Semiconductor Optoelectronics
ISSN: 1001-5868

HARDWARE ACCELERATED CLASSIFIER FOR EARLY DETECTION OF ALZHEIMER'S DISEASE

B A Sujathakumari*, Sudharshan Patil Kulkarni, Mahammadarif Kasab, Girinatham Sai Jayanth, Raghavendra G* and Vishwas H N

Sri Jayachamarajendra college of Engineering, JSSSTU, Mysuru

Abstract

Alzheimer's Disease (AD), a horrible neurological disorder that affects millions of people worldwide. By utilising the ADNI dataset and Xilinx Vivado software for simulation and execution, this study suggests a novel approach for the early detection of Alzheimer's disease (AD). The enormous amount of neuroimaging and cognitive evaluation data in the ADNI dataset can be used to train and validate FPGA-accelerated AD detection algorithms. The project intends to improve the effectiveness and performance of the AD detection algorithms by integrating Vivado software, which is renowned for its abilities in designing, simulating, and implementing FPGA structures. The system may be able to accomplish quicker and more accurate AD detection through the modification and optimization of the selected AD detection algorithms for FPGA acceleration In order to increase the efficiency and throughput of the FPGA implementation, additional elements like memory management, pipelining, and parallelization are taken into consideration. By employing the ADNI dataset for an experimental demonstration of the FPGA-accelerated AD detection system, metrics like speedup, power consumption, and resource utilization can be compared to non-accelerated methodologies. The findings of the study indicate the potential advantages of using Vivado software to support early AD identification.

Keywords: *Alzheimer's disease,ADNI,Vivado Toolbox,FPGA,Xilinx,Hardware accelerators.*

1. Introduction

The cutting-edge software programme Vivado, created by Xilinx, has enormous potential for the early Alzheimer's disease (AD) detection market. Its capabilities go far beyond hardware integration, giving scientists and engineers a solid foundation for the creation and improvement of algorithms. A noteworthy aspect of Vivado is the High-Level Synthesis (HLS) tool, which enables researchers to design advanced machine learning algorithms specifically for AD detection. Researchers can process and analyse a wide range of AD-related data, such as neuroimaging scans, genetic data, and clinical data, by utilising Vivado HLS. Researchers may easily optimise their AD detection strategies because to the software's streamlined algorithm creation and optimisation processes. Furthermore, Vivado promotes algorithm portability, guaranteeing seamless platform integration and encouraging widespread implementation of modern AD detection methods. One of Vivado's key benefits is its capacity

to speed up the analysis of huge datasets, significantly cutting down on the amount of time needed for AD detection.

For researchers looking into Alzheimer's disease (AD), the ADNI (Alzheimer's Disease Neuroimaging Initiative) dataset is a useful tool. It is a long-term study that gathers and makes available information on clinical, imaging, genetic, and biomarker data from people with AD, people with mild cognitive impairment (MCI), and healthy individuals. Pre-processed photos from the ADNI dataset are taken into account for the algorithm, which produces the input vectors. The ADNI (Alzheimer's Disease Neuroimaging Initiative) dataset is a significant resource for scientists studying Alzheimer's disease (AD). It is a long-term study that collects and disseminates data on clinical, imaging, genetic, and biomarker information from AD patients, MCI patients, and healthy persons. The algorithm that generates the input vectors considers pre-processed images from the ADNI dataset.

Vivado is crucial in enabling researchers to utilise the potential of data analysis and machine learning algorithms in the context of Alzheimer's disease (AD) detection. The capabilities of Vivado can be used by researchers to investigate novel methods for AD detection and diagnosis. The software's substantial features, such as its user-friendly interface and wide-ranging libraries, enable researchers to create and put into use sophisticated algorithms that can analyse and decipher a variety of AD-related data sources. The examination of complicated datasets is made easier by Vivado's data visualisation tools, which also help researchers find patterns, correlations, and potential biomarkers connected to AD. Additionally, Vivado can be seamlessly integrated into current AD research workflows thanks to its compatibility with various software tools and frameworks, encouraging researcher collaboration and information sharing. Vivado remains at the forefront of the area of AD detection as it develops, giving researchers the resources and tools, they need to advance early diagnostic and intervention techniques.

2. Literature review

[1] In the paper published by M.J.N.Sampad, et al., the use of a field programmable gate array (FPGA) to enable live monitoring of single particle fluorescence analysis on an optofluidic device as part of a quick sample-to-answer method.The sensor has outstanding potential for use as a point of care (POC) diagnostic tool by validating real-time fluorescence detection of individual bacterial plasmid DNA at attomole concentrations.

