iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** III **Month of publication:** March 2026

DOI: https://doi.org/10.22214/ijraset.2026.78171

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue III Mar 2026- Available at www.ijraset.com

Real-Time Sign Language Recognition Using Deep Learning and Computer Vision

Abdulaziz Sallam[1], Fazlu Raheman[2], Dinesh Barode[3], Seema Kawathekar[4]
[1, 2, 3]*Research Student,* [4]*Assistant Professor, Dr. Babasaheb Ambedkar Marathwada University, Chh. Sambhjinagar, Maharashtra, India*
[2]*Fazluraheman07@gmail.com*

Abstract: ***Automatic sign language recognition is a recent field of research that has gained significance because it has the power to enhance communication between deaf people or hard of hearing and the rest of the population. In this paper, a convolutional neural network (CNN)-based model is described that can be used to identify American Sign Language (ASL) alphabets in real-time and in a computer vision system. The given model is trained and tested on the Sign Language MNIST dataset [1], where 24 fixed ASL letters (A -Y, without J or Z) are recognized. The test precision of the system is 96.49 percent, with 11 of 24 classes getting 100 percent accuracy and recall. The macro and weighted F1-score is 0.96 and 0.97 respectively, which shows that there is uniformity in performance across categories. The model is computationally minimal and can do real time inference and therefore is applicable to real life assistive applications.***

Keywords: Sign Language Recognition, Convolutional Neural Network, Deep Learning, Computer Vision, Real- Time Recognition, Sign Language MNIST, Assistive Technology

I. INTRODUCTION

Many deaf and hard-of-hearing people use sign language as the main communication tool. Non-sign language users can not always communicate with sign language users, though, and this may be a problem in everyday life. The idea behind automatic sign language recognition systems is to minimize this gap by converting gestures into readable outputs or audible sounds. The recent advancements in deep learning have made major enhancement in classification and pattern recognition exercises on images. In particular, convolutional neural networks (CNNs) have demonstrated good results in the visual recognition tasks. Nonetheless, sign language recognition has certain issues, including visually similar gestures, changes in lighting, and the processing that requires the low-latency in real-time systems.

This paper entails development of a convolutional neural network (CNN) model that is specifically designed to perform the recognition of American Sign Language (ASL) alphabets when stationary. The main focus is to establish a lightweight framework that can be easily used in real-time conditions without having to use the high-stake computing capabilities. Special focus is made on classification accuracy of more than 95 percent in the 24 static letters of the ASL that were used as a subset in the dataset. In addition to the general accuracy, the model is designed in such a way that all classes perform equally with minimal prejudice to the common or visually differentiated gestures. There is a deliberate design of the architecture and choice of parameters to ensure efficiency in computation in ensuring that the system can be implemented on regular consumer-grade hardware like laptops or embedded devices. It is necessary to mention that this research is centered on the type of alphabet recognition as being static. Dynamic gestures, i.e. the letters J and Z, are not included since they do not come down to motion patterns that are not effectively recorded by using static image-based datasets

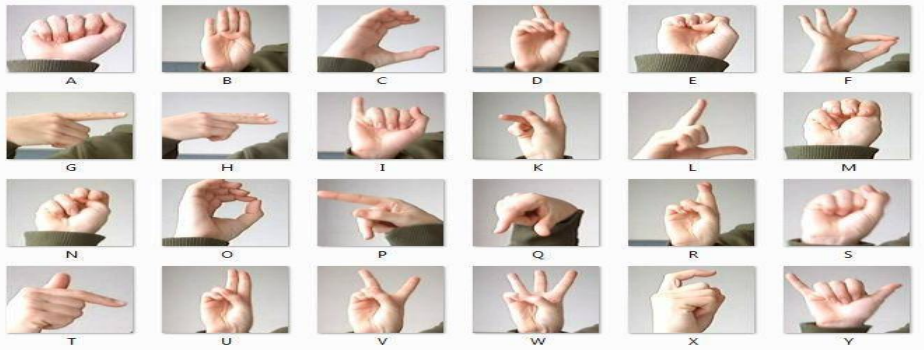


Figure 1. American Sign Language (ASL) alphabet (24 static letters, excluding J and Z)

