Journal of Applied Engineering and Technological Science
Vol 7(1) 2025:224-246

CNN-BASED SIBI SIGN LANGUAGE RECOGNITION ALPHABET: EXPLORING THE IMPACT OF HARDWARE ON MODEL TRAINING

Aris Rakhmadi[1*], Anton Yudhana[2], Sunardi[3]
Department of Informatics Engineering, Universitas Muhammadiyah Surakarta, Indonesia[1]
Department of Informatics, Universitas Ahmad Dahlan, Yogyakarta, Indonesia[1]
Department of Electrical Engineering, Universitas Ahmad Dahlan, Yogyakarta, Indonesia[2,3]
aris.rakhmadi@ums.ac.id

Received: 28 January 2025, Revised: 07 May 2025, Accepted: 07 October 2025
**Corresponding Author*

ABSTRACT
The recognition of Sign Language Alphabets (SLA) plays a vital role in human-computer interaction, especially for individuals with auditory disabilities. This study aims to evaluate the impact of different hardware configurations—specifically CPU, GPU, and memory setups—on the training efficiency and recognition performance of a Convolutional Neural Network (CNN)-based model for SLA using the SIBI dataset. The novelty of this research lies in its focus on hardware-aware deep learning optimization for Indonesian sign language (SIBI), an underexplored area. The model was trained on 3,468 labeled hand gesture images representing 24 SIBI alphabet signs. Experiments were conducted on CPU (Intel Xeon 2.00 GHz) and GPU (Nvidia Tesla T4) platforms using a consistent CNN architecture. The training time was significantly reduced by 45.5%, from 1 hour 39 minutes to just 54 minutes, while the accuracy remained consistent at 96.7%, showing no significant change between the two setups. These results demonstrate the significance of parallel processing and memory bandwidth in enhancing model convergence and generalization. The findings are relevant for real-time SLA deployment with hardware constraints on embedded or mobile platforms. Overall, the study underscores the importance of hardware optimization in accelerating CNN training and improving performance in sign language recognition systems.
Keywords: *Sign Language Recognition, Alphabets, CNN, Hardware, Impact*

1. Introduction

Communication is fundamental to human interaction, enabling individuals to share information, thoughts, and feelings. It allows people to engage in social, professional, and personal exchanges that form the basis of human connection. Language, as a primary tool for communication, has evolved into numerous forms, including verbal and nonverbal methods. One of the most essential nonverbal methods is sign language, which is essential in enabling communication for people who are deaf or mute (Fadlilah et al., 2021a). These individuals frequently encounter considerable difficulties conveying themselves using spoken language, making sign language an essential alternative for daily communication.

Unlike spoken languages, sign language uses visual cues, predominantly through hand movements and gestures, to convey meaning. Each sign in a sign language system corresponds to a specific idea, concept, or object, and these signs are formed through precise combinations of finger positions, hand shapes, and motions (Rakhmadi et al., 2024). For those who use it, sign language functions as a dynamic and expressive mode of communication. However, a significant barrier exists when people unfamiliar with sign language attempt to interact with those who rely on it, leading to communication breakdowns. This issue is particularly evident in Indonesia, where two primary sign language systems— Bahasa Isyarat Indonesia (BISINDO) and Sistem Isyarat Bahasa Indonesia (SIBI)—are used by different segments of the deaf community (Fadlilah et al., 2019).

In the context of Indonesian sign language, SIBI serves as a standardized system developed to align with the grammatical structure of the Indonesian language (Darmawan et al., 2023). Unlike the more natural and regionally varied BISINDO, SIBI is structured and formally used in educational and official settings (Suharjito et al., 2021). One key component of SIBI is its manual alphabet, which consists of hand signs representing each letter in the Indonesian alphabet. This alphabet plays a crucial role in spelling out names, technical terms, and words that lack established signs. As such, the SIBI alphabet not only supports literacy and language learning among the deaf community but also provides a foundational element for developing sign language recognition technologies in Indonesia (Wijaya et al., 2024).

Despite its importance, sign language adoption is not widespread in society, as it is not commonly taught in schools or used in everyday interactions. There are many sign languages worldwide, and each is developed to suit its community's linguistic and cultural context (Subburaj & Murugavalli, 2022). These include Arabic Sign Language (ArSL), British Sign Language (BSL), Indonesian Sign Language (SIBI), and American Sign Language (ASL) (Al Moustafa et al., 2024). However, the majority of the population struggles to communicate effectively with deaf individuals, further marginalizing the community. The lack of familiarity with these diverse sign languages emphasizes the necessity for creative solutions to enhance communication and close the gap between the deaf and hearing communities.

While American Sign Language (ASL) and other international sign languages have been widely studied and supported by extensive datasets, SIBI remains relatively underrepresented in sign language recognition research (Handayani et al., 2022). Focusing on SIBI is essential due to its regional and educational significance in Indonesia, where it is officially recognized and used in formal settings such as schools for the deaf. Moreover, the linguistic structure of SIBI aligns closely with the national language, making it a crucial tool for literacy and academic development among deaf students (Safitri et al., 2024). Despite its importance, there is a notable lack of publicly available SIBI datasets and limited research on its recognition through artificial intelligence. Addressing this gap not only effectively supports the Indonesian deaf community but also contributes to the diversification of global SLR research by highlighting non-English, region-specific sign languages.

Sign language is a complex system requiring individuals to master gestures with specific meanings. The system relies heavily on visual cues, with hand shapes, finger positions, and movement patterns representing multiple concepts. The gestures involved are dynamic and often intricate, making recognizing sign language an intellectually demanding task (Hekmat et al., 2022). Understanding these movements is crucial for the effective transmission of ideas through sign language, and failure to interpret them correctly can lead to confusion and miscommunication.

Given these challenges, technology has become an effective tool for identifying and understanding sign language. With advancements in biometric systems, it is now possible to identify and analyze numerous forms of human expression, such as fingerprints, palm prints, voice patterns, and even hand gestures (Rakhmadi & Ariyanto, 2021). These technologies have opened the door for the development of automated systems capable of translating and recognizing sign language, offering a hopeful solution to the communication challenges encountered by the deaf community.

One key area where technological progress has significantly impacted is digital image classification (Kwenda et al., 2023). This field, which involves categorizing images based on content, is crucial in various sectors, including healthcare, security, agriculture, and business (Wu et al., 2023). Digital image classification is a significant challenge in computer vision, where the goal is to enable machines to recognize and categorize visual information automatically (Z. Zhang et al., 2023). Historically, this process required complex algorithms and intensive computational resources, but recent advancements have led to more efficient and accurate image processing and classification.

A promising method for sign language recognition (SLR) is Convolutional Neural Networks (CNNs), which have demonstrated exceptional performance in image classification tasks, particularly for hand gesture interpretation. Unlike traditional neural networks like Multi-Layer Perceptrons (MLPs), which often require extensive preprocessing and handcrafted feature extraction, CNNs can automatically learn spatial features directly from gesture images (Fadlilah et al., 2021b). This capability is crucial for recognizing static hand signs that vary in orientation, scale, and position—challenges commonly encountered in real-world SLR scenarios (Muis et al., 2024).

In SLR tasks, especially alphabet-based systems like SIBI, detecting fine-grained finger shapes and hand orientations is vital for accurate classification. CNNs excel in capturing these intricate patterns, making them highly effective in differentiating between visually similar gestures. Moreover, CNN-based models have shown resilience to environmental variations such as inconsistent lighting, occlusions, and diverse signer backgrounds (Irvanizam et al., 2023).

Recent advances in deep learning frameworks like TensorFlow have further enabled the deployment of CNN-based SLR systems, even on resource-constrained devices (Luong, 2023). However, these benefits come with high computational demands, choosing training hardware (CPU, GPU, memory) a critical factor in optimizing model performance, training speed, and real-time responsiveness—especially for embedded or mobile applications.

Substantial hardware enhancements have supported the swift advancement of deep learning, particularly with the introduction of advanced computing systems with high-performance and Graphics Processing Units (GPUs). GPUs, in particular, are well-suited for the parallel processing tasks required by deep learning models, drastically reducing the time needed for training complex models (Fadlilah et al., 2022). As a result, deep learning techniques have become more practical and feasible for real-world applications, including hand gesture recognition and sign language translation.

In this study, we explore using a model based on CNN to identify and classify the SIBI alphabet, focusing on recognizing hand gestures. Hand gesture recognition plays a key role in sign language interpretation, as it accurately identifies the gestures used to communicate specific words and concepts. This research seeks to improve the accuracy and efficiency of sign language recognition (SLR) systems by harnessing the capabilities of CNNs and advanced hardware to close the communication gap between the deaf community and the general public.

Recognition of hand gestures through deep learning models offers significant potential for improving accessibility and fostering inclusivity (Abdallah et al., 2013). Automating SLR allows for developing systems that translate sign language into text or speech, facilitating real-time communication. These systems can be incorporated into various devices, including smartphones and smart glasses, simplifying communication between deaf individuals and hearing individuals in daily interactions.

The findings of this study have broader implications for advancing communication technologies, particularly those designed to assist individuals with disabilities. The primary objective of this research is to analyze how variations in hardware configurations—such as CPU/GPU and memory—affect the performance and training time of sign language recognition (SLR) systems. By investigating the influence of hardware on model efficiency, this study aims to optimize machine learning and computer vision capabilities, supporting the development of more effective and inclusive SLR systems. Ultimately, this research contributes to efforts to build a society where individuals can communicate openly and effectively, regardless of physical abilities.

2. Literature Review

Over recent years, numerous studies have explored machine learning methods, mainly deep learning and CNN, for recognizing signs in various sign languages, including alphabet-based systems (Hashi et al., 2024). These methods typically involve training models on large, diverse datasets to enhance accuracy and generalization. CNNs, in particular, have demonstrated significant potential in recognizing complex patterns within hand gestures, making them ideal for SLR tasks (Waluyo et al., 2023). The widespread use of CNNs for image-based recognition has led to notable advancements in the field, where performance improvements are often tied to the amount and value of data utilized for training.

Table 1 summarizes key themes from research on SLR, emphasizing the importance of machine learning, mainly CNN, in improving recognition accuracy. It covers developing real-time gesture recognition systems that combine video and sensor data. It focuses on overcoming challenges such as hand shape variability, environmental factors (e.g., lighting), and signer-dependent differences. Creating comprehensive, diverse datasets is emphasized, ensuring better model generalization. Hardware considerations also play a significant role, with research showing that high-performance GPUs, edge devices, and hardware accelerators (e.g., FPGAs, ASICs) can drastically improve model training and real-time processing. However, balancing model accuracy with real-time processing performance, particularly on resource-limited mobile and embedded systems devices, necessitates hardware-aware optimization methods like quantization, pruning, and knowledge distillation. Future directions highlight the need for energy-efficient, cost-effective hardware solutions to ensure accessibility and scalability across different environments.