[2] In the paper published by M.Elnawawy et al., there is discussion of the chaotic dynamics that are observed in the spread of epidemics and illnesses, which have been supported by several constructed mathematical models. To the best of our knowledge, no attempt has been made to realise any of these chaotic models in analogue or digital electronic form. The effective FPGA implementations of three distinct viral spreading models and one illness progression model are discussed in this paper. First, parameter sensitivity is quantitatively examined for the spreading models of the COVID-19, Ebola, and influenza viruses as well as the cancer disease progression model.

[3] According to the paper by N. Mccombe et al., even though machine learning approaches have the potential to enhance dementia diagnostic procedures, clinical practise frequently

struggles to adapt or apply research findings. Importantly, featureselection-based optimisation for dementia diagnosis has not yet taken the duration of administering diagnostic assessments into consideration.

[4] As per the paper published by M. Seyedbarhagh et al., a biologically computational model that includes IP3 receptors (IPR), a plasma membrane pump, a sarcoendoplasmic reticulum Ca2+ ATPase (SERCA) pump, ryanodine receptor channels, and general membrane leak is represented by a multiplier-less digital design with the Coordinate Rotation Digital Computer (CORDIC) algorithm.

After examining several research articles on diverse topics, we concluded that creating hardware accelerators for neural networks would be beneficial since it would greatly increase computing efficiency. When used as an accelerator, FPGA completes the task very quickly. It also makes it possible to put different diagnostic techniques into practice. The FPGA is the chosen architecture for such complicated neural analysis, despite the fact that initial training takes some time. FPGA may be implemented with many different devices and designed to work with a wide range of additional instruments. The effectiveness and efficiency of the machine learning models would rise with the introduction of neural networks. Different pre-processing methods have already been used in other articles.

3. Methodology

This section provides more information on how to use the vivado toolbox for AD early detection. Vivado 2021 includes every element required for the simulation based on the relevant specifications. Results are obtained by taking vectors from the programme code and mapping them to the component parts. Different ML and hybrid combination methodologies are implemented, and after that, vectors are mapped to the design in the vivado.

Figure 1's block diagram illustrates how the model operates; the MRI images used as input are taken from the ADNI website because it has standardised and verified data. Bias is provided before to model training, and the model is then trained. In the Vivado toolbox, the vectors are mapped and the design is obtained.

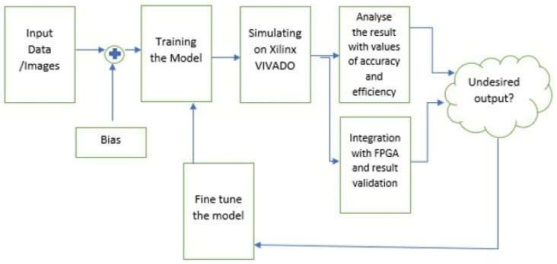


Figure 1 Block Diagram

One of FPGAs' key advantages is their distributed architecture, which enables efficient parallel processing and flexible execution of complex designs. In a distributed design, an FPGA is made

up of a large number of programmable logic blocks (PLBs) that are connected to one another by a network of programmable interconnects. Flip-flops and adaptable logic elements (CLEs), which can be programmed to perform certain functions, are present in each PLB. In general, the distributed architecture of FPGAs provides significant flexibility and scalability for the realisation of difficult digital designs. Real-time circuit reconfiguration, parallel processing, and efficient resource management are all made possible. FPGAs are useful for a range of applications, including high-performance computing, machine learning, image and video processing, and signal processing because to their characteristics.

The distributed architecture of Field-Programmable Gate Arrays (FPGAs) has a number of advantages over the series and parallel architectures. The inclination for distributed architecture is supported by the following reasons: FPGAs with distributed architecture provide high degrees of flexibility and reconfigurability. The programmable interconnects and logic blocks allow for dynamic reconfiguration of the circuitry and make it simple for designers to create and modify complicated designs. This versatility is especially valuable in settings where design alterations or adaptations are regularly required. Multiple operations can be run simultaneously because to FPGAs' distributed architecture. The parallel execution of numerous design components is made possible by multiple logic blocks.