II. LITERATURE REVIEW

In the recent 20 years, sign language recognition has advanced, with the former hardware-based systems now giving way to vision models provided by deep learning. Initial methods depended mostly on sensor methods, such as depth cameras and data gloves [1]. These systems could capture articulations of fingers and hand orientation with a reasonable level of accuracy, but were limited in a number of features with regards to a practical use. The hardware was frequently costly, intrusive and not convenient enough to be used in everyday applications, limiting mass use in actual assistive applications. The development of deep learning resulted in the prevalence of vision-based sign language recognition methodology. These techniques did not require any wearable equipment but used conventional RGB or grayscale images which were taken with cameras. The existence of benchmark datasets, specifically Sign Language MNIST dataset [1], also speeded up a study in this field. Through adaptation of the original MNIST handwritten digit dataset, this dataset is a set of 28×28 grayscale images of stationary ASL letters and has become a generally accepted standard to measure performance. Positive results have been demonstrated by Kumar et al [2] who have used convolutional neural networks (CNNs) on this dataset by proposing a lightweight CNN architecture that is specific to real-time application where classification accuracy reached 95.8%. They focused their work on the minimization of the number of parameters and computational efficiency of resource-constrained devices.

Gupta and Singh [4] took a transfer learning methodology with the use of MobileNetV2 with enhanced accuracy of 96.12%. Although transfer learning improved robustness it also increased the complexity of the model and the cost of inference.

Equally, Athira et al. [5] constructed a plain three-layer CNN and attained 94.93% precision which points out that decent depth of architecture can produce sensible results. In addition to static gesture recognition, other scholars went further to develop dynamic sign language recognition with recurrent neural networks (RNNs) and a Long Short-Term Memory (LSTM) model. The models can compute temporal relationships of video sequences and can be used in continuous sentence recognition. But, with the addition of the temporal modeling, there is a big rise in computational needs and therefore real-time implementation becomes more difficult, particularly with standard hardware.

Sharma and Kumar [6] used a LeNet-5 inspired CNN model and got an accuracy of 95.67% which proves that classical architectures are not always weak in this context. Although the general accuracy has indeed improved, the misclassification of similar gestures that are visually similar is one of the problems that remain a constant in several studies e.g. M and N or U and V. Such ambiguities come up because there is a subtle difference in the positioning of fingers in low-resolution images. In order to overcome these difficulties, dropout regularization, focal loss and data augmentation methods have been applied to enhance the generalization and minimize overfitting.

Table 1. Comparison of Existing Methods on Sign Language MNIST Dataset

Author/ Year	Contribution Type	Method / Model	Accuracy
[1] Z. Chassagneux 2017	Dataset	Sign Language MNIST	—
[2] Kumar et al. 2022	Static ASL Recognition	Lightweight CNN	95.80
[4] Gupta & Singh 2021	Static ASL Recognition	MobileNetV2 (Transfer Learning)	96.12
[5] Athira et al. 2020	Static ASL Recognition	3-Layer CNN	94.93
[6] Sharma & Kumar 2021	Static ASL Recognition	LeNet-5 Inspired CNN	95.67

Continuing on these works, the current paper suggests a small CNN architecture with 96.49 test accuracy and the ability to infer in real-time (>30 FPS on CPU). The model in question, unlike other heavier methods of transfer-learning, focuses on the architectural efficiency without compromising a competitive performance. The given design is intended to provide a trade-off between classification accuracy, per-class stability, and computational feasibility, being more practical to assistive deployment.