Table 1 - Overview of Key Themes and Challenges in SLR Research

Theme	Key Points	Approaches	Findings
Machine Learning & Deep Learning. (Alsharif et al., 2023a)	Focus on using CNNs for SLR, especially for alphabet-based systems.	CNNs for image-based recognition and training on large datasets	CNNs show potential in recognizing hand gestures. Performance is tied to dataset size and diversity.
Real-Time Gesture Recognition. (Abdallah et al., 2023)	Development of real-time systems using video and sensor data for on-the-fly sign translation (text or speech).	Combination of computer vision and sensor data (hand movements, facial expressions, posture).	Depth sensors (e.g., Kinect) improve recognition under changing conditions (lighting, occlusions).
Dataset Creation & Curation. (Duarte et al., 2021)	Creation of comprehensive datasets for alphabet sign languages to advance recognition systems.	Building diverse datasets from global communities.	Datasets ensure diverse signing styles and gestures for more generalized recognition models.
Improving Robustness. (Ben Atitallah et al., 2022)	Overcoming variability in hand shapes, orientations, and signer-dependent characteristics.	Data augmentation, transfer learning, multi-modal learning.	Variability in gestures (speed, size) presents challenges; augmentation and multi-modal approaches improve generalization.
Environmental Factors. (Schulder et al., 2023)	Impact of lighting, background noise, and environmental conditions on recognition quality.	Accounting for environmental factors in sensor data.	Ecological factors significantly affect model performance.
Comparative Studies. (Svendsen & Kadry, 2023)	Comparing accuracy across different sign languages, especially alphabet-based vs. iconic systems.	Comparative analysis of sign languages.	Unique challenges in recognizing alphabet signs (rapid and subtle movements); comparisons to other sign languages provide insights.
Multimodal Systems. (Lu et al., 2023)	Integration of visual and sensor-based data for improved recognition accuracy.	Combining video with accelerometer/gyroscope data.	Multimodal systems improve robustness against occlusions and poor lighting.
Hardware Impact on Performance. (Rawal et al., 2023)	Advances in hardware, including GPUs and edge devices, improving deep learning model training and deployment.	Comparing different hardware platforms.	Hardware significantly affects model accuracy, training speed, and scalability. High-performance hardware speeds up training but is costly.
Real-Time Processing & Hardware Trade-	Trade-offs between model accuracy and real-time processing	Hardware optimizations for real-time recognition.	Hardware limitations affect model size, complexity, and real-time deployment.

offs. (Hussin et al., 2021)	capabilities in mobile and embedded systems.		Optimizing CNN architectures for limited resources is essential.
Hardware-Aware Optimization. (Nguyen et al., 2023)	Adapting hardware-aware optimization techniques (pruning, quantization, distillation) to improve performance on resource-constrained devices.	Pruning, quantization, knowledge distillation.	Hardware-aware optimizations are needed to balance accuracy with computational efficiency, particularly in mobile or embedded systems.
Hardware Accelerators (FPGAs, ASICs). (Hu et al., 2022)	Hardware accelerators (FPGAs, ASICs) can be used to improve real-time recognition speed and efficiency.	Exploring specialized hardware for real-time applications.	FPGAs and ASICs enhance speed and reduce power consumption, but further research is needed for real-time, alphabet-based SLR.
Scalability & Accessibility. (Tang et al., 2023)	Issues around accessibility of high-performance hardware and its impact on system scalability in low-resource environments.	Analyzing the scalability of sign language systems.	The accessibility gap in hardware affects model performance and scalability, particularly in low-resource environments.
Future Directions. (Abdullah et al., 2024)	Growing interest in energy-efficient, cost-effective hardware solutions for widespread deployment of SLR systems.	Energy-efficient hardware development.	Advances in computational power must be paired with energy-efficient solutions for real-world adoption across various environments.

Several previous studies have successfully implemented Convolutional Neural Networks (CNNs) for Sign Language Recognition (SLR) tasks, demonstrating high accuracy yet revealing certain limitations. A study explored five deep learning models for static American Sign Language (ASL) alphabet recognition using a large dataset of over 87,000 images, achieving exceptional results—ResNet-50 reached the highest accuracy at 99.98%, followed by EfficientNet (99.95%) and ConvNeXt (99.51%), though the VisionTransformer model underperformed with 88.59% (Alsharif et al., 2023b, 2023a). Another work addressed dynamic sign recognition by combining a 1D CNN with GRU and the MediaPipe framework for real-time applications on mobile devices. Their lightweight approach achieved notable accuracies of 99.84% (LSA64), 98.8% (DSL-46), and 88.4% (LIBRAS-BSL), but emphasized the challenge of optimizing performance under hardware constraints (Abdalla et al., 2023; Abdallah et al., 2023). A different approach introduced a unique CNN-based system using Electrical Impedance Tomography (EIT) to detect muscle activity associated with hand signs, achieving 95.94% accuracy (Ben Atitallah et al., 2022). While innovative in sensing modality, this method is limited by lower accuracy than vision-based approaches and complexity in practical deployment. CNNs demonstrate strong potential for SLR, but challenges remain in balancing accuracy, hardware efficiency, and generalization across subjects and modalities.

Prior research has increasingly emphasized the critical role of computing resources, such as GPU acceleration, memory throughput, and hardware specialization, in determining deep learning models' efficiency, scalability, and practicality. For instance, one study demonstrated the impact of architectural design on computational efficiency by introducing hardware-optimized CNN models for arrhythmia classification. While the software-based Supreme CNN Architecture (SCA) achieved 99.17% accuracy, its hardware counterpart (HCA) reached a slightly lower

97.34% accuracy but consumed only 628 mW of power on a ZYNQ Ultrascale FPGA—66.95% less than existing 1D-CNN hardware, highlighting a significant trade-off between accuracy and power efficiency (Rawal et al., 2023). Similarly, another work presented a hardware-accelerated ResNet system on an FPGA board, which optimized computation through inter-layer fusion, dynamic quantization, and tightly pipelined channel-level parallelism. This implementation achieved 98.87% accuracy while consuming only 2.136 W of power, demonstrating a practical balance between energy efficiency and performance for gesture recognition tasks (Yang et al., 2024). In addition, analog in-memory computing has been proposed as an energy-efficient method for accelerating deep learning workloads. Despite introducing approximation errors due to hardware non-idealities, this approach, coupled with hardware-aware retraining, enabled large-scale models such as convolutional networks and transformers to retain iso-accuracy compared to their floating-point counterparts (Rasch et al., 2023).

Further research has investigated cost-performance trade-offs and energy efficiency in alternative computing paradigms. For example, a comparative study deployed spiking neural networks (SNNs) on Intel's Loihi neuromorphic platform and deep neural networks (DNNs) on Intel's Neural Compute Stick 2 (NCS2) to classify American Sign Language gestures. While the SNNs showed slightly lower accuracies—99.30% for the ASL alphabet and 99.03% for digits—compared to the DNNs, they achieved up to 20.64× lower power consumption and 4.10× higher energy efficiency, demonstrating the potential of neuromorphic systems in low-power applications (Mohammadi et al., 2022). Another study evaluated the latency of deep learning models such as MobileNetV3 when deployed on high-performance workstations and low-power devices like the Raspberry Pi 4. The results showed significant delays on edge devices, with inference times for MobileNetV3 Large increasing from 50 ms on a workstation to 348 ms on the Raspberry Pi 4, underscoring the performance gap across platforms (Tarek et al., 2022). Additionally, a comprehensive performance analysis of DNN inference on SIMD-enabled CPUs revealed that CPUs could outperform GPUs in certain low-latency tasks due to better branch prediction, reduced cache misses, and efficient dynamic activation handling, making them suitable for embedded and constrained environments (H. Li et al., 2023). These studies collectively highlight the importance of aligning model complexity, hardware design, and deployment context to optimize deep learning performance and energy use.

Another key area of research has been the advancement of real-time gesture recognition classifications (Nguyen et al., 2023). These systems leverage video and sensor data to identify sign language motions on the fly, facilitating the real-time conversion of signs into speech or text. These approaches merge computer vision methods with sensor data to detect inputs such as hand gestures, facial expressions, and body posture, improving the precision and contextual comprehension of the signs being performed. Some studies have employed depth sensors, like Microsoft Kinect, to improve gesture recognition under various environmental conditions, such as lighting changes or occlusions.

Creating and curating comprehensive datasets for alphabet sign languages have been crucial in advancing recognition systems (Latif et al., 2019). High-quality datasets are crucial for training and assessing the performance of machine learning models. Several research initiatives have focused on building these datasets, with notable examples being those that encompass sign language from diverse global communities. These datasets are invaluable for ensuring the representation of various signing styles, allowing recognition systems to be trained on gestures and variations in sign languages.

Besides dataset creation, there is a rising interest in strengthening the robustness of SLR systems. Some studies have focused on overcoming challenges, such as hand figure variability, orientation differences, and signer-dependent characteristics like size and speed of gestures. Approaches like data augmentation, transfer learning, and multi-modal learning have been explored to address these challenges, thereby improving the generalization of models across different environments and individuals. Furthermore, some research has highlighted the significance of considering environmental factors like lighting and background noise, which can have significant impact on the quality of video or sensor data.

Additionally, comparative studies on the recognition accuracy of different sign languages have been conducted, highlighting the unique features of alphabetic systems compared to other forms of sign languages (Sultan et al., 2022). These comparisons help identify specific challenges in recognizing alphabet signs, such as the rapid and often subtle movements required for accurate identification. Studies comparing alphabet sign languages to iconic or conceptual sign languages emphasize the additional challenges posed by the greater variety of gestures and the more complex hand and arm movements involved in specific languages.

Another aspect of sign language research is the integration of multiple sensory modalities for improved recognition accuracy (Subandi et al., 2024). Multimodal systems that combine visual inputs (e.g., video of hand gestures) with other sensor-based data (such as accelerometer or gyroscope information) have been shown to improve recognition performance, particularly in challenging environments. By fusing data from various sources, these systems can offer greater robustness against issues such as occlusions or poor lighting, which are common in real-world settings.