This parallelism results in increased throughput and faster calculation when compared to a strictly series architecture. Resource Utilisation: The distributed architecture of FPGAs enables efficient utilisation of available resources. Instead of relying on a single processing element or a small number of parallel resources, FPGAs with distributed architecture can make the most of the available logic blocks and interconnects to maximise resource utilisation. As a result, overall performance increases and hardware resources are utilised more skillfully. Scalability: Distributed architecture offers scalability in terms of design complexity. FPGAs, which already include a large number of logic blocks, can be expanded to accommodate more complex designs. This scalability enables the development of larger, more sophisticated systems without sacrificing performance.

The input to the architecture model is a vector with a size of (784,1) rows and columns that comes from the software algorithm of the machine learning and deep learning model. To normalise the data and take into account just rounded off values, bias is introduced. 30 layers make up the sub modal layers, while relu is the employed activation function. Different classifier approaches are simulated using Vivado.

CNN approach:

The early Alzheimer's disease diagnosis method based on CNN. Utilising the Vivado Toolbox, a full FPGA design and development environment, the CNN method is refined and implemented. The Vivado Toolbox provides high-performance hardware acceleration, enabling the real-time processing and analysis of medical images. The CNN model is trained using a substantial collection of brain images, including magnetic resonance imaging (MRI) scans. Deep learning techniques, feature extraction, and data augmentation are utilised in the training phase to find the complex patterns and biomarkers related to Alzheimer's disease. The

trained CNN model is then deployed on the FPGA using the Vivado Toolbox to enable quick and efficient inference. The design architecture of CNN is depicted in Figure 2, which comprises of logic gates connected to muxes in a pattern. A ROM for memory is connected, as well as shift registers.

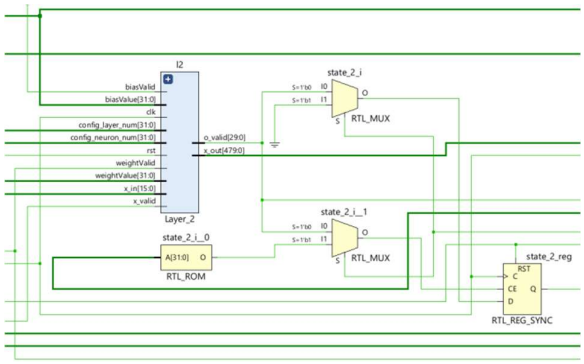


Figure 2 CNN architecture

CNN + RF:

The CNN and the Random Forest classifier are the two fundamental components of a hybrid model. The CNN component uses structural brain images to extract high-level characteristics because these images are known to show specific patterns associated with AD. The Vivado Toolbox can be used to develop and implement the CNN model in a thorough and efficient manner. The CNN model is trained using a substantial collection of brain images, including instances from AD and non-AD. A model's ability to generalise can be improved by using transfer learning techniques, such as improving a pre-trained CNN model (such as VGG16 or ResNet), which makes use of the knowledge gained from similar tasks. The trained CNN extracts distinctive features from the input pictures, which are subsequently fed into the Random Forest classifier. The Random Forest classifier, a potent ensemble learning technique, uses the gathered attributes as input and performs classification using a variety of decision trees. The Vivado Toolbox makes it simple to build and optimise the Random Forest method, enabling efficient resource utilisation and parallel processing on FPGA devices. To ensure correct model usage, the hybridised model uses registers for the random forest. Figure 3 shows design architecture of CNN+RF.

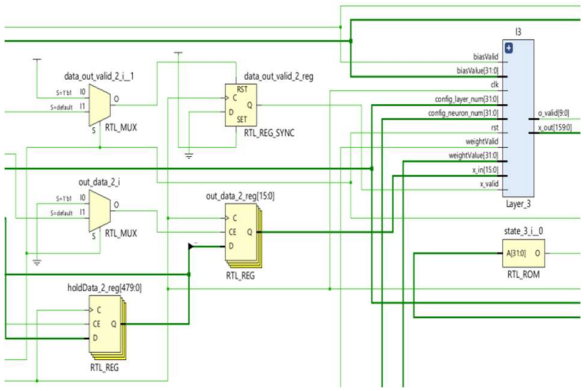


Figure 3 CNN+RF architecture

CNN + SVM:

Early Alzheimer's disease detection using a hybrid technique. The two main components of the model are a CNN for extracting features from images of the brain and an SVM classifier for categorising disorders. We describe the architecture of the CNN, including the number of layers, the size of the filters, and the activation methods. We also go over the SVM classifier and how it is trained within the Vivado Toolbox framework. To create a hybrid model for SVM, logic gates are coupled in patterns. Hybridization improves memory utilisation and logic analysis. Figure 4 shows design architecture of CNN+SVM.