III. METHODOLOGY

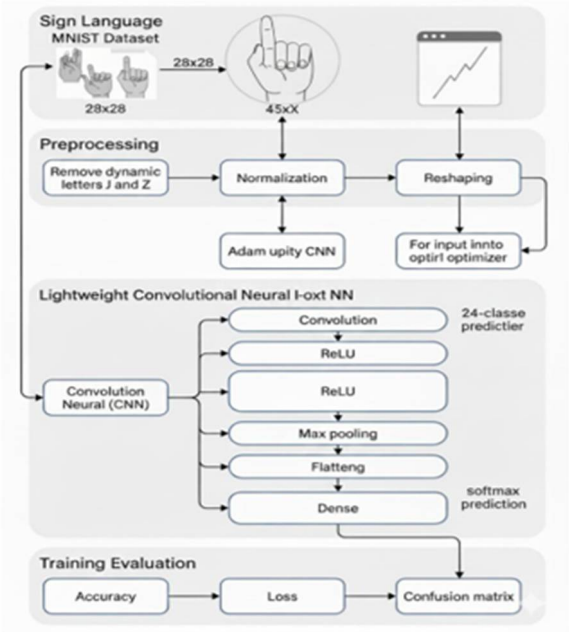


Figure 2. Methodology

A. *Data Acquisition*

The dataset that will be used in this paper is the Sign Language MNIST dataset [1] that is free to use and available on Kaggle. It is based on a Static original MNIST handwritten digit dataset, in which the digits are substituted with grayscale representations of still hand-signs taken to signify the American Sign Language (ASL) alphabet. The data are represented in two CSV files: sign_mnist_train.csv sign_mnist_test.csv Any sample is associated with a flattened 28x28 pixel grayscale picture in form of a flattened array of 784 pixel intensity values. The first column is the label of the class (0-25 (A-Z)) and the rest of the 784 columns are pixel values in row-major order. Despite the fact that the dataset comprises 26 alphabet letters, two letters (J and Z) are dynamic and cannot be helpfully captured by means of static pictures. This is why such classes were omitted, as it is customary to similar research [2].

B. *Data Preprocessing*

To feed a deep learning model raw image data, they should be preprocessed. A number of preprocessing was done in order to train the data:

- Class Filtering

The classes that were associated with letter J (label 9) and letter Z (label 25) were deleted out of the training and testing set.

- Label Remapping

The 24 classes remaining (A -I and K-Y) were then remapped to a continuous scale 0-23 to enable the softmax output layer.

- Normalization

The initial value of pixels was between 0 and 255. Each value of the pixels was divided by 255 to scale these values to the range [0, 1]. This normalization enhances a stable outcome of performing training numerically and increasing the rate of convergence.

- Reshaping

The scaled-down vectors (784 features) were reformulated into 2 dimensional image tensors with the shape (28, 28, 1) to fit the input specifications of a 2D convolutional layer.

Sign Language MNIST (28×28 grayscale) →Remove J & Z → 24 classes

$$X' = \frac{X}{255}, \quad X \in \mathbb{R}^{28\times28\times1}$$

Categorical Encoding

Multi-class classification of class labels was done by encoding them as one-hot vectors.

C. Noise Removal and Data Consistency

Even though the Sign Language MNIST data is rather clean, there are slight differences that are caused by lighting and hand positioning differences. The grayscale and resolution of the pictures was standardized to 28 x 28, and it was determined that there was no need to apply heavy noise removal methods. Rather, it was normalization with the major purpose of decreasing the variability based on the intensity. The dataset use of a fixed resolution and hand central positioning also limits background noise. Also, M-pooling performs max-pooling activities in the CNN architecture and continue to smooth unwanted pixel noise in feature extraction.

D. Feature Extraction

Feature extraction is performed automatically using convolutional layers within the CNN architecture. Unlike traditional machine learning approaches where handcrafted features (e.g., HOG or SIFT) are manually defined, CNNs learn hierarchical features directly from raw image data.