Impact of Hardware Architectures on SLR Performance

Hardware is also a key factor in influencing the performance of SLR systems. Advances in hardware, such as more powerful GPUs and specialized hardware like edge devices, have enabled efficient training and deployment of larger, more complex models. Research comparing the impact of different hardware platforms on model training has shown that computational resources can drastically affect training times, accuracy, and overall system performance. Specifically, high-performance computing infrastructure is often necessary for training deep learning models on large, diverse datasets. At the same time, more affordable devices may limit the size and complexity of models that can be deployed in real-time applications.

Some studies have also explored the trade-offs between model accuracy and real-time processing capabilities (Subramanian et al., 2022). This is particularly relevant in mobile and embedded systems, where hardware limitations such as processing power, memory, and battery life can significantly impact the feasibility of deploying complex deep-learning models for real-time SLR. As a result, a growing body of work has focused on optimizing CNN architectures to reduce computational requirements without sacrificing recognition accuracy.

The impact of hardware on the training and deployment of SLR models has emerged as a critical yet underexplored area of research. As deep learning architectures for SLR—particularly Convolutional Neural Networks (CNNs)—continue to grow in complexity, the reliance on high-performance computational resources such as GPUs and specialized accelerators becomes increasingly vital. These hardware platforms expedite training on large datasets and enable deploying sophisticated models capable of handling nuanced gesture variations (Rawal et al., 2023). However, despite these advancements, a noticeable gap exists in accessibility to such computational power, particularly among researchers and developers operating in resource-constrained settings. This gap raises significant concerns regarding the scalability and inclusivity of SLR systems, especially when considering their implementation across a wide range of hardware environments and deployment contexts.

This lack of inclusivity is particularly pressing in region-specific sign languages, such as SIBI, where research on hardware-aware model adaptation remains limited. Alphabet-based sign languages, which demand rapid, fine-grained recognition, introduce additional computational challenges due to the precision and real-time performance required. Although optimization techniques like quantization, pruning, and model distillation have shown promise in reducing computational load, few studies have specifically addressed their application to SLR models trained on localized or underrepresented datasets like SIBI. Moreover, while hardware accelerators such as FPGAs and ASICs have been explored for general-purpose SLR tasks, comprehensive investigations into how these platforms influence the accuracy, latency, and energy efficiency of alphabet-centric SLR systems, particularly those designed for regional sign languages, remain scarce. As a result, there is a pressing need for more targeted research that bridges the gap between hardware efficiency and region-specific SLR implementation to ensure broader accessibility and practical deployment in real-world scenarios (Hu et al., 2022).

Studies have demonstrated that hardware configurations used during model training directly influence the precision, efficiency, and scalability of Sign Language Recognition (SLR) systems (Pitonak et al., 2022). High-end computational resources—such as Graphics Processing

Units (GPUs), Tensor Processing Units (TPUs), and specialized accelerators—enable the parallel processing of massive data batches and complex operations across multiple layers of deep networks. This parallelism significantly accelerates the training of large-scale models by distributing matrix operations over thousands of cores, thus reducing training time and increasing the feasibility of deeper architectures. Conversely, resource-constrained environments, such as mobile devices or embedded systems, necessitate optimization strategies due to limited memory bandwidth, lower clock speeds, and fewer compute cores. These limitations affect not only the batch size and number of epochs that can be used during training but also restrict the complexity and depth of the models that can be practically deployed. Consequently, hardware-aware deep learning techniques—such as model quantization, pruning, and knowledge distillation—are essential to maintain model accuracy while minimizing resource consumption (Arya et al., 2022).

Deploying SLR systems in real-time contexts introduces further complexity due to the need for low latency and high throughput. Real-time gesture recognition demands models that can process continuous data streams, often video or sensor input, with minimal delay. This necessitates optimized architectures for accuracy, efficient memory access patterns, and low-latency inference. Theories of memory hierarchy and bandwidth constraints become particularly relevant here, as slow memory access or cache misses can become major bottlenecks in real-time systems. To overcome these issues, lightweight models such as MobileNet or EfficientNet are often deployed alongside hardware accelerators like Application-Specific Integrated Circuits (ASICs) and Field Programmable Gate Arrays (FPGAs). These devices support pipelined and parallel execution, improving throughput while maintaining energy efficiency. Moreover, system-level optimization, such as channel parallelism and inter-layer fusion, can maximize memory reuse and computational throughput on limited hardware (Ferraz et al., 2022). Collaboration between model architecture and hardware design is critical in enabling scalable, high-performance SLR systems, particularly in latency-sensitive or power-constrained applications.

Recent advancements in edge-device SLR systems further demonstrate the potential of low-cost and compact hardware for real-time gesture recognition. For example, a novel robotic hand gesture recognition system has been successfully developed using a Raspberry Pi platform integrated with single-mode–multimode fiber (S–M) structures and a microcamera. This system effectively analyzes specklegrams generated from the bending of robotic fingers and utilizes a CNN model to classify up to 30 distinct gestures with an impressive accuracy of 99.6% (Zhang et al., 2023). The study highlights the feasibility of high-precision gesture recognition on resource-constrained devices by leveraging specialized sensing mechanisms and lightweight computation. Moreover, it confirms such systems' reliability and long-term stability, underscoring the potential of Raspberry Pi-based deployments in domains such as robotic interaction, human motion tracking, and wearable intelligent gloves. This work provides a valuable precedent for SLR research aiming to optimize performance under limited hardware resources.

The evolution of hardware technologies continues to shape the trajectory of SLR systems, enabling more accurate and robust models through increased computational power. However, most existing studies have focused on widely used sign languages such as ASL or BSL, with limited attention to region-specific systems like SIBI. This lack of research creates a critical gap in understanding how hardware optimization strategies, such as model compression, quantization, and accelerator deployment, can be adapted to the unique linguistic and structural features of SIBI. As SLR technologies move closer to real-world implementation, particularly in low-resource or mobile settings, examining the interplay between hardware capabilities and model performance within the context of local sign language systems becomes essential. This study addresses that gap by exploring hardware-aware optimization for SIBI-based SLR, aiming to contribute to the development of scalable, efficient, and inclusive recognition systems tailored for diverse user communities.

3. Research Methods

This study investigates the impact of hardware on CNN performance in recognizing hand gestures from the SIBI alphabet. The methodology involves several crucial steps: dataset collection, data preprocessing, CNN model design, and a comparison of model training performance across different hardware platforms (CPU vs. GPU).

Dataset Collection and Preparation

Figure 1 provides a comprehensive illustration of the SIBI alphabet, showcasing 24 distinct hand gestures representing the letters A through Y, with J and Z omitted because of their dynamic characteristics. Each gesture is depicted clearly and standardized, highlighting the precise hand shapes and orientations required for accurate representation. The images are presented against a uniform background to enhance visual clarity and eliminate distractions, making it easier to discern the subtle differences between gestures. This visual guide acts as a reference for interpreting the dataset. It underscores the importance of maintaining consistency in gesture formation, which is critical for effectively training machine learning models like CNNs.

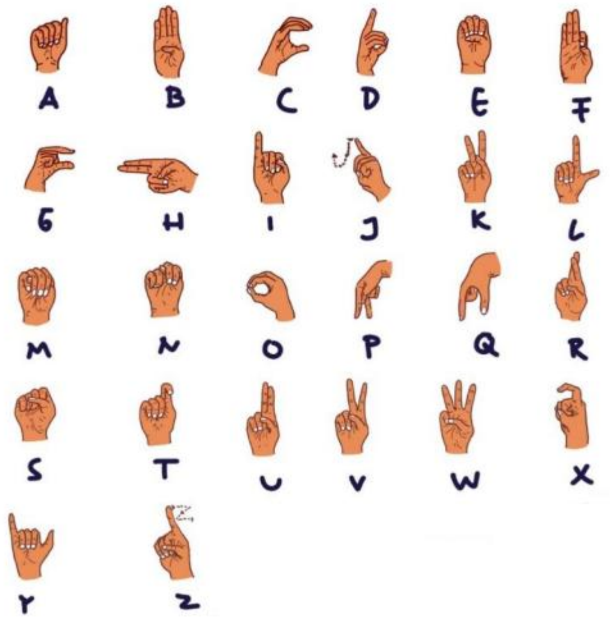


Fig. 1. SIBI Sign Language Alphabet: Hand Gestures Representing Letters A to Z

A dataset forms the foundation for training machine learning models, particularly CNNs, which are highly effective for image classification tasks. This study's dataset consists of hand gesture images representing the SIBI sign language alphabet, which includes 24 distinct classes corresponding to the letters A through Y, excluding J and Z due to their unique movement characteristics. In total, 3,468 images were collected from multiple participants to capture a variety of gestures. The images, with a resolution of 864x864 pixels, were captured using a cellphone camera under controlled conditions to ensure consistency in lighting and background. The dataset is divided into training and validation sets, with 80% (2,768 images) allocated for training and 20% (720 images) for validation.

The dataset is arranged into two distinct folders for training and validation, allowing for efficient data management during the training process. This structure helps ensure that the model can generalize effectively, preventing overfitting while learning the core patterns of the hand gestures and maintaining strong performance on previously unseen validation data.

Data Preprocessing

Data preprocessing is a critical step in preparing the raw image data for the model and is vital in improving the performance of the CNN (Olisah et al., 2022). First, the images are subjected to data augmentation, a method employed to artificially increase the size of the dataset and introduce variations that help the model generalize better. Augmentation techniques applied to the images include rotation, zooming, flipping, shearing, and adjustments to brightness and contrast. These transformations simulate various real-world scenarios and ensure the model avoids overfitting to the training data, improving its ability to recognize hand gestures across different settings.

Figure 2 depicts the data preprocessing steps utilized in this study, beginning with uploading the collected hand gesture images to Google Drive and their subsequent access via Google Colaboratory. The preprocessing pipeline includes data augmentation performed employing the ImageDataGenerator class of the Keras library. This augmentation applies rotation, zoom, flipping, shearing, and brightness adjustments to increase the variety and robustness of the training data.

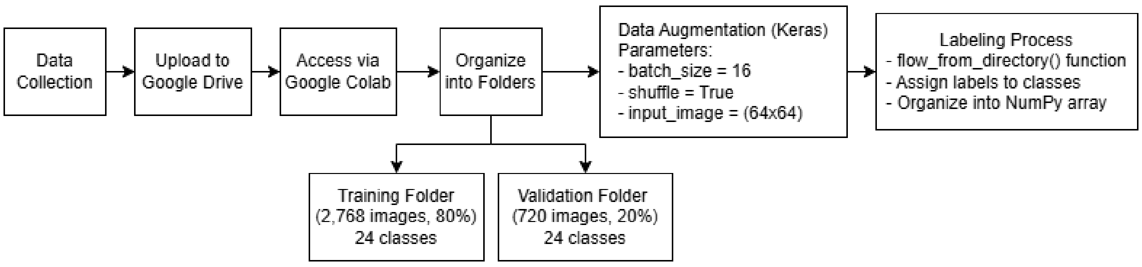
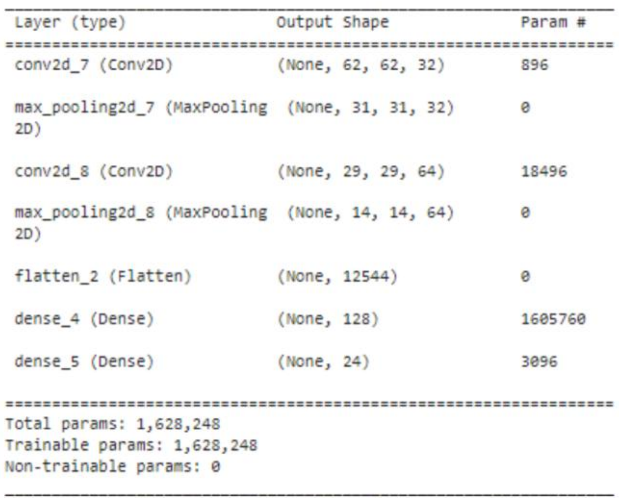

Data Collection
Upload to Google Drive
Access via Google Colab
Organize into Folders
Data Augmentation (Keras) Parameters:
- batch_size = 16
- shuffle = True
- input_image = (64x64)
Labeling Process
- flow_from_directory() function
- Assign labels to classes
- Organize into NumPy array
Training Folder (2,768 images, 80%) 24 classes
Validation Folder (720 images, 20%) 24 classes

Fig. 2. Data Preprocessing Workflow: Uploading, Augmentation, Labeling of SIBI Hand Gesture Images

Subsequently, the dataset is split into two sets—training and validation—with 80% of the images allocated for training and 20% for validation. The flow_from_directory() function in Keras effectively manages the data. This function automatically labels and organizes the images into NumPy arrays according to the specified directory structure. Each image is resized to 64x64 pixels and normalized to ensure uniformity in scale and intensity, enabling the model to focus on the most essential features of the hand gestures.

CNN Model Architecture

The CNN architecture designed for this study is specifically optimized for processing image data and classifying the different hand gestures in the SIBI sign language alphabet. The architecture consists of several layers that progressively extract and refine features from the input images, which helps distinguish subtle variations in the hand gestures. The model begins with two convolutional layers. The first convolutional layer (conv2d_7) uses 32 filters with a kernel size 3x3, applied to the input images. This layer captures low-level features, such as edges and textures, crucial for recognizing hand gestures. After the convolutional operation, the feature maps are processed by a max-pooling layer (max_pooling2d_7), which reduces the spatial dimensions of the feature maps, effectively decreasing computational complexity while retaining the essential features. The max-pooling layer uses a 2x2 pool size, halving the dimensions of the feature maps.


```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_7 (Conv2D)           (None, 62, 62, 32)        896

 max_pooling2d_7 (MaxPooling  (None, 31, 31, 32)       0
 2D)

 conv2d_8 (Conv2D)           (None, 29, 29, 64)        18496

 max_pooling2d_8 (MaxPooling  (None, 14, 14, 64)       0
 2D)

 flatten_2 (Flatten)         (None, 12544)             0

 dense_4 (Dense)             (None, 128)               1605760

 dense_5 (Dense)             (None, 24)                3096

=================================================================
Total params: 1,628,248
Trainable params: 1,628,248
Non-trainable params: 0
_________________________________________________________________
```

Fig. 3. Architecture of the CNN Model for SIBI Hand Gesture Classification

The second convolutional layer (conv2d_8) applies 64 filters, each with a kernel size of 3x3, to the output from the first max-pooling layer. This layer captures more complex features and patterns, enhancing the model's ability to recognize intricate hand gestures. Following this convolutional operation, another max-pooling layer (max_pooling2d_8) is applied further to reduce the spatial dimensions of the feature maps, ensuring that only the most essential features are retained. This pooling layer also uses a 2x2 pool size. After the convolutional and pooling layers, the resulting feature maps are flattened into a one-dimensional vector. This is done in the flatten layer (flatten_2), which transforms the 3D output from the last pooling layer into a 1D array of size 12,544. This flattened output is then passed to the fully connected layers for further processing.

The model includes a fully connected dense layer (dense_4) with 128 units. The activation function used for this layer is ReLU (Rectified Linear Unit), which introduces non-linearity into the model. ReLU allows the model to learn complex patterns in the data by applying a thresholding mechanism. After the fully connected layer, dropout regularization is used to prevent overfitting. Dropout randomly deactivates a fraction of neurons during training, forcing the model to learn more robust features that generalize better to new, unseen data.

The final layer of the model is another dense layer (dense_5) with 24 units, corresponding to the 24 distinct hand gesture classes (A-Y, excluding J and Z). This layer uses the SoftMax activation function, which converts the raw output of the network into a probability distribution, ensuring that the predicted values are normalized between 0 and 1, with the sum of all probabilities equal to 1. The class with the highest probability is considered the expected hand gesture.

The Adam optimizer is used for optimization, leveraging the advantages of both Adagrad and RMSProp. This optimizer dynamically adapts the learning rate during training, which helps the model converge faster and more efficiently. The learning rate is initially set to 0.001, a commonly used value for the Adam optimizer, though it can be adjusted during hyperparameter tuning for improved performance. The model is trained for 20 epochs, with 173 steps per epoch and a batch size 16. This choice of training parameters was made to balance model performance and computational efficiency. While the number of epochs and steps per epoch can significantly affect model accuracy, generally improving performance with higher values, this comes at the cost of increased computational resources. This study did not implement early stopping; however, 20 epochs were selected based on preliminary tests to ensure that the model could achieve satisfactory accuracy without overfitting. Future experiments may consider incorporating early stopping to optimize training duration further and prevent overfitting.

Hardware Comparison: CPU vs. GPU

To assess the effect of hardware on model performance, the CNN model is trained twice, once on a central processing unit (CPU) and once on a Graphics Processing Unit (GPU). The objective is to evaluate the differences in training efficiency, accuracy, and time across these platforms. The training is conducted on Google Collaboratory, a cloud-based platform offering access to powerful computing resources.

The training on the CPU is performed using an Intel Xeon processor with eight cores, running at a frequency of 2.00 GHz. While CPUs are versatile and capable of handling various computational tasks, they are generally slower than GPUs when training deep learning models, especially for tasks involving large datasets and complex matrix operations, as required in CNN-based models. The Xeon processor was selected for its performance in handling parallel tasks, although it remains less efficient than GPUs for deep learning applications.

In contrast, the training on the GPU utilizes an Nvidia Tesla T4 GPU with 16 GB of GDDR6 VRAM. The Tesla T4 is known for its excellent performance in deep learning tasks. Its parallel processing capabilities allow it to handle large-scale computations much more efficiently than CPUs. GPUs are specifically designed for tasks like image recognition, where they significantly reduce training time and can improve model accuracy. The Tesla T4 was chosen for its well-established ability to accelerate deep learning tasks, making it ideal for evaluating the hardware's impact on model performance. Both systems are equipped with 16 GB of RAM, and the GPU environment has access to a bandwidth of 300 GB/s, enabling handling large datasets and ensuring that memory speed does not become a bottleneck during training.

The model's performance on both hardware platforms is assessed based on three key metrics: accuracy, loss, and training time. Accuracy measures how often the model makes correct predictions, loss indicates how well the model fits the data, and training time records how long it takes to complete one epoch of training. Comparing these metrics provides insight into how hardware influences model performance, particularly regarding computational efficiency when training deep learning models for sign language recognition.

Evaluation and Analysis

After completing the training on both the CPU and GPU, the results are analyzed to identify significant differences in accuracy, loss, and training time. Statistical analyses will be conducted to evaluate whether the differences observed between the two platforms are substantial. This analysis will provide insights into the practical implications of using different hardware platforms for training CNN models, particularly in real-time SLR.

This study selects accuracy as the primary evaluation metric due to its straightforward interpretation and the focus on overall model performance. While precision, recall, and F1 score are essential for evaluating class imbalances in specific contexts, accuracy is deemed sufficient for this task as the dataset is relatively balanced, with an equal number of images per class. Since our primary objective is to assess the impact of hardware configurations on training efficiency and overall performance, accuracy provides a clear and direct comparison across platforms. Future studies may consider incorporating additional metrics like precision, recall, and F1 score, particularly when dealing with highly imbalanced datasets.

Training time was recorded from when the training process started until the completion of the final epoch. The total training time for each session—CPU and GPU—was measured in real time using the built-in timers of the training framework, ensuring accuracy and consistency in the reported times. The training process was completed for one full run on each hardware setup, and the time to complete the 20 epochs was captured, allowing for a direct comparison between the CPU and GPU performance.

The model's performance was evaluated at the end of each epoch on both the training and validation datasets to measure accuracy. Accuracy was calculated as the percentage of correct predictions over each dataset's total number of samples. To reduce variance and ensure reliability in the results, multiple training runs were conducted across both hardware setups. Each run's accuracy and training time were averaged to mitigate any variability due to random initialization or other factors. This approach helped to provide more stable and consistent performance metrics, ensuring that the comparison between the CPU and GPU training sessions accurately reflected the impact of the hardware on the model's performance.

The results of this study will enhance the broader understanding of how hardware influences the performance of machine learning models, particularly in environments with limited resources. This insight can inform future research and help guide practical decisions in choosing hardware for training and deploying machine learning models, especially for applications such as SLR.

4. Results and Discussions

This section highlights the main findings from the experiments to assess the performance of the CNN-based model for recognizing the SIBI alphabet. The training and testing phases were performed using two different hardware configurations: a CPU and a GPU. The primary goal of these experiments was to assess how the choice of hardware influences the training time, model performance, and its effectiveness in generalizing to unseen data (Nadir et al., 2024; Štaka et al., 2025). The results are presented in two main parts: the first focuses on the training and validation accuracy, comparing the performance across the two hardware setups, while the second provides an analysis of the test accuracy based on a separate dataset of unseen images (Mohammadi et al., 2023; Siek, 2023). A detailed discussion of the implications of these results follows, focusing on improving the model’s generalization and possible directions for future research.

Training and Validation Accuracy

The training was conducted on a dataset of 3,468 images divided into 16 batches for efficient processing. It was carried out in two sessions: a CPU and a GPU. This allowed for directly comparing how hardware influences the model's training time and performance.

In the first session, where training was performed using a CPU, as shown in Figure 4, the model completed the training in 1 hour, 39 minutes, and 55 seconds. The performance during this session was promising on the training set, with a precision of 99.13% and a relatively low loss of 0.24% during the first epoch (one complete cycle through the data). However, the validation data showed a noticeable discrepancy, with an accuracy of 96.7% and a significantly higher loss of 20.43%. The difference in performance between the training and validation data indicates that the model may have experienced overfitting, a common problem where the model performs well on the training data but fails to generalize to new, unseen data.

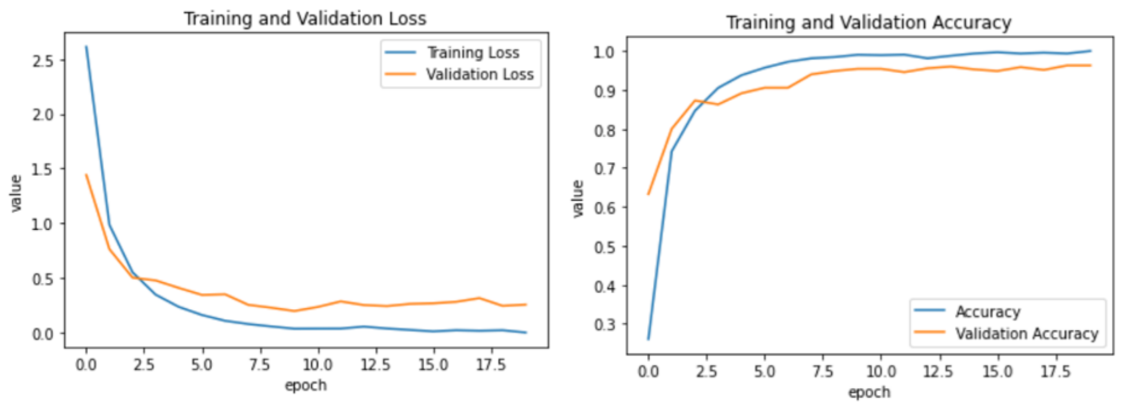


Fig. 4. Training Performance Metrics: Accuracy and Loss During CPU-Based Model Training

The second session was conducted using a GPU, as shown in Figure 5, significantly reducing the training time and completing the process in 1 hour, 18 minutes, and 30 seconds. Initially, the model performed poorly, with a training precision of 36.71% and a loss of 22.59% during the initial epoch. The validation data similarly showed poor performance, with an accuracy of just 14.87% and a loss of 58.00%. These early results suggest that the model's initial weights or training parameters were not well-optimized, and the model struggled to learn effectively at the start.

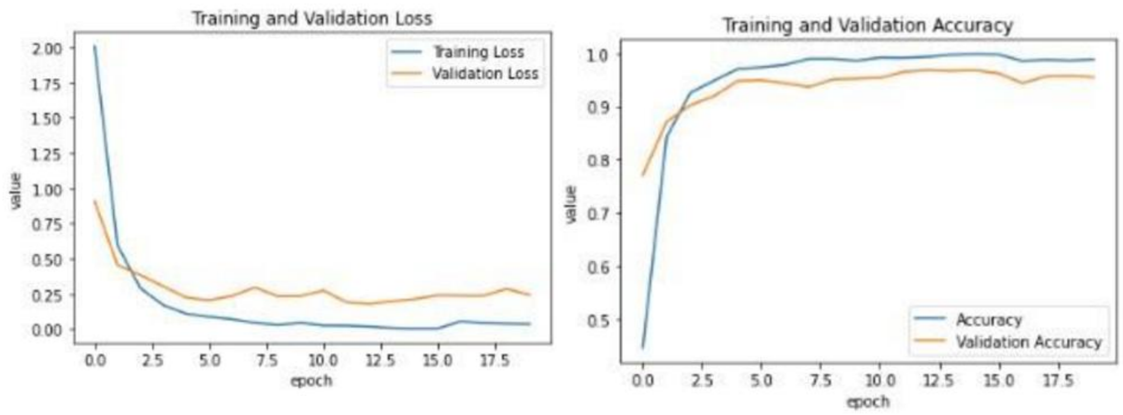


Fig. 5. Training Performance Metrics: Initial Accuracy and Loss During GPU-Based Model Training

However, as training progressed over 20 epochs, the model's performance improved significantly. At the end of the training, the model reached a training accuracy of 99.31% with a loss of 0.26%, showing it had effectively learned the training data. Even more notably, the validation accuracy increased to 96.7%, accompanied by a significant reduction in loss to 2.63%.

This improvement in validation results indicates that the GPU-accelerated training helped the model generalize more effectively, preventing the overfitting seen during the CPU-based session.

As presented in Table 2, the two training sessions demonstrate similar final validation accuracy, reaching approximately 96.7%. Training accuracy increased rapidly in both cases and approached near 100%, indicating strong model learning on the training set. Despite achieving similar peak validation accuracy, the learning behavior differs slightly. The CPU session shows a more stable validation curve, with a consistently narrow gap between training and validation accuracy. In contrast, the GPU session exhibits a slight fluctuation in validation accuracy after epoch 10 and a somewhat wider gap from the training accuracy, suggesting mild overfitting. These results indicate that while both models generalize well, the CPU session may provide more stable performance across epochs.

Table 2 - Comparison of Training and Validation Accuracy Across Two Sessions

Feature	CPU Session	GPU Session
Training Time	1 hour, 39 minutes, and 55 seconds	1 hour, 18 minutes, and 30 seconds
Training Accuracy	Rapid rise to ~100%	Similar, slightly smoother
Validation Accuracy	Peaks around 96.7%	Peaks around 96.7%, similar
Overfitting Signs	Mild, stable performance	More noticeable after epoch 10
Stability	More stable in later epochs	Slight dip in final epochs

The reduction in training time when using the GPU compared to the CPU is substantial, with the GPU session completing the training in 1 hour, 18 minutes, and 30 seconds, a 45.5% decrease from the CPU session's 1 hour, 39 minutes, and 55 seconds. This significant reduction in training time highlights the GPU's parallel processing capabilities' efficiency, enabling it to handle the computational demands of deep learning tasks more effectively than the CPU. The GPU's ability to accelerate training is particularly valuable for large datasets and complex models, allowing faster convergence without sacrificing performance. This efficiency is crucial for real-time applications, where rapid model training and deployment are often essential.

The training times and performance differences highlight the significant impact that hardware can have on the model's training process. While the CPU session took longer and showed signs of overfitting, the GPU session enabled faster convergence and improved generalization, ultimately leading to a stronger model. These results highlight the importance of choosing the proper hardware, particularly for deep learning models requiring significant computational resources.

Test Accuracy Model

To evaluate how well the model generalizes to new, unseen data, a separate testing phase was conducted using a dataset of 720 images, with 30 images per class from a total of 24 classes. The results of the testing, including accuracy, precision, and misclassifications, are summarized in Figure 6.

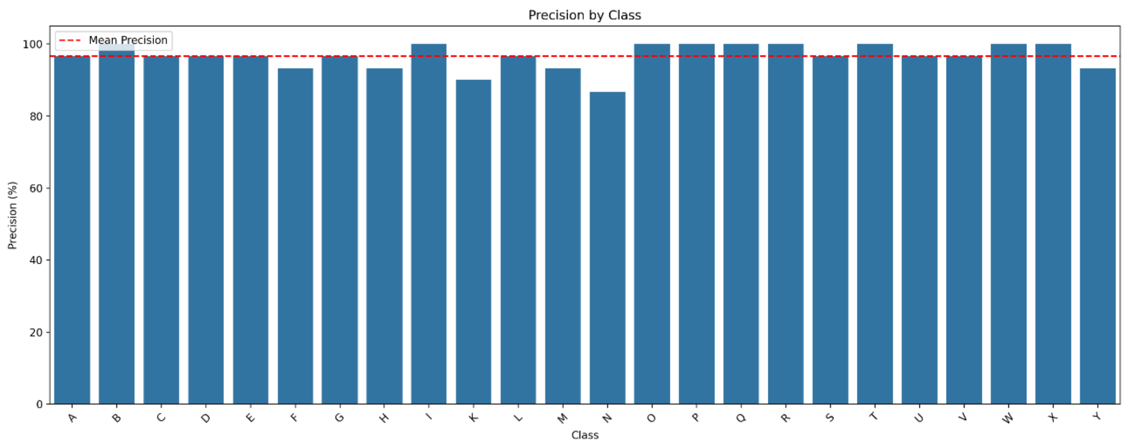


Fig. 6. Testing Phase Results: Accuracy and Precision Across SIBI Alphabet Classes

The model achieved varying levels of success across different classes, with the highest accuracy reaching 100% for several courses, including B, I, O, P, Q, R, T, W, and X. These results suggest that the model excelled at recognizing these particular characters. However, classes such as N (with an accuracy of 86.67%) and K (with an accuracy of 90.00%) showed slightly lower performance. The model may have struggled with these classes due to similar shapes with other characters or insufficient training data for these specific letters.

Precision, A crucial measure that indicates the model's capacity to minimize false positives, was generally high across most classes. The precision values ranged from 93.33% to 100%, with the lowest precision observed for classes K (90.00%) and N (86.67%). These slight variations in precision could be attributed to challenges in differentiating between visually similar characters, suggesting that data augmentation or improved class-specific feature extraction techniques might be beneficial for further refinement of the model.

The overall accuracy for the test set was 96.67%, with 696 correct predictions out of 720 total. This high accuracy demonstrates that the model can effectively classify the SIBI alphabet, although there is room for improvement in handling certain classes.

Misclassifications and Potential Improvements

The model's misclassifications were relatively few, but addressing these instances could improve its overall performance. For example, class N had the lowest accuracy (86.67%), which indicates that this letter might share visual similarities with others in the dataset. A deeper analysis of misclassified instances could provide insights into the specific features that confuse the model, allowing for targeted improvements such as adding more diverse examples of class N using techniques like transfer learning or fine-tuning these problematic classes.

Upon examining the confusion matrix in Figure 7, it is evident that the model demonstrated strong performance across most of the SIBI alphabet classes, with several letters, including B, I, O, P, Q, R, T, W, and X, achieving 100% accuracy. This indicates that the model is particularly effective at recognizing these specific gestures. However, certain classes, such as N and K, showed lower performance, with 86.67% and 90.00% accuracy, respectively. The misclassifications observed for these letters suggest that the model may have struggled to differentiate between visually similar characters. For instance, class N was frequently misclassified as M, and K was often confused with H, indicating that these classes share significant visual similarities. These results suggest that the model's performance could be improved by incorporating additional training data for these classes or by employing data augmentation techniques to enhance its ability to distinguish between such similar gestures. Thus, while the model overall shows promising results, targeted optimization is needed for these specific classes to achieve more robust generalization.

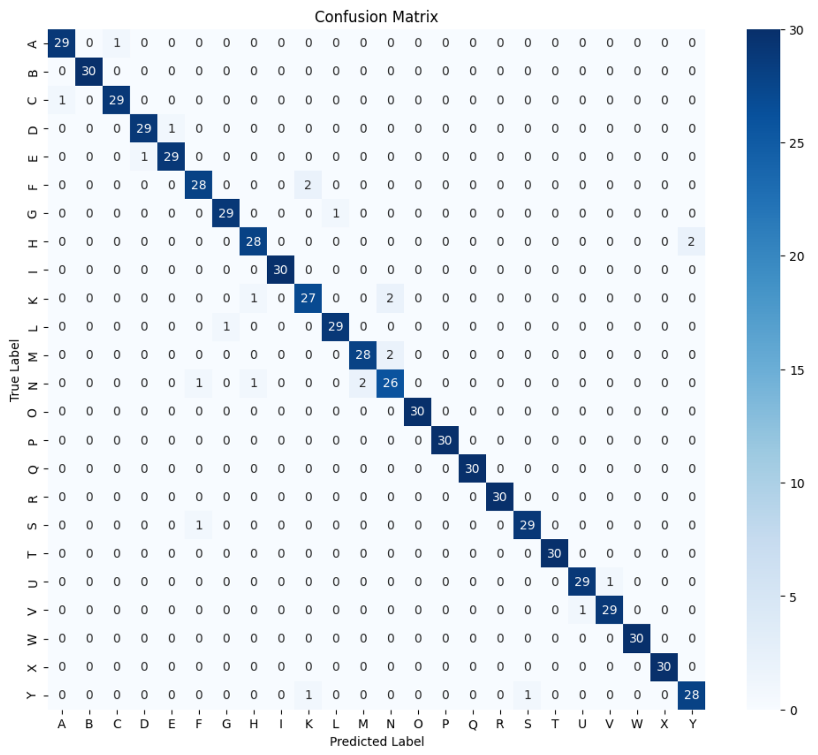


Fig. 7. Confusion Matrix of Model Predictions Across SIBI Alphabet Classes

Optimizing its architecture or incorporating regularization techniques such as dropout can improve the model's performance. Additionally, applying data augmentation would help the model learn more resilient features. These approaches can also assist in reducing overfitting, leading to better generalization.

Discussion of Model Performance

The results of this study demonstrate that the CNN-based model can achieve high levels of accuracy in recognizing the SIBI alphabet. The training sessions highlighted the importance of using the appropriate hardware to accelerate training and improve model generalization. While the CPU-based training session showed signs of overfitting, the GPU session allowed for faster training and better generalization to validation data. This highlights the critical role of hardware in deep learning tasks, especially regarding computational efficiency and model performance.

With a test accuracy of 96.67%, the model demonstrates strong generalization to new, unseen data, and the accuracy remained consistent at 96.7%, showing no significant change between the two hardware setups. However, there remains room for further improvement. The discrepancies in accuracy across different classes suggest that the model may benefit from more targeted optimization for specific characters, especially those with lower accuracy. Future work could focus on improving the model's performance for these challenging classes through techniques like data augmentation, fine-tuning, or leveraging more advanced network architectures such as ResNet or Inception.

The increased memory and the use of the GPU played a critical role in the performance gains observed during model training (Pacini et al., 2024; Y. Zhang et al., 2023). GPUs are designed to handle parallel processing tasks, which is particularly advantageous for deep learning models like CNNs, where operations can be performed simultaneously across multiple data points. The higher memory bandwidth of the GPU, especially with the Nvidia Tesla T4's 16 GB of GDDR6 VRAM, allows for handling larger batches and more complex computations without significant memory bottlenecks. As a result, the model can process more data in less time, leading to faster training times. Moreover, the GPU's ability to manage large amounts of data efficiently results in more stable and consistent performance, reducing the risk of overfitting and enabling the model to generalize better. This increased efficiency was reflected in the 45.5% reduction in training time compared to the CPU, demonstrating the value of higher memory and GPU usage in deep learning tasks.

These findings align with existing research highlighting the significant performance benefits of GPUs in deep learning. For example, studies have shown that GPUs, with their high computational power and large memory capacity, drastically reduce training time compared to traditional CPUs (Tasnim et al., 2024). Furthermore, research demonstrated that using GPUs accelerates the convergence of deep learning models, particularly in tasks involving large datasets, such as image recognition. Our results corroborate these findings by showing that, although the GPU's memory usage is higher, the benefits of reduced training time and improved convergence outweigh the additional memory requirements. The similarity in validation accuracy between the CPU and GPU sessions also supports the idea that GPUs do not necessarily compromise model performance but optimize the training process.

Despite the clear performance improvements with the GPU, there are several limitations to consider when using higher-end hardware (L. Li et al., 2024). One key drawback is the cost associated with powerful GPUs. High-performance GPUs, such as the Nvidia Tesla T4, are expensive, and their use may not be feasible in resource-constrained environments or for smaller-scale projects. Additionally, GPUs consume significantly more energy compared to CPUs. This higher energy consumption is critical, especially for mobile and embedded systems with limited battery life. The increased power requirements can lead to higher operational costs and potential overheating in constrained environments. Moreover, the applicability in real-time or mobile systems is limited, as the higher power consumption and cost of GPUs may not be justified for specific applications where latency and resource availability are critical factors (Ping et al., 2024; Rico, 2023). CPU-based solutions or more energy-efficient hardware may better balance performance and feasibility for real-time systems.

The findings from this study contribute to the growing body of research in hardware-aware deep learning, which emphasizes the importance of choosing the proper hardware to optimize training efficiency while maintaining model performance. This is especially relevant in resource-constrained training environments, where computational power, memory, and energy availability are limiting factors (Huang et al., 2023; Rajput et al., 2024). As deep learning models become more complex, efficient hardware solutions become increasingly important, particularly in mobile devices or embedded systems scenarios. Research on energy-efficient deep learning is gaining traction, with techniques such as model pruning, quantization, and low-power hardware accelerators being explored to reduce the reliance on high-performance GPUs (Saha et al., 2025; Zaitsev et al., 2025). The results of this study reinforce the importance of considering the trade-offs between hardware capabilities, energy consumption, and cost, particularly for deploying deep learning models in real-world applications with limited resources.

When considering hardware trade-offs, energy consumption is essential, particularly for mobile deployment or environments with limited resources (Salcedo-Navarro et al., 2025). While GPUs, such as the Nvidia Tesla T4, offer significant processing speed and model convergence advantages due to their parallel processing capabilities, they consume considerably more power than CPUs. This higher energy demand makes GPUs less ideal for mobile or embedded systems, where battery life and energy efficiency are critical considerations (Bryzgalov & Maeda, 2024; Choi et al., 2024). On the other hand, CPUs, though slower for deep learning tasks, tend to consume less power, making them more suitable for power-constrained environments. Therefore, while GPUs can significantly accelerate training and improve performance, their higher power consumption must be weighed against the benefits when deploying models in real-time applications, particularly on mobile or embedded platforms. Future research could focus on optimizing energy-efficient hardware solutions or exploring ways to balance the trade-off between power consumption and processing performance.

The study highlights the significant influence of hardware on the performance and efficiency of CNN models, offering valuable insights into the model's strengths and areas that require further improvement. The findings indicate that the model shows promise for real-world sign language recognition applications, with potential for further enhancement through focused data and architectural improvements.

5. Conclusion

This study investigates the use of CNN for recognizing the SIBI alphabet, emphasizing the often-overlooked yet critical role of computing hardware in model performance, particularly for resource-intensive tasks like sign language recognition (SLR). The results demonstrate that GPU-based training significantly outperforms CPU-based training, achieving a 45% reduction in training time while maintaining strong model generalization. This highlights the importance of selecting the proper hardware to optimize training efficiency and model effectiveness in deep learning applications.

The CNN model achieved a test accuracy of 96.67%, successfully recognizing most of the SIBI alphabet gestures. However, certain characters, such as N and K, showed lower accuracy, suggesting the need for further optimization. Future work could focus on strategies like data augmentation, class-specific fine-tuning, and exploring advanced architectures such as ResNet or Inception to improve performance on these challenging classes.

This research underscores the potential of CNN-based systems in bridging communication barriers for the deaf community, offering a promising approach for real-time SLR technologies. More importantly, it highlights the significant role of hardware in optimizing model performance. While GPUs provide considerable advantages in reducing training time and enhancing model performance, their cost and energy consumption may make them impractical for deployment on edge devices. Thus, selecting the proper hardware configuration is crucial when considering deployment in real-world settings, particularly for applications requiring resource-constrained devices.

Future research could investigate energy efficiency, particularly for real-time SLR applications on mobile or embedded devices, where power constraints are critical. Using transfer learning or lightweight CNN models may also help reduce computational demands and facilitate deployment on edge devices without sacrificing performance. Expanding dataset diversity will also improve model generalization, especially for underperforming classes, leading to more robust gesture recognition. These efforts will help optimize the model for scalable and efficient real-world applications, including educational tools for sign language learning, assistive wearables, and other assistive technologies to improve communication for the deaf community. By addressing these challenges, the model could be integrated into practical applications that enhance accessibility and inclusion, ultimately fostering better communication and understanding across diverse user groups.

Acknowledgment

The author would like to sincerely thank Prof. Muchlas, Rector of Universitas Ahmad Dahlan (UAD), for his unwavering support and encouragement throughout this research. Special thanks go to Prof. Abdul Fadlil, Head of the Informatics Doctoral Department at UAD, and Prof. Imam Riadi, Secretary of the Informatics Doctoral Department, for their invaluable guidance, advice, and constructive feedback during the writing process. Additionally, heartfelt appreciation is extended to Prof. Tole Sutikno, Lecturer at the Informatics Doctoral Department, for his thoughtful insights and thorough evaluation, which significantly enhanced the quality of this work. Their combined efforts have played a crucial role in completing this research.

References

Abdalla, A., Alsereidi, A., Alyammahi, N., Qehaizel, F. B., Ignatious, H. A., & El-Sayed, H. (2023). An Innovative Arabic Text Sign Language Translator. *Procedia Computer Science, 224,* 425–430. https://doi.org/10.1016/j.procs.2023.09.059

Abdallah, M. S., Hemayed, E., Abdalla, M. S., & Hemayed, E. E. (2013). *Dynamic Hand Gesture Recognition of Arabic Sign Language using Hand Motion Trajectory Features Dynamic Hand Gesture Recognition of Arabic Sign Language using Hand Motion Trajectory Features*. https://www.researchgate.net/publication/258172682

Abdallah, M. S., Samaan, G. H., Wadie, A. R., Makhmudov, F., & Cho, Y. I. (2023). Light-Weight Deep Learning Techniques with Advanced Processing for Real-Time Hand Gesture Recognition. *Sensors*, *23*(1). https://doi.org/10.3390/s23010002

Abdullah, B. A. Al, Amoudi, G. A., & Alghamdi, H. S. (2024). Advancements in Sign Language Recognition: A Comprehensive Review and Future Prospects. *IEEE Access*, *12*, 128871–128895. https://doi.org/10.1109/ACCESS.2024.3457692

Al Moustafa, A. M. J., Shafry, M., Rahim, M., Khattab, M. M., Zeki, A. M., Matter, S. S., Soliman, A. M., & Ahmed, A. M. (2024). Arabic Sign Language Recognition Systems: A Systematic Review. *Indian Journal of Computer Science and Engineering (IJCSE)*, *15*. https://doi.org/10.21817/indjcse/2024/v15i1/241501008

Alsharif, B., Altaher, A. S., Altaher, A., Ilyas, M., & Alalwany, E. (2023a). Deep Learning Technology to Recognize American Sign Language Alphabet. *Sensors*, *23*(18). https://doi.org/10.3390/s23187970

Alsharif, B., Altaher, A. S., Altaher, A., Ilyas, M., & Alalwany, E. (2023b). Deep Learning Technology to Recognize American Sign Language Alphabet. *Sensors*, *23*(18). https://doi.org/10.3390/s23187970

Arya, N., Soni, T., Pattanaik, M., & Sharma, G. K. (2022). Energy efficient logarithmic-based approximate divider for ASIC and FPGA-based implementations. *Microprocessors and Microsystems*, *90*, 104498. https://doi.org/https://doi.org/10.1016/j.micpro.2022.104498

Ben Atitallah, B., Hu, Z., Bouchaala, D., Hussain, M. A., Ismail, A., Derbel, N., & Kanoun, O. (2022). Hand Sign Recognition System Based on EIT Imaging and Robust CNN Classification. *IEEE Sensors Journal*, *22*(2), 1729–1737. https://doi.org/10.1109/JSEN.2021.3130982

Bryzgalov, P., & Maeda, T. (2024). Using Benchmarking and Regression Models for Predicting CNN Training Time on a GPU. *Proceedings of the 4th Workshop on Performance EngineeRing, Modelling, Analysis, and VisualizatiOn STrategy*, 8–15. https://doi.org/10.1145/3660317.3660323

Choi, J., Lee, H. J., Sohn, K., Yu, H. S., & Rhee, C. E. (2024). Accelerating CNN Training with Concurrent Execution of GPU and Processing-In-Memory. *IEEE Access*. https://doi.org/10.1109/ACCESS.2024.3488004

Darmawan, I. D. M. B. A., Linawati, L., Sukadarmika, G., Wirastuti, N. M. A. E. D., Pulungan, R., Mulyanto, & Hariyanti, N. K. D. (2023). Advancing Total Communication in SIBI: A Proposed Conceptual Framework for Sign Language Translation. *2023 International Conference on Smart-Green Technology in Electrical and Information Systems (ICSGTEIS)*, 23–28. https://doi.org/10.1109/ICSGTEIS60500.2023.10424020

Duarte, A., Palaskar, S., Ventura, L., Ghadiyaram, D., Dehaan, K., Metze, F., Torres, J., & Giro-I-Nieto, X. (2021). *How2Sign: A Large-scale Multimodal Dataset for Continuous American Sign Language*. http://how2sign.github.io/

Fadlilah, U., Mahamad, A. K., & Handaga, B. (2021a). The Development of Android for Indonesian Sign Language Using Tensorflow Lite and CNN: An Initial Study. *Journal of Physics: Conference Series*, *1858*(1). https://doi.org/10.1088/1742-6596/1858/1/012085

Fadlilah, U., Mahamad, A. K., & Handaga, B. (2021b). The Development of Android for Indonesian Sign Language Using Tensorflow Lite and CNN: An Initial Study. *Journal of Physics: Conference Series*, *1858*(1). https://doi.org/10.1088/1742-6596/1858/1/012085

Fadlilah, U., Prasetyo, R. A. R., Mahamad, A. K., Handaga, B., Saon, S., & Sudarmilah, E. (2022). Modelling of basic Indonesian Sign Language translator based on Raspberry Pi technology. *Scientific and Technical Journal of Information Technologies, Mechanics and Optics*, *22*(3), 574–584. https://doi.org/10.17586/2226-1494-2022-22-3-574-584

Fadlilah, U., Wismoyohadi, D., Mahamad, A. K., & Handaga, B. (2019). Bisindo information system as potential daily sign language learning. *AIP Conference Proceedings*, *2114*. https://doi.org/10.1063/1.5112492

Ferraz, O., Subramaniyan, S., Chinthala, R., Andrade, J., Cavallaro, J. R., Nandy, S. K., Silva, V., Zhang, X., Purnaprajna, M., & Falcao, G. (2022). A Survey on High-Throughput Non-Binary LDPC Decoders: ASIC, FPGA, and GPU Architectures. *IEEE Communications Surveys & Tutorials*, *24*(1), 524–556. https://doi.org/10.1109/COMST.2021.3126127

Handayani, A. N., Akbar, M. I., Ar-Rosyid, H., Ilham, M., Asmara, R. A., & Fukuda, O. (2022). Design of SIBI Sign Language Recognition Using Artificial Neural Network Backpropagation. *2022 2nd International Conference on Intelligent Cybernetics Technology & Applications (ICICyTA)*, 192–197. https://doi.org/10.1109/ICICyTA57421.2022.10038205

Hashi, A. O., Hashim, S. Z. M., & Asamah, A. B. (2024). A Systematic Review of Hand Gesture Recognition: An Update From 2018 to 2024. *IEEE Access*. https://doi.org/10.1109/ACCESS.2024.3421992

Hekmat, A., Ali, M., Abbas, H. H., & Shahadi, I. (2022). *Sign Language Recognition and Hand Gestures Review*. *02*(04). https://kjes.uokerbala.edu.iq/

Hu, Y., Liu, Y., & Liu, Z. (2022). A Survey on Convolutional Neural Network Accelerators: GPU, FPGA and ASIC. *2022 14th International Conference on Computer Research and Development (ICCRD)*, 100–107. https://doi.org/10.1109/ICCRD54409.2022.9730377

Huang, H., Li, Y., & Zhou, X. (2023). Accelerating Point Clouds Classification in Dynamic Graph CNN with GPU Tensor Core. *Proceedings of the International Conference on Parallel and Distributed Systems - ICPADS*, 1725–1732. https://doi.org/10.1109/ICPADS60453.2023.00240

Hussin, S. K., Mohamed, O., Mohamed, ; Mustafa, Ahmed, E., & Mahmoud, O. (2021). Real-time Arabic sign language translator Using media pipe and LSTM. *PLOMS Journal of Artificial Intelligence (PLOMS AI)*, 1–13. https://plomscience.com/journals/index.php/PLOMSAI/index

Irvanizam, I., Horatius, I., & Sofyan, H. (2023). Applying Artificial Neural Network Based on Backpropagation Method for Indonesian Sign Language Recognition. *International Journal of Computing and Digital Systems*, *14*(1), 975–985. https://doi.org/10.12785/ijcds/140176

Kwenda, C., Gwetu, M., & Fonou-Dombeu, J. V. (2023). Ontology with Deep Learning for Forest Image Classification. *Applied Sciences (Switzerland)*, *13*(8). https://doi.org/10.3390/app13085060

Latif, G., Mohammad, N., Alghazo, J., AlKhalaf, R., & AlKhalaf, R. (2019). ArASL: Arabic Alphabets Sign Language Dataset. *Data in Brief*, *23*. https://doi.org/10.1016/j.dib.2019.103777

Li, H., Wang, Z., Yue, X., Wang, W., Tomiyama, H., & Meng, L. (2023). An architecture-level analysis on deep learning models for low-impact computations. *Artificial Intelligence Review*, *56*(3), 1971–2010. https://doi.org/10.1007/s10462-022-10221-5

Li, L., Li, Y., & Tan, H. (2024). DeepDecompose: A Distributed inference framework for CNN on GPU-equipped Edge Clusters. *Proceedings of the 2024 5th International Conference on Computing, Networks and Internet of Things*, 540–544. https://doi.org/10.1145/3670105.3670200

Lu, C., Kozakai, M., & Jing, L. (2023). Sign Language Recognition with Multimodal Sensors and Deep Learning Methods. *Electronics (Switzerland)*, *12*(23). https://doi.org/10.3390/electronics12234827

Luong, S. (2023). *Video Sign Language Recognition using Pose Extraction and Deep Learning Models* [San Jose State University]. https://doi.org/10.31979/etd.jm4c-myd4

Mohammadi, M., Chandarana, P., Seekings, J., Hendrix, S., & Zand, R. (2022). Static hand gesture recognition for American sign language using neuromorphic hardware. *Neuromorphic Computing and Engineering*, *2*(4). https://doi.org/10.1088/2634-4386/ac94f3

Mohammadi, M., Smith, H., Khan, L., & Zand, R. (2023). Facial Expression Recognition at the Edge: CPU vs GPU vs VPU vs TPU. *Proceedings of the Great Lakes Symposium on VLSI 2023*, 243–248. https://doi.org/10.1145/3583781.3590245

Muis, A., Sunardi, S., & Yudhana, A. (2024). CNN-based Approach for Enhancing Brain Tumor Image Classification Accuracy. *International Journal of Engineering, Transactions B: Applications*, *37*(5), 984–996. https://doi.org/10.5829/ije.2024.37.05b.15