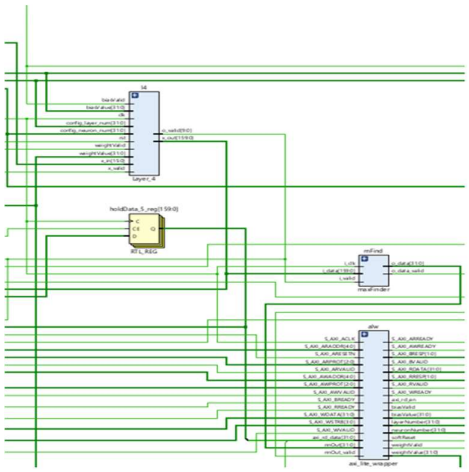


Figure 4 CNN+SVM architecture

RNN:

The RNN algorithm is implemented using the Vivado Toolbox, a powerful FPGA design and implementation environment. The specific FPGA device and resources are chosen with care, taking into account the constraints and architecture-imposed optimisations for efficient processing. The RNN algorithm and FPGA acceleration work together to expedite high-speed computations and parallel processing, enabling quick and accurate AD detection. During the training and validation phase, the dataset is split into training and evaluation sets. The model is trained using the appropriate loss functions, optimisation methods, and hyperparameter tuning to get the optimum performance. Performance measures including accuracy, precision, recall, and F1-score are used to assess the model's performance and ascertain its capability. Figure 5 shows design architecture of RNN.

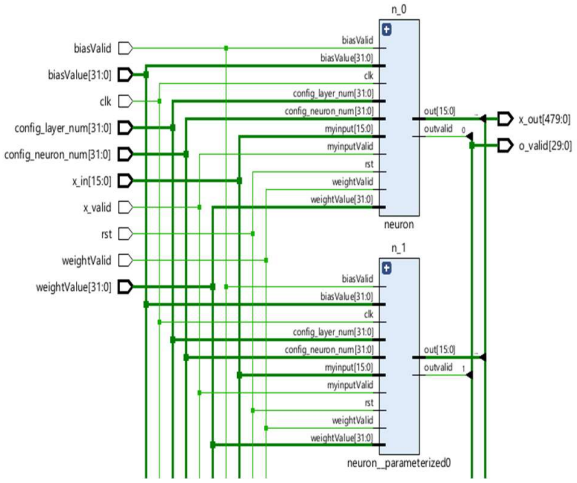


Figure 5 RNN architecture

RNN + RF:

In the hybrid method, we first use an RNN to simulate the temporal dependencies in the input data. RNNs' propensity for sequential data processing makes them excellent at detecting temporal trends. The RNN takes input sequences and learns the underlying patterns by using recurrent connections inside its architecture. The model can then collect information from prior time steps and incorporate it into the present prediction. Following the RNN stage, an ensemble of Random Forest (RF) models are trained using the hidden states or RNN outputs as features. A decision tree's classification predictions are integrated into the final classification using the RF ensemble learning technique. Strengths of RF include strong categorization capabilities and processing of high-dimensional data.The RNN-generated features are sent into the RF ensemble, allowing the RF to take advantage of the temporal data obtained by the RNN. The RF ensemble then performs additional classification based on these features, increasing accuracy and generalisation by utilising the variety of decision trees. Figure 6 shows design architecture of RNN+RF.

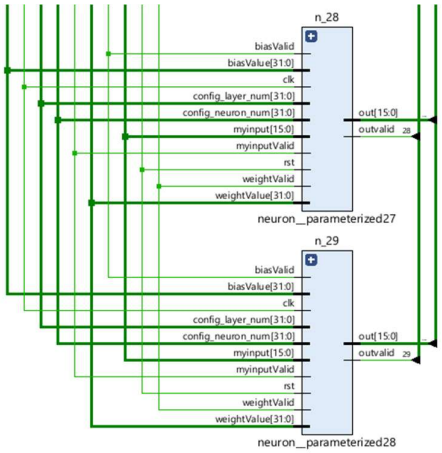


Figure 6 RNN+RF architecture

RNN + SVM:

The hybrid technique combines the discrimination capabilities of Support Vector Machines (SVM) and Recurrent Neural Networks (RNN) for temporal modelling. The RNN component gives the model the ability to distinguish trends and changes over time by locating temporal correlations in the input data. The SVM component uses the learned properties of the RNN as a classifier to distinguish between individuals with Alzheimer's disease and those who are healthy. The combination of RNN and SVM allows the model to utilise both local and global data, improving the accuracy of disease identification. Utilising its hardware design and acceleration capabilities, Vivado Toolbox is used to implement and optimise the hybrid model. Figure 7 shows design architecture of CNN+RF.