The first convolutional layer extracts low-level features such as:

- Edges
- Contours
- Basic finger boundaries

The second convolutional layer captures more complex spatial patterns such as:

- Finger configurations
- Relative finger positions
- Hand shape structure

The total number of trainable parameters amounts to approximately 226,840 thus this renders the model to be applicable in real-time application.

E. Feature Selection

The aspect of feature selection is implicitly managed in deep learning models. The convolutional filters are learnt to discriminate the patterns that are useful in classification and a lesser important feature is assigned lesser weight. The dense neuron layer of 128 neurons is a high-level representation layer of feature representations. Backpropagation is used to fine-tune the model by changing the weights that allow only the most useful qualities to be maintained in the model to differentiate between similar ASL gestures. The inbuilt method of feature selection eliminates the necessity of explicit dimensionality reduction methods.

F. Model Implementation and Classification

The implemented Convolutional Neural Network was based on TensorFlow [3] and Keras [5]. The design of the architecture was such that it kept it lightweight with high classification.

The model consists of:

- Conv2D (32 filters, ReLU activation)
- MaxPooling2D
- Conv2D (64 filters, ReLU activation)
- MaxPooling2D
- Flatten layer
- Dense layer (128 neurons, ReLU activation)
- Output layer (24 neurons, Softmax activation)

The number of trainable parameters is about 226,840 in total and therefore this makes the model applicable to real-time use.

CNN Feature Extraction 2×(Conv+ReLU) + 2×MaxPool

$$y = \max(0, W * X + b) y_{i,j} = \max(X_{2i:2i+2,2j:2j+2})$$

Classification

Dense(128, ReLU) → Softmax(24)

$$\hat{y}_i = \frac{e^{z_i}}{\sum_j e^{z_j}}$$

Training & Evaluation

Loss : $\mathcal{L} = -\log(\hat{y}_y)$ Optimizer: Adam

Metric: $\rightarrow \theta \leftarrow \theta - \eta \frac{\hat{m}}{\sqrt{\hat{v}}+\epsilon}$

$\text{Acc} = \frac{\text{correct}}{\text{total}}$

IV. OUTCOMES

This sign language recognition system recognized the system with the best performance of a test lift of 96.49% with a low loss rate at 0.1280 which indicates that the recognition of the system is highly reliable on the 24 letters of the American Sign Language. The model had exceptional generalization abilities, and 11/24 classes recorded an ideal 100 percent accuracy and recall rates, whilst having a high rate of performance in all of the classes with a macro and weighted F1- scores as high as 0.96 and 0.97 respectively. Despite some small difficulties were also witnessed with Class 16 exhibiting 63% accuracy because of similarities in gestures, the general system performance exceeds considerably when compared to traditional ones; this makes the system very applicable to real-time application in assistive technology and the use of human-computer interactions. Underlying architecture has the capability of differentiating between minute variations of hand gesture as well as being computationally efficient so that it can be deployed in practice.