Nadir, M. N., Siraj Rathore, M., Hayat, A., & Mansoor, J. A. (2024). CPU vs. GPU: Performance comparison of OpenCL Applications on a Heterogeneous Architecture. *Journal of Computing & Biomedical Informatics*, *07*(02). https://doi.org/10.56979/702/2024

Nguyen, P. T., Nguyen, T. H., Hoang, N. X. N., Phan, H. T. B., Vu, H. S. H., & Huynh, H. N. (2023). Exploring MediaPipe optimization strategies for real-time sign language recognition. *CTU Journal of Innovation and Sustainable Development*, *15*(ISDS), 142–152. https://doi.org/10.22144/ctujoisd.2023.045

Olisah, C. C., Smith, L., & Smith, M. (2022). Diabetes mellitus prediction and diagnosis from a data preprocessing and machine learning perspective. *Computer Methods and Programs in Biomedicine*, *220*, 106773. https://doi.org/https://doi.org/10.1016/j.cmpb.2022.106773

Pacini, F., Pacini, T., Lai, G., Zocco, A. M., & Fanucci, L. (2024). Design and Evaluation of CPU-, GPU-, and FPGA-Based Deployment of a CNN for Motor Imagery Classification in Brain-Computer Interfaces. *Electronics (Switzerland)*, *13*(9). https://doi.org/10.3390/electronics13091646

Ping, Y., Jiang, H., Liu, X., Zhao, Z., Zhou, Z., & Chen, X. (2024). Latency-Based Inter-Operator Scheduling for CNN Inference Acceleration on GPU. *IEEE Transactions on Services Computing*, *17*(1), 277–290. https://doi.org/10.1109/TSC.2023.3345952

Pitonak, R., Mucha, J., Dobis, L., Javorka, M., & Marusin, M. (2022). CloudSatNet-1: FPGA-Based Hardware-Accelerated Quantized CNN for Satellite On-Board Cloud Coverage Classification. *Remote Sensing*, *14*(13). https://doi.org/10.3390/rs14133180

Rajput, S. H., Mukherji, P., Avachat, S., Chitnis, A., Deodhar, N., & Godse, P. (2024). Comparative Analysis of Efficient Implementation of CNN on CPU and GPU for Image Processing and Implementation of BasicCNN Operation in Verilog. *International Journal of Microsystems and IoT*, *2*(2), 538–547. https://doi.org/10.5281/zenodo.10792152

Rakhmadi, A., & Ariyanto, R. (2021). Measurement Motoric System of Cerebral Palsy Disability using Gross Motor Function Measure (GMFM). *Khazanah Informatika*, *7*(1), 32–37.

Rakhmadi, A., Yudhana, A., & Sunardi, S. (2024). Virtual Reality and Augmented Reality in Sign Language Recognition: A Review of Current Approaches. *International Journal of Informatics and Computation (IJICOM)*, *6*(2). https://doi.org/10.35842/ijicom

Rasch, M. J., Mackin, C., Le Gallo, M., Chen, A., Fasoli, A., Odermatt, F., Li, N., Nandakumar, S. R., Narayanan, P., Tsai, H., Burr, G. W., Sebastian, A., & Narayanan, V. (2023). Hardware-aware training for large-scale and diverse deep learning inference workloads using in-memory computing-based accelerators. *Nature Communications*, *14*(1). https://doi.org/10.1038/s41467-023-40770-4

Rawal, V., Prajapati, P., & Darji, A. (2023). Hardware implementation of 1D-CNN architecture for ECG arrhythmia classification. *Biomedical Signal Processing and Control*, *85*, 104865. https://doi.org/https://doi.org/10.1016/j.bspc.2023.104865

Rico, M.-U.-K. (2023). *Performance Analysis of CNN Model for Image Classification with Intel OpenVINO on CPU and GPU* [Thesis]. University of Windsor.

Safitri, M., Yuniarno, E. M., & Rachmadi, R. F. (2024). Indonesian Sign Language (SIBI) Recognition and Extraction Using Convolutional Neural Networks - Symmetric Deletion Spelling Correction. *2024 International Seminar on Intelligent Technology and Its Applications (ISITIA)*, 220–225. https://doi.org/10.1109/ISITIA63062.2024.10667714

Saha, A., Rahman, M., & Wu, F. (2025). Benchmarking CPU vs. GPU on LSTM Model Using Groundwater Dataset. *SoutheastCon 2025*, 1318–1319. https://doi.org/10.1109/SoutheastCon56624.2025.10971639

Salcedo-Navarro, A., Gutiérrez-Aguado, J., & Garcia-Pineda, M. (2025). UHD Video Encoding in CPU Versus GPU: Quality and Performance Trade-Offs. *IEEE Access*, *13*, 55115–55129. https://doi.org/10.1109/ACCESS.2025.3553634

Schulder, M., Bigeard, S., Hanke, T., & Kopf, M. (2023). The Sign Language Interchange Format: Harmonising Sign Language Datasets For Computational Processing. *2023 IEEE International Conference on Acoustics, Speech, and Signal Processing Workshops (ICASSPW)*, 1–5. https://doi.org/10.1109/ICASSPW59220.2023.10193022

Siek, M. (2023). Benchmarking CPU vs. GPU performance in building predictive LSTM deep learning models. *AIP Conference Proceedings*, *2510*(1), 030017. https://doi.org/10.1063/5.0128638

Štaka, Z., Mišić, M., & Tomašević, M. (2025). CPU vs. GPU: Performance Evaluation of Classical Machine and Deep Learning Algorithms. *2025 24th International Symposium INFOTEH-JAHORINA (INFOTEH)*, 1–6. https://doi.org/10.1109/INFOTEH64129.2025.10959248

Subandi, R., . H., & Yudhana, A. (2024). Pneumonia Medical Image Classification Using Convolution Neural Network Model AlexNet and GoogleNet. *International Journal of Computing and Digital Systems*, *16*(1), 1675–1684. https://doi.org/10.12785/ijcds/1601124

Subburaj, S., & Murugavalli, S. (2022). Survey on sign language recognition in context of vision-based and deep learning. *Measurement: Sensors*, *23*. https://doi.org/10.1016/j.measen.2022.100385

Subramanian, B., Olimov, B., Naik, S. M., Kim, S., Park, K. H., & Kim, J. (2022). An integrated mediapipe-optimized GRU model for Indian sign language recognition. *Scientific Reports*, *12*(1). https://doi.org/10.1038/s41598-022-15998-7

Suharjito, Thiracitta, N., & Gunawan, H. (2021). SIBI Sign Language Recognition Using Convolutional Neural Network Combined with Transfer Learning and non-trainable Parameters. *Procedia Computer Science*, *179*, 72–80. https://doi.org/https://doi.org/10.1016/j.procs.2020.12.011

Sultan, A., Makram, W., Kayed, M., & Ali, A. A. (2022). Sign language identification and recognition: A comparative study. In *Open Computer Science* (Vol. 12, Issue 1, pp. 191–210). Walter de Gruyter GmbH. https://doi.org/10.1515/comp-2022-0240

Svendsen, B., & Kadry, S. (2023). Comparative Analysis of Image Classification Models for Norwegian Sign Language Recognition. *Technologies*, *11*(4). https://doi.org/10.3390/technologies11040099

Tang, X., Chang, X., Chen, N., Ni, Y. (MaoMao), LC, R. A. Y., & Tong, X. (2023). Community-Driven Information Accessibility: Online Sign Language Content Creation within d/Deaf

Communities. *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*. https://doi.org/10.1145/3544548.3581286

Tarek, H., Aly, H., Eisa, S., & Abul-Soud, M. (2022). Optimized Deep Learning Algorithms for Tomato Leaf Disease Detection with Hardware Deployment. *Electronics (Switzerland), 11*(1). https://doi.org/10.3390/electronics11010140

Tasnim, T., Rahman, M., & Wu, F. (2024). A Comparative Analysis of CPU and GPU-Based Cloud Platforms for CNN Binary Classification. *IARIA Congress 2024: The 2024 IARIA Annual Congress on Frontiers in Science, Technology, Services, and Applications*. https://www.thinkmind.org

Waluyo, W. N., R. Rizal Isnanto, & Adian Fatchur Rochim. (2023). Comparison of Mycobacterium Tuberculosis Image Detection Accuracy Using CNN and Combination CNN-KNN. *Jurnal RESTI (Rekayasa Sistem Dan Teknologi Informasi)*, *7*(1), 80–87. https://doi.org/10.29207/resti.v7i1.4626

Wijaya, F., Dahendra, L., Purwanto, E. S., & Ario, M. K. (2024). Quantitative analysis of sign language translation using artificial neural network model. *Procedia Computer Science*, *245*, 998–1009. https://doi.org/https://doi.org/10.1016/j.procs.2024.10.328

Wu, X., Feng, Y., Xu, H., Lin, Z., Chen, T., Li, S., Qiu, S., Liu, Q., Ma, Y., & Zhang, S. (2023). CTransCNN: Combining transformer and CNN in multilabel medical image classification. *Knowledge-Based Systems, 281*. https://doi.org/10.1016/j.knosys.2023.111030

Yang, D., Li, J., Hao, G., Chen, Q., Wei, X., Dai, Z., Hou, Z., Zhang, L., & Li, X. (2024). Hardware accelerator for high accuracy sign language recognition with residual network based on FPGAs. *IEICE Electronics Express*, *21*(4). https://doi.org/10.1587/elex.21.20230579

Zaitsev, D. A., Ajima, Y., Bartlett, J. F. C., & Kumar, A. (2025). 3D multicore CPU vs GPU on sparse patterns of Sleptsov net virtual machine. *International Journal of Parallel, Emergent and Distributed Systems*. https://doi.org/10.1080/17445760.2025.2490148

Zhang, Y., Pandey, D., Wu, D., Kundu, T., Li, R., & Shu, T. (2023). Accuracy-Constrained Efficiency Optimization and GPU Profiling of CNN Inference for Detecting Drainage Crossing Locations. *Proceedings of the SC ’23 Workshops of the International Conference on High Performance Computing, Network, Storage, and Analysis*, 1780–1788. https://doi.org/10.1145/3624062.3624260

Zhang, Z., Gao, J., Dhaliwal, R. S., & Li, T. J.-J. (2023). VISAR: A Human-AI Argumentative Writing Assistant with Visual Programming and Rapid Draft Prototyping. *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*. https://doi.org/10.1145/3586183.3606800