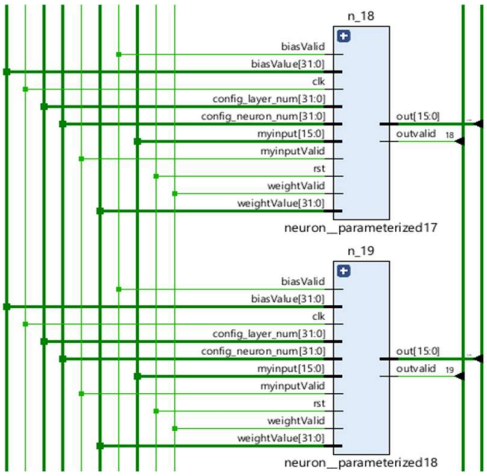


Figure 7 RNN+SVM architecture

4. Results

Due to its greater utilisation and inclusion of both serial and parallel features, distributed architecture is taken into consideration. Slice LUTs, the fundamental building blocks of the FPGA unit, are mostly used in the result analysis. Additionally, it informs us about the functions of other parts like flip-flops, latches, and muxes. On the basis of a utilization percentage, the optimum mode of classifier is taken into consideration.

The results are with respect to the LUT utilization and memory consumption. The hybrid models show near same percentage for utilization. Here is a detailed report of the results obtained from the vivado simulation. The results contain on-chip power management which has dynamic and static division. Individual percentage is considered for logic, RAM coverage, digital signal processing and input output. BUFG is the global buffer which stores the output.

RNN

Figure 2 shows results obtained after RNN classifier simulation.

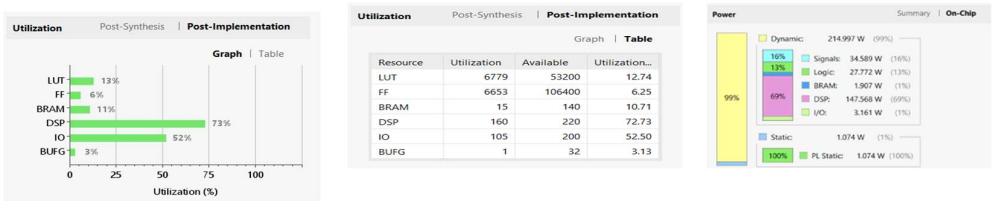


Figure 2 Results of RNN

The post-synthesis and post-implementation results of the RNN are obtained with the same number of LUTs used in synthesis and implementation, a one-unit reduction in FF, and the same amounts of BRAM, DSP, IO, and BUFG. On-chip memory splits its power consumption between Dynamic and Static. Power consumption for signals is 34.589W, logic is 27.772W, BRAM is 1.907W, DSP is 147.568W, and I/O is 3.161W.

RNN + RF.

Figure 3 shows results obtained after RNN+RF classifier simulation.

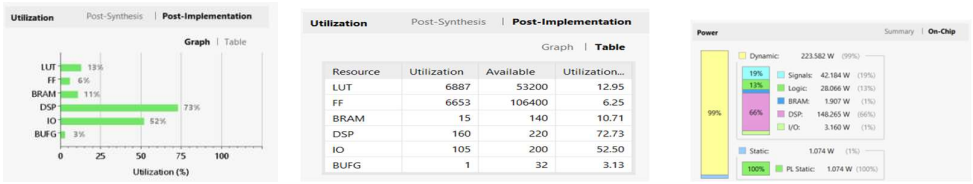


Figure 3 Results of RNN+RF

Results of RNN+RF's post-synthesis and post-implementation are obtained; 6895 LUT were used in synthesis and 6887 LUT were used in implementation; this decrease in LUT shows that RNN+RF is superior to RNN; FF is reduced by one in the implementation; and BRAM, DSP, IO, and BUFG are used the same in both. On-chip memory splits its power consumption between Dynamic and Static. Power consumption breakdown: Signals use 42.184W, Logic use 28.066W, BRAM use 1.907W, DSP use 148.265W, and I/O use 3.160W.