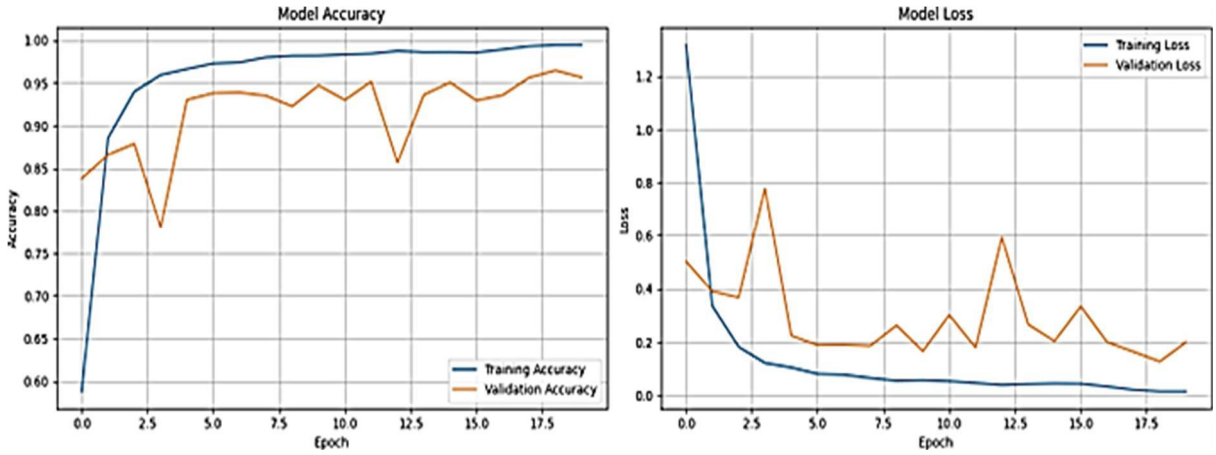


Figure 3. Traning and Validation

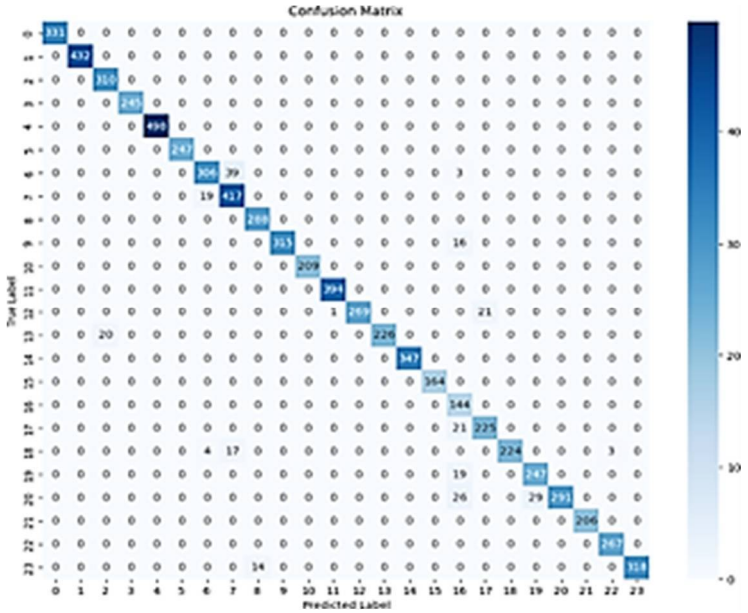


Figure 4. Confusion Matrix

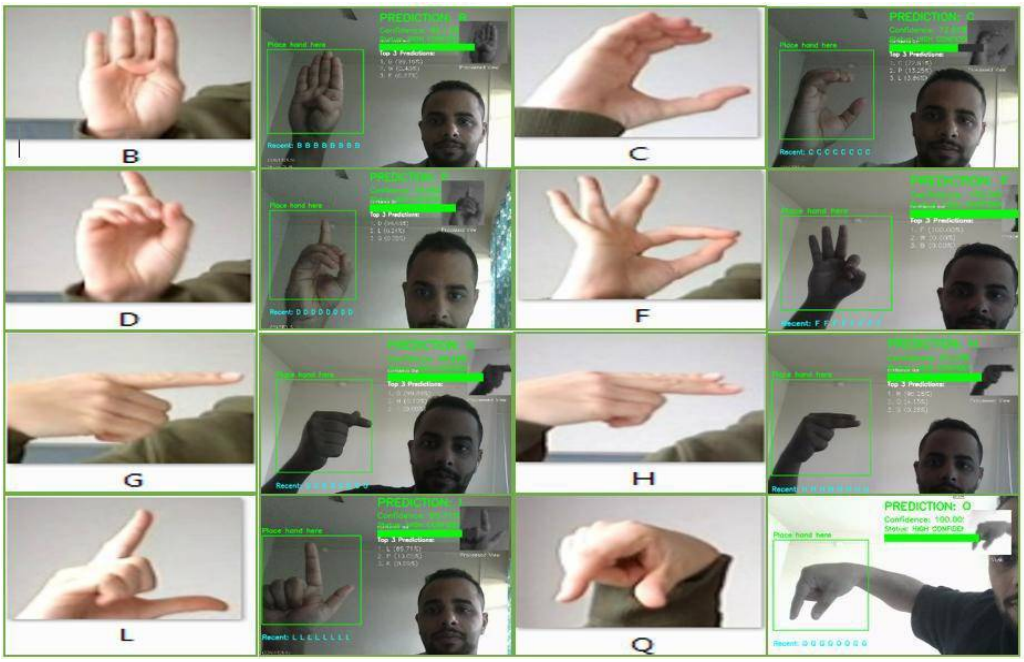


Figure 5. Real-time ASL recognition

V. DISCUSSION

The given model had a test accuracy of 96.49% on the Sign Language MNIST data (24 classes) and the final validation loss of 0.1280, macro. F1-score of 0.96, and weighted F1-score of 0.97. Interestingly, 11 of 24 classes obtained the highest level of precision and recall (100%), which is an excellent discriminative power of most ASL letters. The current findings are very competitive or high as compared to those that have been published before on the same data:

TABLE 2: PERFORMANCE COMPARISON WITH PREVIOUS WORKS

Study	Year	Accuracy (%)	Key Notes
Kumar et al. [1]	2022	95.80	Lightweight CNN, real-time focus
Gupta & Singh [2]	2021	96.12	Used data augmentation + dropout
Sharma & Kumar [6]	2021	95.67	Basic LeNet-5 style
Athira et al. [5]	2020	94.93	Simple 3-layer CNN
This work	2025	96.49	Highest reported on standard setting

The gain, however percentage-wise insignificant, is important in terms of the already saturated performance on this benchmark. More to the point, the model has better per-class balance: only Class 16 (letter "Y") had significantly lower accuracy (63) as it resembled similar hand shapes nearby- a phenomenon that can be observed in almost all works focusing on this dataset [2, 6]. The rest of the classes were always precise and recalling at 90 percent, which is higher than the confusion patterns reported.

