normal	3.825	1356544	271	72	12	91
Amer_Normal_5_5_2011S001R03.dat	Normal	3.665	1356544	271	71	15	96
Dhelal_Normal_15_6_2011S001R02.dat	Normal	10.384	3692672	739	101	13	92
Mahmud_Normal_5_5_2011S001R01.dat	Normal	5.254	1865216	373	82	18	103
Omran_Normal_5_5_2011S001R01.dat	Normal	10.399	3698304	740	102	12	88
Bader_Autism_24_11_2011S001R01.dat	ASD	2.358	753408	151	62	16	118
Mada_Autism_26_5_2011S001R01.dat	ASD	9.972	3545984	709	98	16	116
Mada_Autism_26_5_2011S001R02.dat	ASD	430	147456	29	50	14	109
Mohammed_Autism_9_11_2011S01R01.dat	ASD	12.794	4535040	907	123	10	89
Nour_Autism_2_10_2011S001R01.dat	ASD	8.821	3136256	627	96	11	94
Nour_Autism_2_10_2011S001R02.dat	ASD	3.557	1261312	252	49	10	95
Saud_Autism_1_5_2011S001R01.dat	ASD	8.654	3073536	615	94	14	106
Shahad_Autism_5_6_2011S001R01.dat	ASD	14.162	5038464	1.008	133	14	101
			Total	6.692			

In this study, a mobile application for diagnosing ASD was developed, applying asynchronous EEG brain signal data. The study applied preprocessing techniques and a deep learning CNN model deployed through Python-Flask, facilitating diagnosis via restAPI on Android devices using the Flutter framework. The model demonstrated effective classification of ASD patients and normal individuals, presenting a potential framework for early neurodevelopmental disorder detection. M. N. A. Tawhid [33] introduced a system for automatic ASD detection using time-frequency spectrogram images of EEG signals. Machine learning (ML) and deep learning (DL) models were employed to classify spectrogram images, achieving remarkable accuracy rates of 99.15% with the DL-based model and 95.25% with the ML-based model on an ASD EEG dataset. This approach opens the door to the development of a computer-aided diagnosis system. In [1], a method for identifying ASD from EEG data was presented using time-frequency spectrogram images and applying preprocessing techniques, including re-referencing, filtering, and normalization. The study used principal component analysis (PCA) to reduce feature dimensions and achieved a noteworthy accuracy rate of 95.25% with the support vector machine (SVM) classifier in ten-fold cross-validation. This method promises improved efficiency and accuracy in ASD diagnosis, offering a practical tool for healthcare professionals.

Compared to existing research, this study stands out due to the development of an Android-based mobile application for early ASD diagnosis. The utilization of CNN for EEG signal analysis is a significant strength, offering robust classification. However, it's essential to recognize potential limitations, such as the focus on the Android platform, which may restrict its user base. Despite these limitations, this study represents a promising step towards accessible and automated tools for early ASD diagnosis and therapy.

IV. Conclusion

This research successfully built an Android Mobile Application using Flutter to differentiate normal and ASD EEG signals asynchronously. Test results showed an average delay of 100-238 ms and a latency of 50-133 ms. The automatic EEG signal preprocessing algorithm succeeded with 6692 efficient sub-signals. The Res-Net101 model yielded 98.7% (training) and 94.3% (validation) accuracy, allowing identification of ASD or Normal. Integration of the model into Android applications via Flask and Firebase has the potential for early detection of neurodevelopmental disorders. The main conclusion is that this study creates a potential framework for early detection of neurodevelopmental disorders based on EEG signal analysis, contributing to developing advanced clinical analysis methods for identifying potential disorders such as ASD.

Acknowledgment

We acknowledge Universitas Syiah Kuala and all parties that have contributed to this work.

References

[1] M. N. A. Tawhid, S. Siuly, and H. Wang, "Diagnosis of autism spectrum disorder from EEG using a time-frequency spectrogram image-based approach," Electron. Lett., vol. 56, no. 25, pp. 1372–1375, 2020, doi: 10.1049/el.2020.2646.

[2] S. Islam, T. Akter, S. Zakir, S. Sabreen, and M. I. Hossain, "Autism Spectrum Disorder Detection in Toddlers for Early Diagnosis Using Machine Learning," 2020 IEEE Asia-Pacific Conf. Comput. Sci. Data Eng. CSDE 2020, 2020, doi: 10.1109/CSDE50874.2020.9411531.

[3] A. K. Subudhi, M. Mohanty, S. K. Sahoo, S. K. Mohanty, and B. Mohanty, "Automated Delimitation and Classification of Autistic Disorder Using EEG Signal," IETE J. Res., vol. 69, no. 2, pp. 951–959, 2023, doi: 10.1080/03772063.2020.1844076.

[4] V. Kaya, S. Tuncer, and A. Baran, "Detection and classification of different weapon types using deep learning," Appl. Sci., vol. 11, no. 16, 2021, doi: 10.3390/app11167535.

[5] M. Jagadeesan, P. A. Selvaraj, M. Harikrishnan, T. Kamalavalli, and V. Jayakumar, "Behavioral Features based Autism Spectrum Disorder Detection using Decision Trees," 2021. [Online]. Available: http://annalsofrscb.ro

[6] A. Subramanyam and P. Patkar, "Autism Services in Maharashtra," Ann. Indian Psychiatry, vol. 6, no. 3, p. 218, 2022, doi:10.4103/aip.aip_92_21.

[7] M. J. Alhaddad et al., "Diagnosis autism by Fisher Linear Discriminant Analysis FLDA via EEG," Int. J. Bio-Science Bio-Technology, vol. 4, no. 2, pp. 45–54, 2012.

[8] L. P. A. Arts and E. L. van den Broek, "The fast continuous wavelet transformation (fCWT) for real-time, high-quality, noise-resistant time–frequency analysis," Nat. Comput. Sci., vol. 2, no. 1, pp. 47–58, 2022, doi: 10.1038/s43588-021-00183-z.

[9] S. Chatterjee, R. S. Thakur, R. N. Yadav, L. Gupta, and D. K. Raghuvanshi, "Review of noise removal techniques in ECG signals," IET Signal Process., vol. 14, no. 9, pp. 569–590, 2020, doi: 0.1049/iet-spr.2020.0104.

[10] T. Xu, Y. Zhou, Z. Hou, and W. Zhang, "Decode Brain System: A Dynamic Adaptive Convolutional Quorum Voting Approach for Variable-Length EEG Data," Complexity, vol. 2020, 2020, doi:10.1155/2020/6929546.

[11] S. Slobounov, M. Gay, B. Johnson, and K. Zhang, "Concussion in athletics: Ongoing clinical and brain imaging research controversies," Brain Imaging Behav., vol. 6, no. 2, pp. 224–243, 2012, doi:10.1007/s11682-012-9167-2.

[12] M. Sahu and R. Dash, A survey on deep learning: Convolution neural network (cnn), vol. 153, no. January. Springer Singapore, 2021. doi:10.1007/978-981-15-6202-0_32.

[13] W. L. Mao, H. I. K. Fathurrahman, Y. Lee, and T. W. Chang, "EEG dataset classification using CNN method," J. Phys. Conf. Ser., vol. 1456, no. 1, 2020, doi: 10.1088/1742-6596/1456/1/012017.

[14] H. Polat and H. D. Mehr, "Classification of pulmonary CT images by using hybrid 3D-deep convolutional neural network architecture," Appl. Sci., vol. 9, no. 5, 2019, doi: 10.3390/app9050940.

[15] V. Yaloveha, A. Podorozhniak, H. Kuchuk, and N. Garashchuk, "Performance comparison of CNNs high-resolution multispectral dataset applied to land cover classification problem," Radioelectron. Comput. Syst., vol. 106, no. 2, pp. 107–115, 2023, doi:10.32620/reks.2023.2.09.

[16] C. Y. Zhu et al., "A Deep Learning Based Framework for Diagnosing Multiple Skin Diseases in a Clinical Environment," Front. Med., vol. 8, no. April, pp. 1–13, 2021, doi: 10.3389/fmed.2021.626369.

[17] A. El Mouatasim and M. Ikermane, "Control learning rate for autism facial detection via deep transfer learning," Signal, Image Video Process., vol. 17, no. 7, pp. 3713–3720, 2023, doi: 10.1007/s11760-023-02598-9.

[18] U. Shukla, G. J. Saxena, M. Kumar, A. S. Bafila, A. Pundir, and S. Singh, "An improved decision support system for identification of abnormal EEG signals using a 1D convolutional neural network and Savitzky-Golay filtering," IEEE Access, vol. 9, pp. 163492–163503, 2021, doi: 10.1109/ACCESS.2021.3133326.

[19] G. Xu et al., "A Deep Transfer Convolutional Neural Network Framework for EEG Signal Classification," IEEE Access, vol. 7, pp. 112767–112776, 2019, doi: 10.1109/access.2019.2930958.

[20] Z. Khakim and S. Kusrohmaniah, “Dasar - Dasar Electroencephalography (EEG) bagi Riset Psikologi,” Bul. Psikol., vol. 29, no. 1, p. 92, 2021, doi: 10.22146/buletinpsikologi.52328.

[21] Z. A. T. Ahmed et al., "Facial Features Detection System to Identify Children with Autism Spectrum Disorder: Deep Learning Models," Comput. Math. Methods Med., vol. 2022, 2022, doi:10.1155/2022/3941049.

[22] F. W. Alsaade and M. S. Alzahrani, "Classification and Detection of Autism Spectrum Disorder Based on Deep Learning Algorithms," Comput. Intell. Neurosci., vol. 2022, 2022, doi:10.1155/2022/8709145.

[23] M. López and A. Peñalver, "Health Seek: A Deep Learning-Based Intelligent System to Aid Medical Diagnosis," J. Biomed. Sci. Eng., vol. 15, no. 01, pp. 66–81, 2022, doi: 10.4236/jbise.2022.151007.

[24] A. Namburu et al., "FPGA-Based Deep Learning Models for Analysing Corona Using Chest X-Ray Images," Mob. Inf. Syst., vol. 2022, 2022, doi: 10.1155/2022/2110785.

[25] M. M. Fahmy, "Confusion Matrix in Binary Classification Problems: A Step-by-Step Tutorial," J. Eng. Res., vol. 6, no. 5, 2022.

[26] J. Xu, Y. Zhang, and D. Miao, "Three-way confusion matrix for classification: A measure driven view," Inf. Sci. (Ny)., vol. 507, pp. 772–794, 2020, doi: 10.1016/j.ins.2019.06.064.

[27] N. N. Sari, M. N. Gani, R. A. Maharani Yusuf, and R. Firmando, "Telemedicine for silent hypoxia: Improving the reliability and accuracy of Max30100-based system," Indones. J. Electr. Eng. Comput. Sci., vol. 22, no. 3, pp. 1419–1426, 2021, doi:10.11591/ijeecs.v22.i3.pp1419-1426.

[28] J. Singh, S. Srivastva, D. Raj, S. Singh, and M. Junaid Rasool, "Flutter and Firebase making Cross-Platform Application Development Hassle-Free," Int. Res. J. Mod. Eng. Technol. Sci., vol. 4, no. 4, pp. 1819–1827, 2022, [Online]. Available: www.irjmets.com

[29] T. Vo, "WEB Application Development with React and Google Firebase," Turku University of Applied Sciences, 2006. doi:10.1007/978-1-4302-0154-0_3.

[30] I. P. G. A. Sudiatmika, K. H. S. Dewi, A. A. R. Jayaningsih, and W. W. Artana, "Application Using Android-based Firebase and JetPack Services for Thesis Guidance," 2nd Int. Conf. Gov. Educ. Manag. Tour. (ICoGEMT)+TECH, pp. 1–11, 2022.

[31] A. Amanuel, "Supabase vs Firebase: Evaluation of performance and development of Progressive Web Apps," no. May, 2022.

[32] D. Sharma and H. Dand, "Firebase as BaaS for College Android Application," Int. J. Comput. Appl., vol. 178, no. 20, pp. 1–6, 2019, doi: 10.5120/ijca2019918977.

[33] M. N. A. Tawhid, S. Siuly, H. Wang, F. Whittaker, K. Wang, and Y. Zhang, "A spectrogram image based intelligent technique for automatic detection of autism spectrum disorder from EEG," PLoS One, vol. 16, no. 6 June, pp. 1–20, 2021, doi:10.1371/journal.pone.0253094.