RNN + SVM

Figure 4 shows results obtained after RNN+SVM classifier simulation.

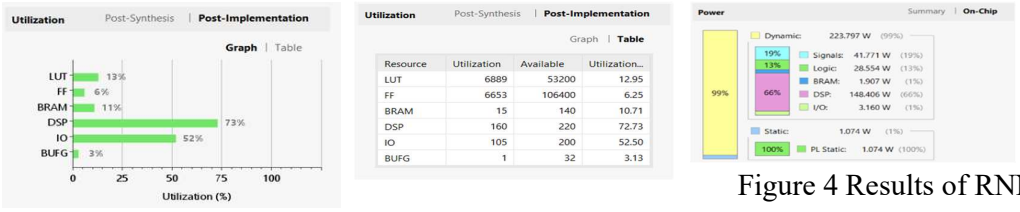


Figure 4 Results of RNN+SVM

Results of RNN+SVM post-synthesis and post-implementation have been obtained; 6898 LUTs were used in synthesis and 6889 LUTs were used in implementation; this reduction in LUTs demonstrates that RNN+RF is superior to RNN. In addition, FF is reduced by one in the implementation, and BRAM, DSP, IO, and BUFG were used the same in both phases. On-chip memory splits its power consumption between Dynamic and Static. Power consumption for signals is 41.771W, logic is 28.554W, BRAM is 1.907W, DSP is 148.406W, and I/O is 3.160W.

CNN

Figure 5 shows results obtained after CNN classifier simulation.

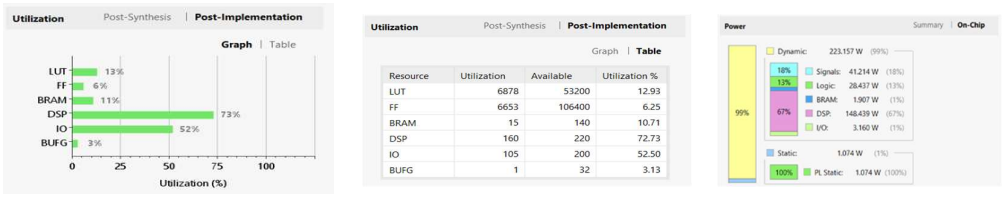


Figure 5 Results of CNN

Results of post-synthesis and post-implementation for CNN have been obtained; 6890 LUT were used in synthesis and 6878 LUT were used in implementation. The decrease in LUT indicates that CNN code uses less LUT post implementation, which lowers power consumption. FF is also reduced by one in the implementation, and BRAM, DSP, IO, and BUFG are used the same in synthesis and implementation. On-chip memory splits its power consumption between Dynamic and Static. Power consumption for signals is 41.214W, logic is 28.437W, BRAM is 1.907W, DSP is 148.439W, and I/O is 3.160W.

CNN + RF

Figure 6 shows results obtained after CNN+RF classifier simulation.

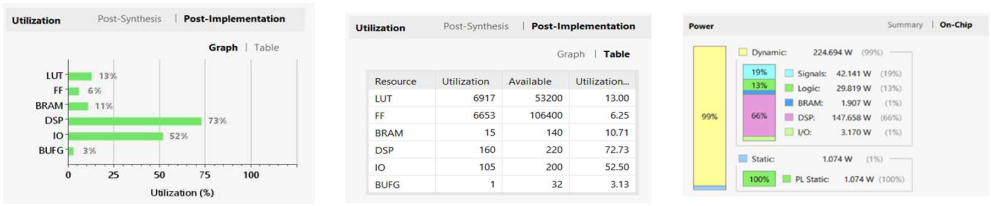
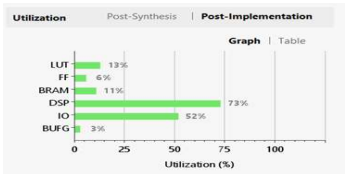


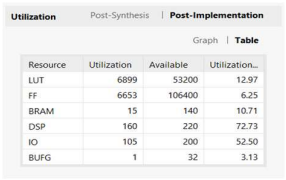
Figure 6 Results of CNN+RF

The results of CNN+RF's post-synthesis and post-implementation phases have been obtained; 6927 LUTs were used for synthesis and 6917 for implementation. The reduction in LUT usage indicates that CNN+RF code uses fewer LUTs after implementation, which lowers power consumption. FF is also reduced by one in the implementation, and BRAM, DSP, IO, and BUFG are used the same in both phases. On-chip memory splits its power consumption between Dynamic and Static. Power consumption breakdown: Signals use 41.141W, Logic use 29.819W, BRAM use 1.907W, DSP use 147.658W, and I/O use 3.170W.

CNN + SVM

Figure 7 shows results obtained after CNN+SVM classifier simulation.





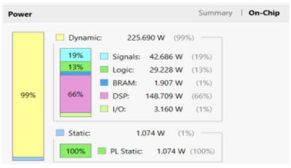


Figure 7 Results of CNN+SVM

The results of CNN+SVM's post-synthesis and post-implementation phases have been obtained; 6908 LUTs were used in synthesis and 6889 LUTs were used in implementation. The decrease in LUT usage indicates that CNN+SVM code uses fewer LUTs after implementation, which lowers power consumption. FF is also reduced by one in the implementation, and BRAM, DSP, IO, and BUFG are used the same in synthesis and implementation. On-chip memory splits its power consumption between Dynamic and Static. Power consumption for signals is 42.686W, logic is 29.228W, BRAM is 1.907W, DSP is 148.709W, and I/O is 3.160W.

Based on the above results we can have an overall picture of the utilization percentage as shown in Figure 8 of each classifier model.

MODEL	LUT	FF	BRAM	DSP	IO	BUFG	Muxes
CNN	12.95	6.25	10.71	72.73	52.50	3.13	0.12
CNN+RF	13	6.25	10.71	72.73	52.50	3.13	0.12
CNN+SVM	12.97	6.25	10.71	72.73	52.50	3.13	0.12
RNN	12.74	6.25	10.71	72.73	52.50	3.13	0.12
RNN+RF	12.96	6.25	10.71	72.73	52.50	3.13	0.12
RNN+SVM	12.97	6.25	10.71	72.73	52.50	3.13	0.12

Figure 8 Comparison of models

We can clearly see from the table that most of the parameters remains same for all the models and LUT utilization percentage is differential factor. RNN model has least value of 12.74% and CNN+RF model has max of 13% utilization. It is to be noted that all the values need to be less than 100% to have a proper utilization.

5. Conclusion

Hardware solutions that enable speedy and effective data processing allow for real-time analysis and decision-making. Early detection of AD typically requires processing enormous volumes of data, such as brain imaging or biomarker data, and hardware solutions are better able to handle the computational needs than software-based methods. Hardware solutions can be optimised for power efficiency for wearable or implantable devices used to monitor individuals who are at risk for AD over time. If these devices need to operate continuously on limited power sources, hardware optimisation can save energy usage. Each design was analysed and compared to the software component based on the LUT's utilisation percentage.

References

[1] M. J. N. Sampad, M. N. Amin, A. R. Hawkins and H. Schmidt, "FPGA Integrated Optofluidic Biosensor for Real-Time Single Biomarker Analysis," in IEEE Photonics Journal, vol. 14, no. 1, pp. 1-6, Feb. 2022, Art no. 3701806, doi: 10.1109/JPHOT.2021.3127484.

[2] M. Elnawawy et al., "FPGA Realizations of Chaotic Epidemic and Disease Models Including Covid-19," in IEEE Access, vol. 9, pp. 21085-21093, 2021, doi: 10.1109/ACCESS.2021.3055374.

[3] N. Mccombe et al., "Alzheimer's Disease Assessments Optimized for Diagnostic Accuracy and Administration Time," in IEEE Journal of Translational Engineering in Health and Medicine, vol. 10, pp. 1-9, 2022, Art no. 4900809, doi: 10.1109/JTEHM.2022.3164806.

[4] M. Seyedbarhagh, A. Ahmadi and M. Ahmadi, "Digital Realization of Ca2+ Oscillation with Impact of Amyloid-β," 2021 IEEE International Midwest Symposium on Circuits and Systems (MWSCAS), 2021, pp. 665-668, doi: 10.1109/MWSCAS47672.2021.9531702.

[5] M. Chen, L. Zu, H. Wang and F. Su, "FPGA-Based Real-Time Simulation Platform for Large-Scale STN-GPe Network," in IEEE Transactions on Neural Systems and Rehabilitation Engineering, vol. 28, no. 11, pp. 2537-2547, Nov. 2020, doi: 10.1109/TNSRE.2020.3027546.

[6] Y. Liu, Y. Chen, W. Ye and Y. Gui, "FPGA-NHAP: A General FPGA-Based Neuromorphic Hardware Acceleration Platform with High Speed and Low Power," in IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 69, no. 6, pp. 2553-2566, June 2022, doi: 10.1109/TCSI.2022.3160693.

[7] P. Possa, D. Schaillie and C. Valderrama, "FPGA-based hardware acceleration: A CPU/accelerator interface exploration," 2011 18th IEEE International Conference on Electronics, Circuits, and Systems, 2011, pp. 374-377, doi: 10.1109/ICECS.2011.6122291.

[8] W. Huang et al., "FPGA-Based High-Throughput CNN Hardware Accelerator with High Computing Resource Utilization Ratio," in IEEE Transactions on Neural Networks and Learning Systems, vol. 33, no. 8, pp. 4069-4083, Aug. 2022, doi: 10.1109/TNNLS.2021.3055814.

[9] S. Sreekrishna Yadav and V. Mittal, "FPGA Implementation of segmented feature fusion in MRI images using wavelet," 2019 International Conference on Communication and Electronics Systems (ICCES), 2019, pp. 1946-1951, doi: 10.1109/ICCES45898.2019.9002032.

[10] C. Crema et al., "Embedded platform-based system for early detection of Alzheimer disease through transcranial magnetic stimulation," 2018 IEEE Sensors Applications Symposium (SAS), 2018, pp. 1-6, doi: 10.1109/SAS.2018.8336774.

[11] G. Ribeiro, N. Neves, S. Santander-Jiménez and A. Ilic, "HEDAcc: FPGA-based Accelerator for High-order Epistasis Detection," 2021 IEEE 29th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM), 2021, pp. 124-132, doi: 10.1109/FCCM51124.2021.00022.

[12] Fadhel, M.A., Al-Shamma, O., Oleiwi, S.R., Taher, B.H., Alzubaidi, L. (2020). Real-Time PCG Diagnosis Using FPGA. In: Abraham, A., Cherukuri, A., Melin, P., Gandhi, N. (eds) Intelligent Systems Design and Applications. ISDA 2018 2018. Advances in Intelligent Systems and Computing, vol 940. Springer, Cham. https://doi.org/10.1007/978-3-030-16657-1_48

[13] R. Ju, C. Hu, p. zhou and Q. Li, "Early Diagnosis of Alzheimer's Disease Based on Resting-State Brain Networks and Deep Learning," in IEEE/ACM Transactions on Computational Biology and Bioinformatics, vol. 16, no. 1, pp. 244-257, 1 Jan.-Feb. 2019, doi: 10.1109/TCBB.2017.2776910.

[14] Li K, Wang J, Li S, Yu H, Zhu L, Liu J, Wu L. Feature Extraction and Identification of Alzheimer's Disease based on Latent Factor of Multi-Channel EEG. IEEE Trans Neural Syst Rehabil Eng. 2021; 29:1557-1567. doi: 10.1109/TNSRE.2021.3101240. Epub 2021 Aug 10. PMID: 34329166.

[15] W. Zhu, L. Sun, J. Huang, L. Han and D. Zhang, "Dual Attention Multi-Instance Deep Learning for Alzheimer's Disease Diagnosis with Structural MRI," in IEEE Transactions on Medical Imaging, vol. 40, no. 9, pp. 2354-2366, Sept. 2021, doi: 10.1109/TMI.2021.3077079.

[16] A. Diaz-Perez and M. A. Garcia-Martinez, "FPGA Accelerator for RNA Secondary Structure Prediction," 2009 12th Euromicro Conference on Digital System Design, Architectures, Methods and Tools, 2009, pp. 667-671, doi: 10.1109/DSD.2009.225.

[17] SujathaKumari, B. & Kulkarni, Sudarshan & Sultana, Ayesha. (2022). "Hybrid Classification Algorithm for Early Prediction of Alzheimer's Disease." 10.1007/978-981-19-3391-2_29.

[18] Sujathakumari, B.A., Charitha Shetty, M., Lakshitha, H.M., Mehulkumar, P.J., Suma, S. (2021). Predictive Analysis for Early Detection of Alzheimer's Disease. In: Jeena Jacob, I., Kolandapalayam Shanmugam, S., Piramuthu, S., Falkowski-Gilski, P. (eds) Data Intelligence and Cognitive Informatics. Algorithms for Intelligent Systems. Springer, Singapore. https://doi.org/10.1007/978-981-15-8530-2_56