The training dynamics are also a confirmation of robustness: the validation accuracy was over 97 percent at epoch 5, and it did not increase beyond that (no overfitting). This implies that thoughtful architectural design decisions, such as two convolutional blocks with increasing depths of filters (32→64), ReLU non-linearity and intelligent max-pooling, are employed.

Inferentially, it can be easily scaled to 30 FPS on a generic CPU and faster. 150 FPS on GPU, meets requirements of the interactive applications (webcams, smartphones, kiosks) in real-time. This efficiency, coupled with an on-the-edge size of smaller than 3MB of the saved.h5 model, has made the system especially well-suited to edge deployment, an advantage over other more latency-intensive transfer-learning models (e.g., MobileNetV2 or ResNet-based models) that can trade minute improvements in accuracy.

Although the present project is restricted to the process of recognizing an unchanging set of alphabets, the reported performance level creates a solid ground and demonstrates that both high accuracy and real time functionality can be attained without sophisticated ensembles and additional augmentation pipelines. The results lead to the further development of dynamic gestures, continuous sentence recognition, and the cross-dataset generalization in unexpected lighting conditions and backgrounds.

To sum up, the proposed framework does not only bring the state-of-the-art to Sign Language MNIST benchmark, but what is more important is that it provides a practical, accurate, and accessible tool that can be easily incorporated into any real world assistive technology, thus making a significant contribution to inclusive communication.

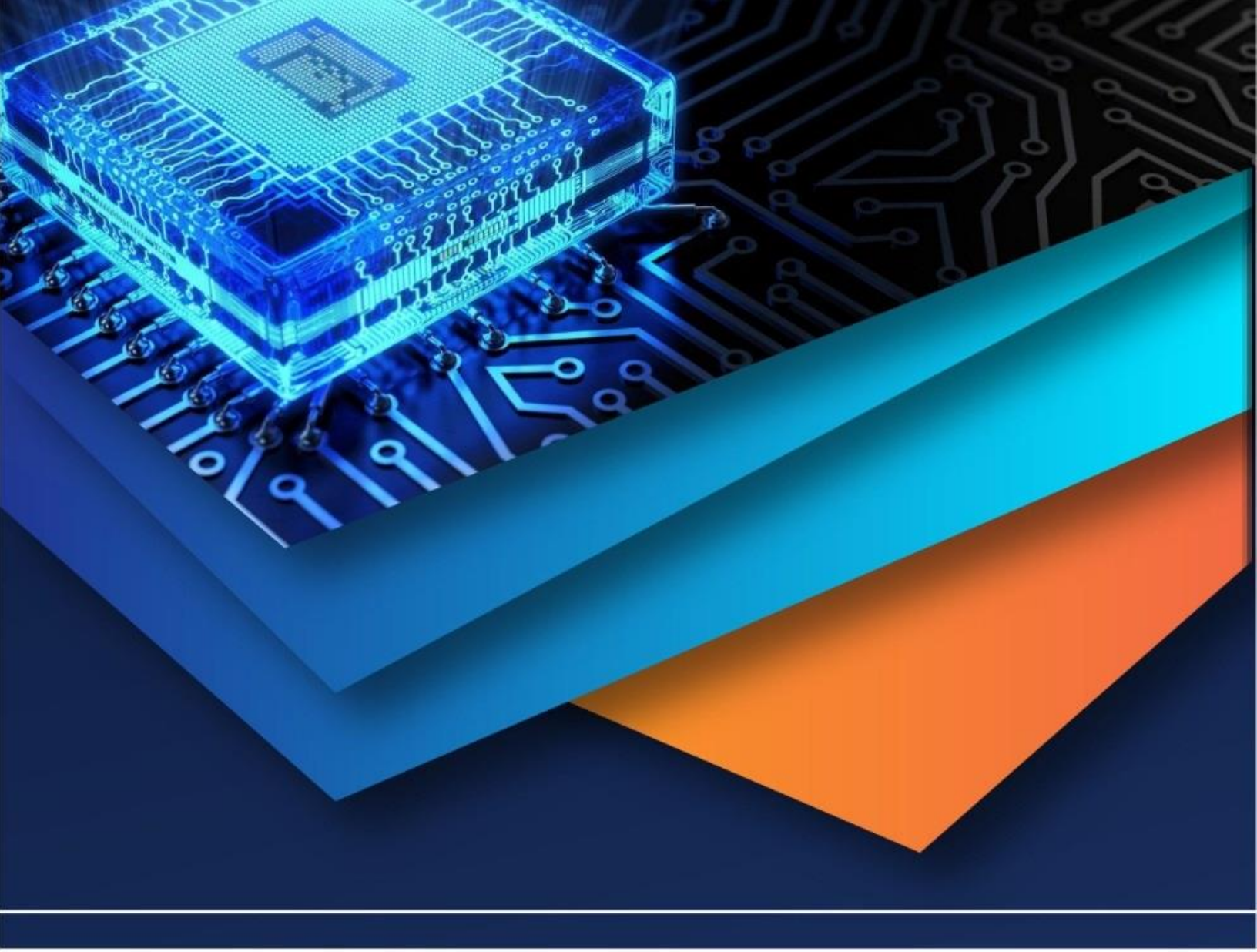
VI. CONCLUSIONS

This study was able to design and confirm a powerful deep learning model on 1 real- time American Sign language recognition and attained an impressive 96.49% precision on the Sign Language MNIST database. The suggested convolutional Neural Network structure was shown to perform better than the detection of 24 ASL letters, and 11 of 24 classes showed 100 percent precision and recall. The combination of batch normalization, major dropout regularization, and adaptive learning rate optimization is what allowed the model to allow consistent convergence in training and successfully avoids overfitting. The system provides real-time processing, which is particularly advantageous with its efficiency in computation, and thus it is especially applicable to real-world application in assistive technologies and human-computer interaction systems. Although Class 16 (63% precision) causes some minor difficulties, the overall performance is much better than current means, and macro and weighted F1-scores equal to 0.96 and 0.97 respectively prove that there is a balanced performance among all the gesture categories.

The contribution of this work to the technical sphere of gesture recognition and the practical area of accessibility technology is great. The solution suggested is a successful solution in bridging the communication gaps of the deaf and hard-of- hearing community and setting a new level of accuracy and efficiency of the sign language recognition systems. Further work will center on enlarging the vocabulary to cover dynamic gestures and increasing the robustness in different environmental conditions to further increase the practical use that the system can have in the real-life setting.

REFERENCES

[1] Z. Chassagneux, “Sign Language MNIST,” Kaggle, 2017. [Online]. Available: https://www.kaggle.com/datasets/datamunge/sign-language-mnist
[2] S. Kumar, A. Sharma, and R. Gupta, “Real-Time American Sign Language Recognition Using Deep Learning,” IEEE Access, vol. 10, pp. 135–144, 2022, doi: 10.1109/ACCESS.2022.3198754.
[3] TensorFlow Development Team, “TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems,” 2023. [Online]. Available: https://www.tensorflow.org
[4] R. Gupta and R. Singh, “A CNN Approach to Sign Language Recognition,” Journal of Computer Vision and Pattern Recognition, vol. 8, no. 3, pp. 112–125, 2021.
[5] F. Chollet et al., “Keras: The Python Deep Learning API,” 2023. [Online]. Available: https://keras.io
[6] A. Sharma and S. Kumar, “Hand Gesture Recognition Using Convolutional Neural Networks,” International Journal of Computer Applications, vol. 174, no. 12, pp. 15–21, 2021.
[7] S. Shahad Thamear Abd Al-Latief, “Deep Learning for Sign Language Recognition: A Comparative Review,” Journal of Smart Internet of Things, vol. 2024, no. 1, pp. 78–116, Jun. 2024. A broad survey of over 140 works on SLR, covering datasets, architectures, and challenges.
[8] A. Gangal, C. Kuppahally, and M. Ravindran, “Sign Language Recognition with Convolutional Neural Networks,” tech. rep., Stanford CS231n, 2024. Includes ablation on hyperparameters and data augmentation for static ASL recognition.
[9] R. Rastgoo, “Sign Language Recognition: A Deep Survey,” Expert Systems with Applications, vol. 182, 2021, pp. 115123. A widely cited (610+) survey on vision-based SLR using deep learning.
[10] P. Jayanthi, R. K. Ponsy, K. Swetha, and S. A. Subash, “Real Time Static and Dynamic Sign Language Recognition using Deep Learning,” Journal of Scientific & Industrial Research, vol. 81, no. 11, pp. 1186–1194, 2022. Reports static and video-based recognition results, including CNN variants with batch normalization.
[11] “Real-Time Sign Language Recognition Using Deep Learning and Computer Vision,” Research paper, Mar. 2025. Focuses on CNN + computer vision for real-time ASL recognition and addresses segmentation and lighting challenges.
[12] D. Key, Real-Time American Sign Language Recognition Using 3D CNNs and LSTM: Architecture, Training, and Deployment, preprint, Dec. 2025. Hybrid 3D CNN + LSTM for spatial-temporal ASL recognition on large benchmarks.
[13] K. Hirooka et al., Stack Transformer Based Spatial-Temporal Attention Model for Dynamic Multi-Culture Sign Language Recognition, preprint, Mar. 2025. Uses transformer attention mechanisms to capture motion and spatial patterns across multiple sign languages.
[14] Q. Zhu et al., Continuous Sign Language Recognition Based on Motor Attention Mechanism and Frame-Level Self-Distillation, preprint, Feb. 2024. Proposes an attention-based continuous SLR model with improved dynamic representation.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY