ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24MAY500

Recognizing Sign Language using Machine Learning and Deep Learning Models

Sohan Maurya[1]; Sparsh Doshi[2]; Harsh Jaiswar[3]; Sahil Karale[4]; Sneha Burnase[5]; Dr. Poonam. N. Sonar[6]
Department of Electronics and Telecommunication, Rajiv Gandhi Institute of Technology, Mumbai, India

Abstract:- **Individuals with hearing impairments communicate mostly through sign language. Our goal was to create an American Sign Language recognition dataset and utilize it in a neural network-based machine learning model that can interpret hand gestures and positions into natural language. In our study, we incorporated the SVM, CNN and Resnet-18 models to enhance predictability when interpreting ASL signs through this new dataset, which includes provisions such as lighting and distance limitations. Our research also features comparison results between all the other models implemented under invariant conditions versus those using our proposed CNN model. As demonstrated by its high levels of precision at 95.10% despite changes encountered during testing procedures like varying data sets or scene configurations where losses are minimal (0.545), there exists great potential for future applications in image recognition systems requiring deep learning techniques. Furthermore, these advancements may lead to significant improvements within various fields related explicitly to speech-language therapy sessions designed specifically around helping people overcome challenges associated with deafness while building bridges towards improved social integration opportunities.**

***Keywords*:-** *Image Recognition, Image Classification, Feature Extraction, Deep Learning, Convolutional Neural Network (CNN), Sign Language Translation, American Sign Language (ASL), Real-Time Recognition.*

I. INTRODUCTION

Sign language is a type of communication used by deaf and hard-of-hearing people that relies on visual cues. While it features its own unique grammar, vocabulary, and syntax, the majority of people in the world are not fluent in sign language, which can make interaction between members who use different forms difficult at times. The advancement in machine learning technology offers a potential solution to this challenge. Machine learning involves teaching computers how to learn from experience without needing explicit programming or instruction throughout the process. This includes training computer algorithms using extensive datasets so they can recognize recurring patterns within their environment while making predictions based upon such past input. Therefore, sign language recognition systems utilizing these advances offer an opportunity for bridging gaps amongst community members with differing abilities to communicate effectively when face-to-face interactions occur. These programs aim towards interpreting any sign language gesture accurately and translating it into either written or spoken language as required, thereby promoting smoother integration among those involved, irrespective of varying individual preferences regarding method preference during personal communications scenarios going forward.

This study investigates the use of a CNN model to improve sign language recognition. The goal is to break down communication barriers between people with hearing impairments and the rest of society by developing an efficient, trustworthy system capable of quickly identifying and understanding sign language motions. By training on vast amounts of American Sign Language (ASL) data, our proposed CNN will gain insights into various subtle shifts within sign characteristics for increased precision when recognizing differences among numerous signs.

II. RELATED WORKS

Sign language identification is an important field of research because it facilitates communication for people with hearing difficulties. Various deep learning-based approaches have been investigated over time to improve sign language recognition systems. This literature analysis seeks to consolidate present research findings and highlight prospective future research paths in the field of sign language recognition with Convolutional Neural Networks (CNNs).

Koller, Zargaran, Ney, and Bowden (2016) delivered a hybrid CNN-HMM version for non-stop signal language popularity [4], which combines the discriminative talents modern day CNNs with the series modelling competencies modern day Hidden Markov models (HMMs). They tested that their stop-to-give up embedding improved the performance on three challenging benchmark continuous signal language reputation duties, reaching relative upgrades among 15% and 38% and up to 13.3% absolute. This has a look at sheds mild at the capacity cutting-edge hybrid CNN-HMM fashions for enhancing the overall accuracy state-of-the-art signal language recognition systems.

Koller, Zargaran, Ney, and Bowden (2018) [5] conducted additional research into the usage of hybrid CNN-HMMs for robust statistical continuous sign language recognition. Although data from this work are not available, it is clear that the study expands on the investigation of CNN-HMM models in sign language identification,

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24MAY500

highlighting the necessity for strong and accurate recognition systems.

Furthermore, Wadhawan and Kumar (2020) discussed the application of 3D-CNNs to train spatio-temporal features from raw video data for sign language recognition [1]. The authors used spatial attention in the network to focus on areas of interest during feature extraction, and temporal attention to choose meaningful motions for categorization. This approach demonstrates how attention mechanisms can improve the overall accuracy of sign language recognition by allowing the network to focus on essential spatiotemporal information.

Huang, Zhou, Li, and Li (2019) also investigated attention-based 3D-CNNs for large-vocabulary sign language recognition [8]. Although no specific research outcomes were presented, the attention-based method emphasizes the necessity of focusing on significant features in sign language recognition, which contributes to the investigation of advanced CNN architectures for this job.

Barbhuiya, Karsh, and Jain (2020) proposed a CNN-based feature extraction and classification method for sign language recognition [7]. While specific results were not available, the study adds to our understanding of CNNs' potential for feature extraction and classification in sign language recognition systems.

Furthermore, Masood, Srivastava, Thuwal, and Ahmad (2018) studied real-time sign language gesture detection with CNN and Recurrent Neural Networks (RNNs) [2]. Their research focuses on detecting gestures from video sequences, demonstrating the power of merging CNNs and RNNs for real-time sign language recognition.

Additionally, Katoch, Singh, and Tiwary (2022) investigated the use of CNNs for American Sign Language identification, combining Speeded Up Robust Features (SURF) with Support Vector Machines (SVM) and CNN [6]. Their work adds to our understanding of CNNs' applicability in recognizing various sign languages.

Finally, Rastgoo, Kiani, and Escalera (2020) underlined the use of Long Short-Term Memory (LSTM) models for isolated hand sign language recognition [3]. Their method entailed linking LSTM to the fully linked layer of a CNN, revealing the potential of combining RNNs and CNNs for sequence learning tasks in sign language recognition.

In conclusion, this literature review explores various CNN-based techniques for sign language recognition and suggests areas for future research to improve overall accuracy and robustness. Despite significant progress in sign language recognition using CNN-based approaches, there are still research gaps and areas for improvement. One key challenge is real-time recognition, as many existing studies focus only on offline scenarios. However, seamless communication requires accurate real-time applications, especially when interacting dynamically with individuals who have hearing impairments. Moreover, recognizing variations in gestures that arise due to the diversity and complexity of different sign languages poses a considerable obstacle that warrants further investigation. Furthermore, integrating multimodal approaches such as incorporating additional physiological cues could enhance the overall accuracy and robustness of current systems significantly towards large-vocabulary scenarios common among people with varying levels of deafness or cultural backgrounds. Investigating various techniques while considering contextual factors would offer insights into generalizing models across diverse user populations beyond their native sign-language habitat. In summary, future research must prioritize addressing these knowledge gaps if we want to realize more effective inclusive solutions for individuals living with hearing disabilities through advanced sign language recognition technology.

III. PROPOSED METHODOLOGY

A. Dataset Description

We used two widely available datasets and one custom dataset to train the model. The American Sign Language Dataset, published on Kaggle.com, consists of 2515 JPEG files totaling 32.46 MB in size. It contains graphics of numbers 0 through 9, as well as all alphabets from A to Z, making it a comprehensive resource for ASL recognition and interpretation. The Indian sign language dataset contains 42k JPEG files, totaling 80mb in size. Similar to the ASL dataset, it contains graphics representing digits 0 to 9 as well as all alphabets from A to Z. These datasets are publicly available to the public, making them useful for study, education, and the development of sign language-related applications and technology. The "American Sign Language Dataset" is meticulously crafted, containing images captured in the ".JPEG" format. Spanning a considerable size of 320 MB, the dataset is rich with detail, consisting of 3600 files carefully organized to encompass a wide array of sign language gestures. It covers numeric hand signs from 0 to 9, as well as the complete set of alphabets in American Sign Language, offering a comprehensive resource for sign language recognition and related studies.

B. Pre-Processing

Pre-processing involves three stages. The first is resizing which refers to adjusting the image size, according to predetermined architecture requirements. Second, the scaling functionality changes the pixel range from 0-255 to 0-1, improving data setup and speeding up processing time. Lastly, normalization occurs where we applied various transformations like random resizing or cropping using PyTorch's 'transforms' module on training and validation datasets for optimal results.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24MAY500

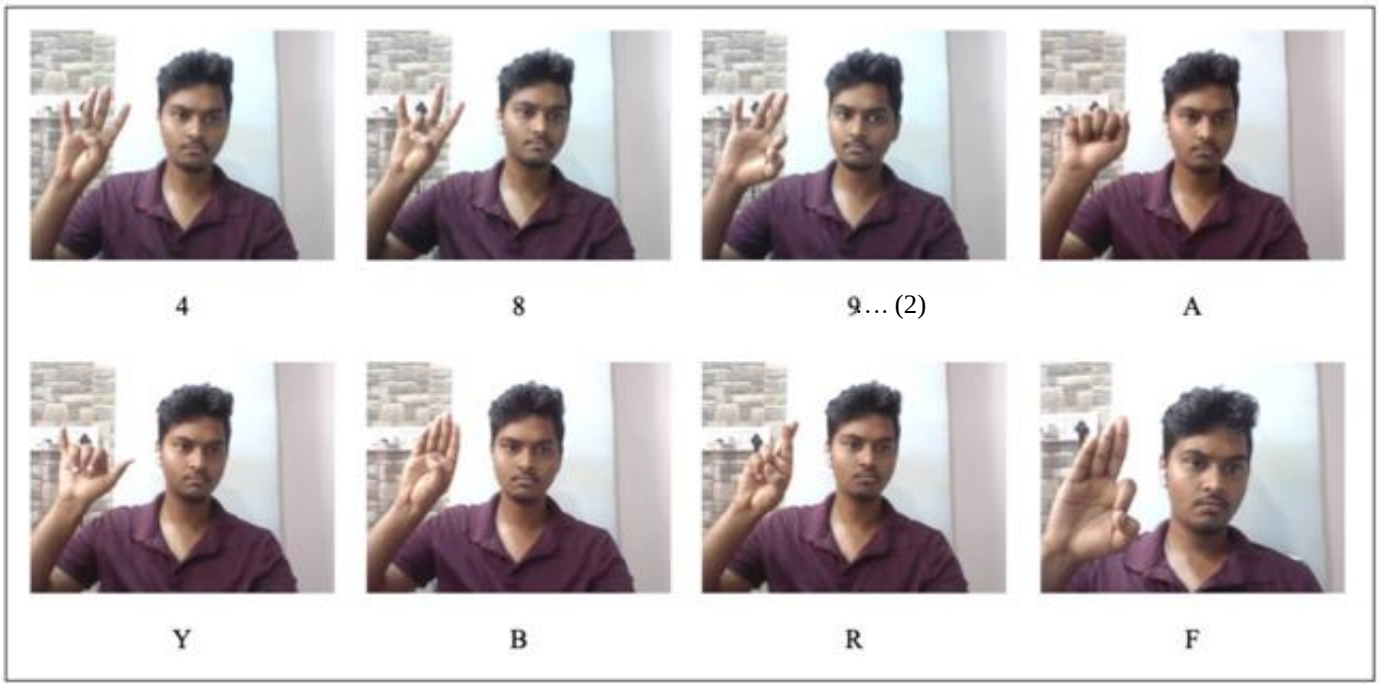

Fig 1: ASL Dataset

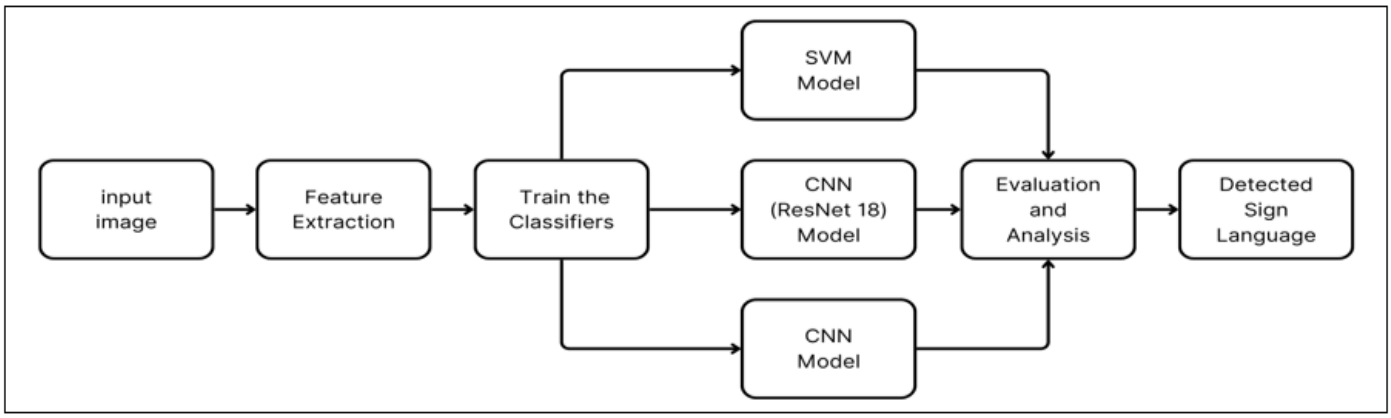

Fig 2: Proposed Methodology for Sign Language Recognition

C. Feature Extraction

➢ *HOG (Histogram of Oriented Gradients)*

Dalal and Triggs proposed HOG, a feature extraction technique for human detection, in 2005[10]. Firstly, the image gradients are calculated using derivative filters to determine the magnitude and direction of gradients for each pixel in both horizontal and vertical directions. The image is then separated into small cells and for each cell, a histogram of gradient orientations is produced, quantizing the orientations into discrete bins. Next, histograms within each block are normalized to enhance robustness against changes in illumination and contrast. Blocks of adjacent cells are formed, and their normalized histograms are concatenated to form block descriptors, capturing spatial gradient orientation patterns. Finally, all block descriptors are concatenated to generate the complete feature vector which represents the HOG features of the entire image, which can be utilized for various computer vision tasks.

➢ *Prewitt, Canny and Sobel*

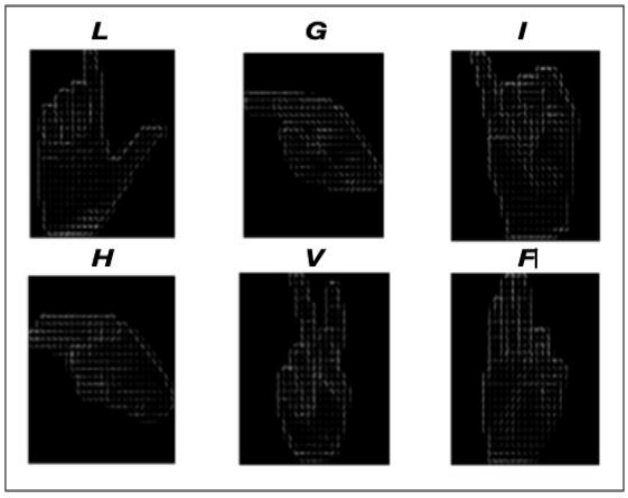

Fig 3: HOG Feature Extraction

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24MAY500

Prewitt, Canny, and Sobel are edge detection algorithms that are widely utilized as feature extraction methods in image classification. These methods detect rapid changes in intensity or color in an image, which are commonly associated with borders or boundaries between different objects or regions.

The Prewitt operator computes both gradient magnitude and direction for every pixel using a straightforward filter. A threshold is then utilized on the computed gradient amplitude to identify edge occurrences. The technique boasts high speed and efficiency; however, it may be vulnerable to inaccuracies caused by noise. A Prewitt mask is shown in the equation (1).

$$G_x = \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix} \quad G_y = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix} \qquad \ldots . (1)$$

John F. Canny created the advanced technique known as the Canny edge detector in 1986[11], which has superior capabilities. The process involves utilizing a Gaussian filter to minimize noise and determining gradient magnitude and direction for each pixel before applying high and low thresholds that establish resilient and feeble edges, respectively; afterward, frail edges attach together to generate a comprehensive perimeter map efficiently. Equation (2) depicts the equation for a Gaussian filter kernel with dimensions of (2k+1)×(2k+1) and we can find edge gradient and direction for each pixel using equation (3)

$$H_{ij} = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{(i-(k+1))^2 + (j-(k+1))^2}{2\sigma^2} \right); 1 \le i, j \le (2k+1) \qquad \ldots . (2)$$

$$Edge\_Gradient\,(G) = \sqrt{G_x^2 + G_y^2} \, Angle\,(\theta) = \tan^{-1}\left(\frac{G_y}{G_x}\right) \qquad \ldots . (3)$$

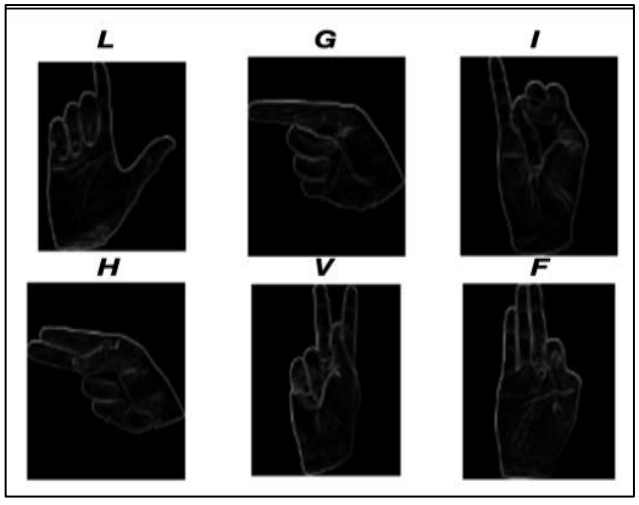

Fig 4: Prewitt Feature Extraction

Compared with its counterparts, this method is less vulnerable to interference; hence, it's widely believed among experts that it's one of the superlative techniques currently available for detecting an object's boundary reliably.

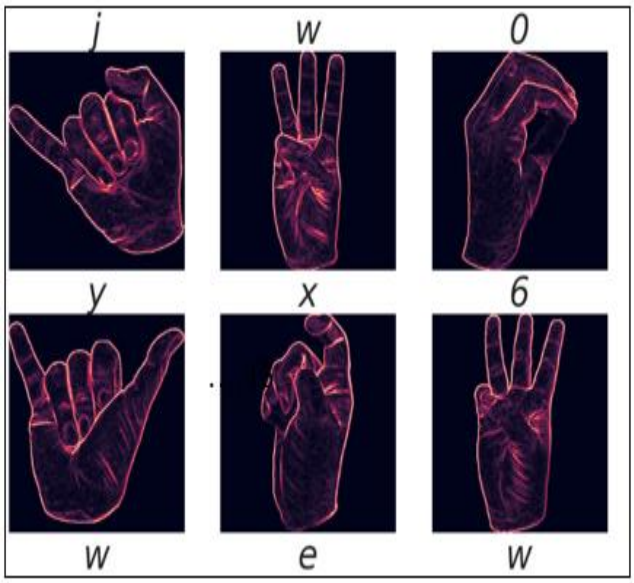

Fig 5: Canny Feature Extraction

The Sobel operator is like the Prewitt operator but instead it uses a different kernel for calculating the gradient magnitude and direction. The equation (4) shows a Sobel mask.

$$G_x = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} \quad G_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \qquad \ldots . (4)$$

The Sobel operator is comparatively more sensitive to noise but is faster and more efficient.

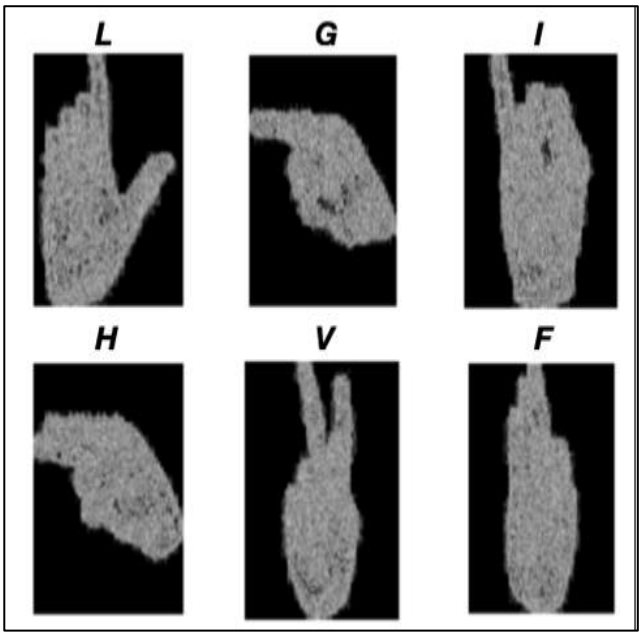

Fig 6: Sobel Feature Extraction

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24MAY500

➢ *Harris Corner Detection.*

Harris Corner Detection is a common technique for locating important areas in photos, which is applied to many computer vision tasks[12], such as sign language recognition. By identifying and tracking the corners of the hand and fingers in sign language, this method allows for the detection of key points and subsequent analysis of hand movements. The Harris Corner Detection method works by analyzing the intensity gradients at each pixel in an image, identifying those with significant changes in intensity. It calculates the intensity difference for a displacement of (u,v) in any direction. The expression for intensity gradients is shown in equation (5).

To detect corners, we must maximize the function *E(u,v)*. These are then designated as corners, which serve as distinguishing characteristics for tracking and recognition. This approach has proven to be quite effective in detecting sign language since it can deal with differences in lighting, orientation, and hand form, all of which are typical in real-world contexts.

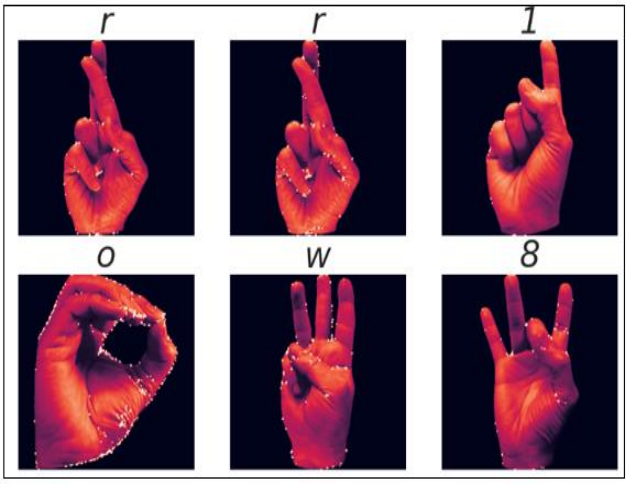

Fig 7: Harris Corner Detection

➢ *Contour Tracing.*

The extraction technique of contour tracing is based on identifying and outlining the shapes of objects within an image [13]. By detecting changes in intensity or color along object boundaries, this method can effectively identify and classify various objects. The shape information obtained through contour tracing enables valuable insights into a given objects.

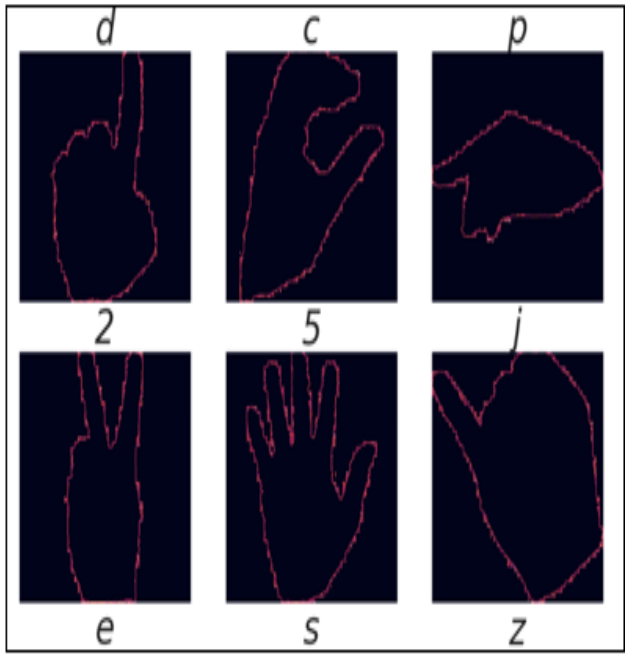

Fig 8: Contour Features

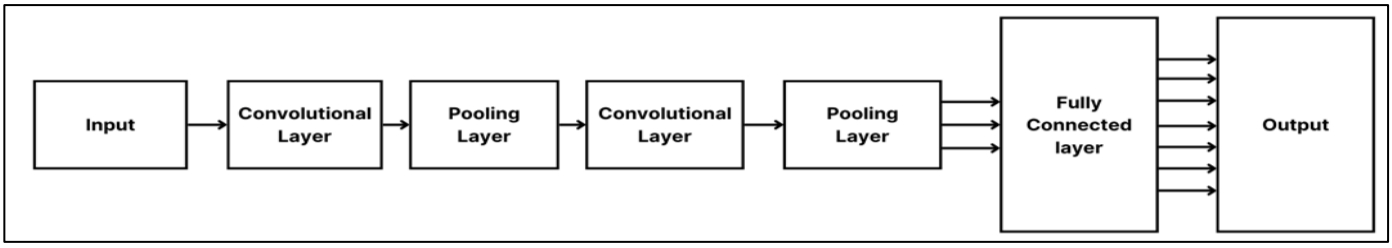

Fig 9: CNN Architecture

D. Models

➢ *Support Vector Machine*

SVMs are supervised learning techniques for classification and regression that are part of the generalized linear classifier family. SVM's capacity to decrease empirical classification error while maximizing geometric margin distinguishes it as a maximum margin classifier. Based on structural risk minimization (SRM), SVMs strive towards increasing separation between decision boundaries, hyperplanes, and nearest points from different classes. The simplest form of an SVM is achieved through a linear formulation where hyperplanes lie in the input data space x subset; hence, hypothesis spaces become subsets involving all hyperplane forms $f(x) = w \cdot x + b$. SVMs are empowered by the kernel trick to effectively manage non-linear connections in data without involving a direct feature transformation. Standard kernels include linear, radial basis functions (RBF), polynomial, and sigmoidal functions. Linearly separable data is best suited for the simplistic linear kernel. The expression for linear kernel is shown in equation (6).

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24MAY500

K(xi, xj) = xi•xj ……..…..(6)

The RBF kernel is widely used for handling non-linear relationships in the data.

K(xi, Xj) = exp (-ɤ||xi- xj||^2)…….(7)

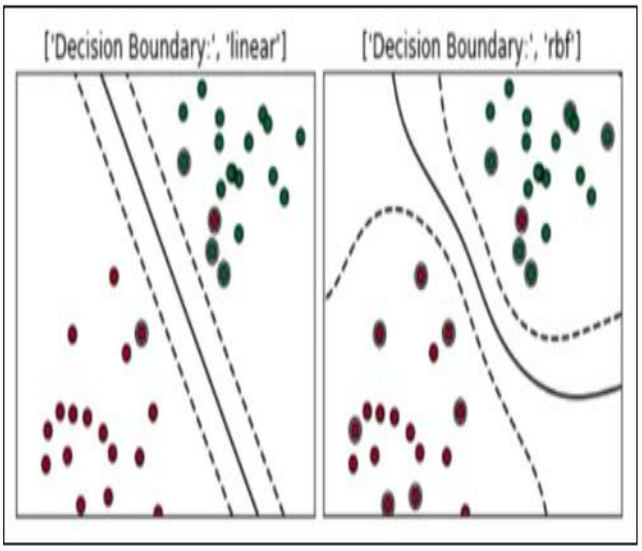

Fig 10: Linear and RBF Kernels [14]

In this classification, we employed SVM with a linear kernel. We used the extracted characteristics as feature vectors in the SVM to classify the signs. The training uses a total of 1760 photos. After training, the classifier's performance is tested on a testing set of 755 images using criteria such as overall accuracy, precision, and recall.

➢ *Convolutional Neural Network*

CNNs i.e. convolutional neural networks are also called deep learning systems that learn directly from input without requiring human feature extraction. These networks excel at identifying patterns in pictures to distinguish objects and environments, but they can also categorize non-image data like audio and signal information. Unlike fully connected multilayer perceptron models prone to overfitting with complete connectivity across all neurons of each layer, CNN's utilize regularization techniques for reducing connections, such as skipped connection dropouts, instead of punishing parameters during training. With a hierarchical approach toward constructing increasingly complex patterns using simple filters imprinted by smaller ground-level structures found earlier on the process chain, making them less demanding on complexity than regular MLPs, this architecture was inspired initially through biological processes where animal visual-brain involves intricated sparsity-processing capabilities driven by selective sensory stimuli based mainly thought neurophysiology's Receptive-Field Theory, whereby Responsive-Cortical-Neurons only reacted upon certain sections, forming maximum perimeter coverage regarding field perception.

Table 1. CNN Architecture

Layer name	Output size	Specifications
conv1 ReLu layer Max pooling	(200, 200, 32) (200, 200, 32) (100, 100, 32)	3x3 convolution, 32 filters, stride = 1
conv2 ReLu layer Max pooling	(100, 100, 64) (100, 100, 64) (50, 50, 64)	3x3 convolution, 64 filters, stride = 1
conv3 ReLu layer Max pooling	(50, 50, 128) (50, 50, 128) (25, 25, 128)	3x3 convolution, 128 filters, stride = 1
1st Fully connected layer Dropout layer	512	
2nd Fully connected layer Dropout layer	128	
Softmax layer Classification layer	37	

Our overall approach is a conventional CNN architecture with many convolutional and dense layers. Each CNN is three layers deep. The architecture has two convolutional layers, each with 32 filters and a 3x3 window size, followed by a max-pool and dropout layer. It is followed by a pair of convolutional layers, each with 64 filters, as well as a max pooling and dropout layer. The third block consists of two convolutional layers with 128 filters and a max pooling layer. A fully linked hidden layer with 512 neurons of the ReLU activation function is then applied, followed by a SoftMax output layer.

➢ *ResNet-18*

ResNet-18 is a convolutional neural network belonging to the Residual Networks (ResNet) family, designed for image classification. Microsoft Research created this architecture by combining an initial convolutional layer with max pooling, which helps reduce spatial dimensions. Its revolutionary feature lies in its residual blocks, which have two 3x3 kernels each and contain skip connections, allowing information to flow from one layer to another without encountering vanishing gradient problem issues, leading to the successful training of deep networks. The residuum block structure has batch normalization functions plus ReLU activation ones, adding capacity to capture complex patterns, followed by average pooling reducing size before fully connected layers predict classes. Researchers appreciate how ingenious incorporating skipping connections was in developing models trained on large datasets like images, giving superlative results trending globally among professionals participating across industries and successfully using it as their go-to algorithm solution. cutting-edge digitized processes workflows involving machine vision aesthetics enabling while avoiding both overfitting and underfitting exceptional performance, especially when processing challenging digital inputs. AI applications open doors. new expanded opportunities for businesses and

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24MAY500

industries counting upon state-of-the-art tools available today, representing future progress moving forward.

Table 2: Architecture of ResNet-18

Layer Name	Output Size	ResNet-18
conv1	112 × 112 × 64	7 × 7, 64, stride 2
conv2_x	56 × 56 × 64	3 × 3 max pool, stride 2 [3 × 3, 64; 3 × 3, 64] × 2
conv3_x	28 × 28 × 128	[3 × 3, 128; 3 × 3, 128] × 2
conv4_x	14 × 14 × 256	[3 × 3, 256; 3 × 3, 256] × 2
conv5_x	7 × 7 × 512	[3 × 3, 512; 3 × 3, 512] × 2
average pool	1 × 1 × 512	7 × 7 average pool
fully connected	1000	512 × 1000 fully connections
softmax	1000	

The Resnet-18 network we have employed consists of two-dimensional convolutional layers, batch normalization, max pooling, and fully linked layers. Each residual black contains 64,128,236 and 512 filters, respectively. The final fully connected layer of the original ResNet-18 is replaced by a new linear layer based on our output classes.

E. Training

During the training process, iterating over the training dataset for a specified number of epochs is crucial for optimizing model performance. Stochastic Gradient Descent (SGD) serves as the optimizer, adjusting model parameters to minimize the Cross-Entropy Loss function. Backpropagation computes the loss and updates the model's parameters accordingly. To align with the number of classes in the dataset, the last fully connected layer is replaced, and the evaluation mode is set. Data splitting involves allocating 70 percent for training and 30 percent for testing across all three datasets, ensuring robust model evaluation and generalization.

Table 3: Training Performance of Different Models

Epochs	SVM	ResNet	CNN
1	22.07%	19.32%	37.52%
2	57.18%	54.71%	83.90%
3	69.30%	70.82%	91.65%
4	77.10%	73.84%	95.33%
5	79.17%	78.17%	96.27%
6	81.11%	81.95%	97.61%
7	82.74%	82.82%	97.91%
8	83.86%	84.37%	98.46%
9	84.61%	84.21%	98.81%
10	87.36%	85.69%	99.20%

Table 3. shows overall accuracy evolution of models. In which it has been seen that accuracy is increasing noticeably. CNN exhibits a significantly higher initial accuracy of 37.5%, which rapidly escalates to 99.20% over the course of the epochs, indicating its remarkable efficiency in learning and capturing complex patterns within the training data. Overall, CNN outperforms ResNet and SVM, showcasing the highest training accuracy rates across all epochs.

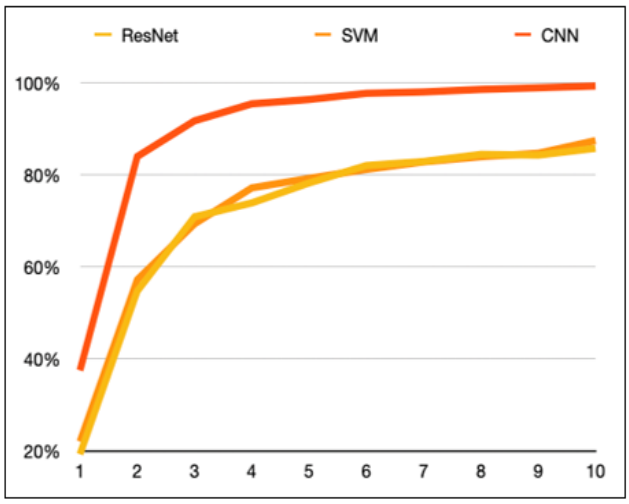

Fig 11: Overall Training Accuracy

Table 4: Comparison of training loss of models

Epochs	SVM	ResNet	CNN
1	2.0033	1.9403	0.5966
2	1.3442	1.3434	0.2440
3	1.0498	1.0246	0.1739
4	0.8753	0.8606	0.1594
5	0.7347	0.7531	0.1592
6	0.6600	0.6630	0.1537
7	0.6026	0.6061	0.1475
8	0.5779	0.5482	0.1342
9	0.5354	0.4817	0.1201
10	0.4732	0.3971	0.1161

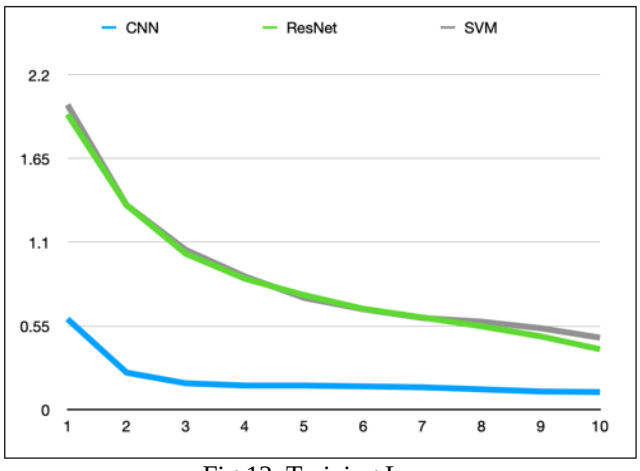

Fig 12: Training Loss

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24MAY500

This trend indicates that all models effectively learn from the training data over the epochs, with CNN consistently exhibiting the most substantial decrease in loss, implying its superior ability to optimize and generalize compared to ResNet and SVM.

IV. RESULTS AND ANALYSIS

The overall accuracy comparison table illustrates the performance of three models CNN, ResNet, and SVM across various feature extraction methods: Canny, contour, Harris, Prewitt, watershed, and Sobel. Across the different feature extraction techniques, CNN consistently achieves competitive accuracy rates, with the highest scores observed in Sobel (96.03%), Prewitt (95.03%), and contour (95.03%) methods.

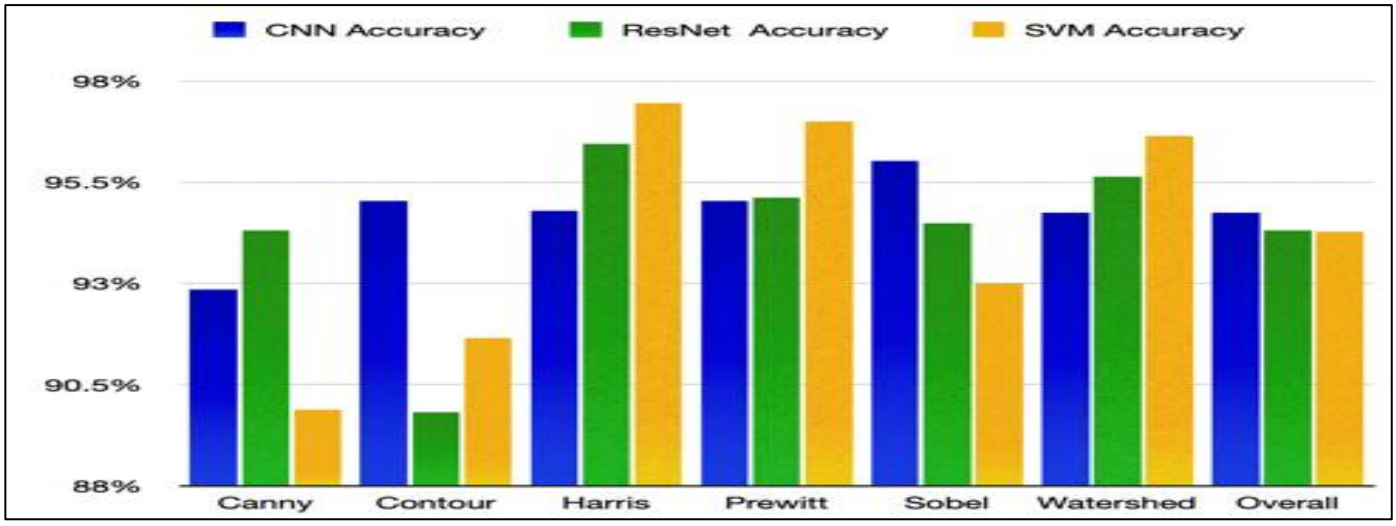

Fig 13: Feature Based Analysis

Table 5: Model Performance based on Different Features

	SVM	ResNet	CNN
Canny	89.88%	94.30%	92.85%
Contour	91.65%	89.81%	95.03%
Harris	97.45%	96.45%	94.8%
Prewitt	97%	95.11%	95.03%
Sobel	93%	94.49%	96.03%
Watershed	96.64%	95.64%	94.748%
Overall	94.27%	94.3%	94.748%

ResNet also demonstrates strong performance, particularly excelling with the Harris feature extraction method (96.45%). SVM generally lags behind CNN and ResNet but still maintains respectable overall accuracy rates, with the highest score achieved using the Harris method (97.45%). Overall, while ResNet and SVM show notable accuracy levels, CNN consistently performs well across most feature extraction techniques, indicating its robustness and effectiveness in handling diverse image features.

The study analyzes categorization using the following parameters: recall value or sensitivity (R), precision (P), F1 score (F1), accuracy (A), and error.

Table 6: Comparison of Models

	Precision	Recall	F1-Score	Accuracy
SVM	95.16%	94.56%	94.22%	92.26%
Resent 18	95.83%	95.30%	94.89%	94.30%
CNN	95.14%	94.81%	94.73%	95.54%

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24MAY500

In terms of precision, ResNet exhibits the highest value at 95.83%, closely followed by SVM at 95.16%, with CNN slightly lower at 95.14%. In terms of overall accuracy, ResNet stands out with a score of 94.3%, followed closely by CNN at 94.74%, and SVM slightly lower at 92.26%. Overall, ResNet consistently demonstrates competitive performance across all metrics, while CNN closely follows, and SVM lags slightly behind in terms of precision, F1 score, overall accuracy, and recall.

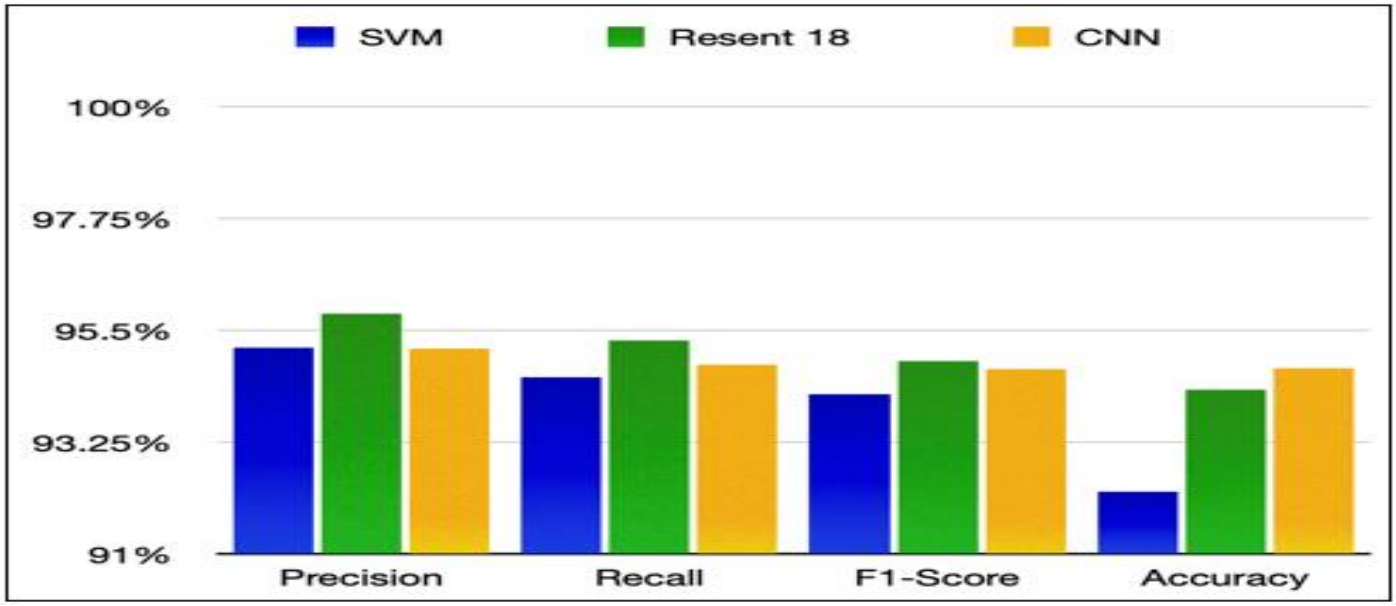

Fig 14: Quantitative Analysis

Table 7: Class Wise Comparison of Each Model

Class	ResNet	SVM	CNN	Class	ResNet	SVM	CNN
0	65%	64%	93%	I	100%	93%	100%
1	96%	91%	88%	J	100%	94%	100%
2	100%	93%	92%	K	100%	81%	100%
3	100%	100%	100%	L	100%	100%	100%
4	100%	100%	100%	M	100%	79%	87%
5	100%	100%	93%	N	100%	70%	92%
6	95%	90%	93%	O	87%	92%	93%
7	100%	100%	100%	P	100%	100%	100%
8	100%	94%	93%	Q	100%	100%	100%
9	100%	100%	100%	R	100%	93%	86%
A	100%	100%	100%	S	100%	100%	100%
B	100%	100%	100%	T	96%	100%	100%
C	100%	100%	100%	U	100%	100%	78%
D	96%	91%	100%	V	96%	86%	92%
E	100%	100%	100%	W	88%	82%	87%
F	100%	86%	100%	X	100%	88%	100%
G	100%	94%	100%	Y	100%	93%	100%
H	100%	100%	100%	Z	100%	80%	100%

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24MAY500

From table (7), we observe a comparative analysis of three distinct classifiers - SVM, CNN and Resnet-18 across a diverse set of 36 classes, encompassing both numerical digits (0-9) and alphabetical letters (A-Z). The ResNet architecture demonstrates a commendable level of accuracy, particularly excelling with a flawless 100% in the majority of classes. It does, however, exhibit some challenges, most notably with the numerical class '0' where it achieves only 65% accuracy, and to a lesser extent with the classes '1', 'O', 'T', and 'V'. The SVM classifier, while achieving perfect scores in several instances, shows a more erratic performance profile with significant dips in accuracy for certain classes. It struggles considerably with the numerical class '0' at 64% and the alphabetical classes 'N' at 70%, 'M' at 79%, and 'K' at 81%, indicating potential weaknesses in its classification capabilities for these particular characters. The CNN classifier maintains a robust performance across the dataset, with perfect scores in numerous classes, but it is not without its shortcomings, as evidenced by lower accuracies in classes such as 'W', 'U', and 'R'. Despite these individual variances, the overall performance of each classifier is impressive, showcasing their ability to effectively discern and classify a wide range of characters. In summary, ResNet appears to be the most consistent and accurate across the majority of classes, with only a few instances of reduced accuracy. SVM, while achieving high accuracy in certain classes, shows more pronounced dips in performance, particularly with specific numerical and alphabetical classes. CNN generally performs well, with a few exceptions where its accuracy falls below that of ResNet.

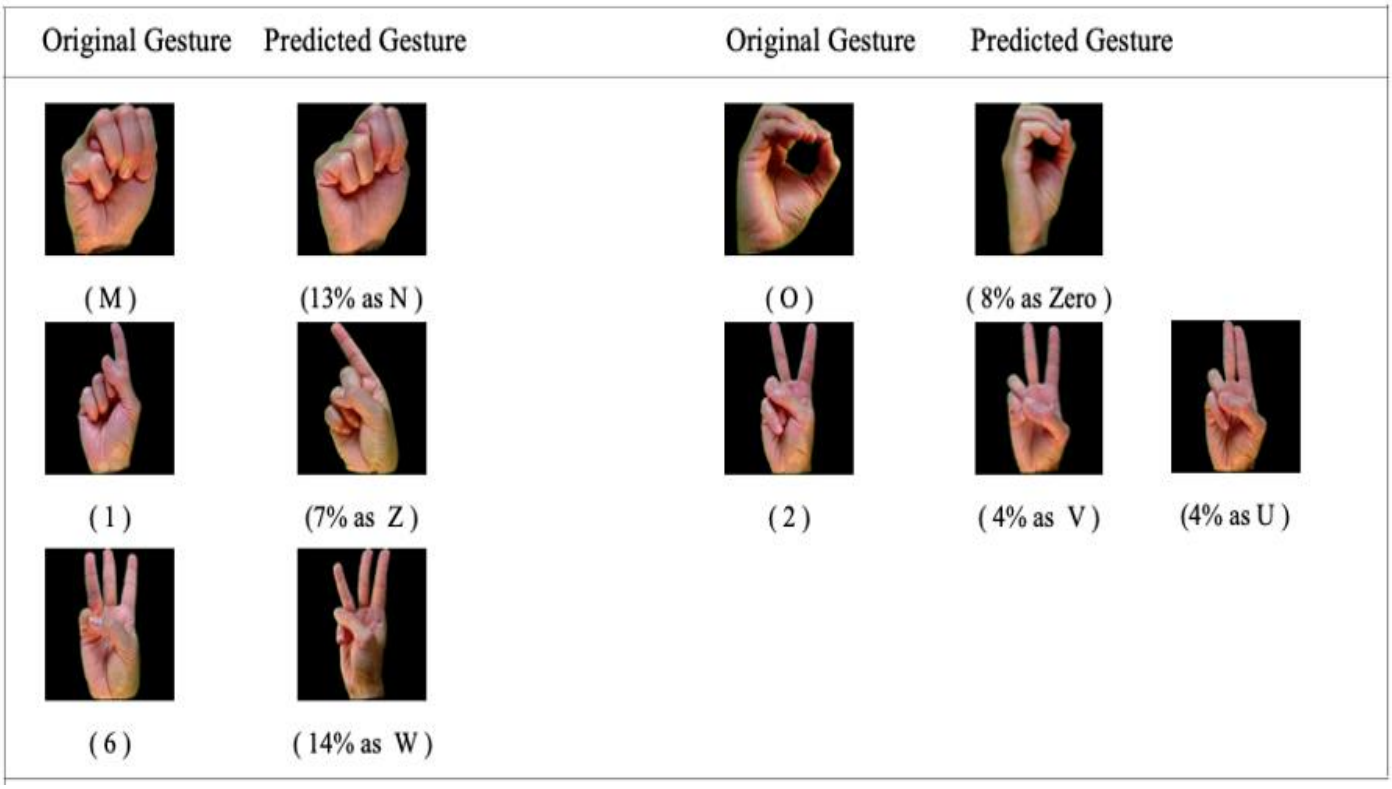

Fig 15: Illustration of Misclassification of Gestures

V. CONCLUSION

Resnet-18 and CNN are deep learning models that can learn and recognize complicated patterns and features in data.. On the other hand, SVM is a machine learning model that focuses on identifying the best border between data points, which may not be suitable for recognizing complex hand gestures in sign language. In sign language detection system, overall accuracy denotes the model's capacity to precisely identify and categorize different signs. The higher the accuracy, the greater the reliability of the system. Our study revealed that CNN achieved the highest accuracy rate at 95.10%, surpassing ResNet 18 (94.3%) and SVM (94.27%).Our research underscores the consistent performance of CNN and ResNet. While CNN exhibits superiority over ResNet 18 and SVM in overall accuracy, precision, and adaptability, making it better suited for sign language recognition systems.

Despite achieving commendable accuracy rates, certain sign language gestures pose challenges due to their resemblance in landmarks. For instance, distinguishing between the hand gestures for the alphabet "W" and the number “6", or between "O" and “0", often leads to misclassifications, as evidenced by their confusion matrices. Future research could focus on developing neural network designs for sign language recognition to improve efficiency in order to capture and interpret the unique properties of sign language motions with even greater accuracy.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24MAY500

REFERENCES

[1]. Wadhawan, Ankita., & Kumar, Parteek. (2020). Deep learning-based sign language recognition system for static signs. Neural Computing and Applications , 32 , 7957 - 7968. [doi.org/10.1007/s00521-019-04691-y]

[2]. Masood, S.., Srivastava, Adhyan., Thuwal, H.., & Ahmad, Musheer. (2018). Real-Time Sign Language Gesture (Word) Recognition from Video Sequences Using CNN and RNN. , 623-632. [doi.org/10.1007/978-981-10-7566-7_63]

[3]. Rastgoo, R.., Kiani, K.., & Escalera, Sergio. (2020). Video-based isolated hand sign language recognition using a deep cascaded model. Multimedia Tools and Applications, 79, 22965 - 22987. [doi.org/10.1007/s11042-020-09048-5]

[4]. Koller, Oscar., Zargaran, Sepehr., Ney, H.., & Bowden, R.. (2016). Deep Sign: Hybrid CNN-HMM for Continuous Sign Language Recognition. [doi.org/10.5244/C.30.136]

[5]. Koller, Oscar., Zargaran, Sepehr., Ney, H.., & Bowden, R.. (2018). Deep Sign: Enabling Robust Statistical Continuous Sign Language Recognition via Hybrid CNN-HMMs. International Journal of Computer Vision, 126, 1311-1325. [doi.org/10.1007/s11263-018-1121-3]

[6]. Katoch, Shagun., Singh, Varsha., & Tiwary, U.. (2022). American Sign Language recognition system using SURF with SVM and CNN. Array , 14 , 100141. [doi.org/10.1016/j.array.2022.100141]

[7]. Barbhuiya, Abul Abbas., Karsh, R.., & Jain, Rahul. (2020). CNN based feature extraction and classification for sign language. Multimedia Tools and Applications , 80 , 3051 – 3069. [doi.org/10.1007/s11042-020-09829-y]

[8]. Huang, Jie., Zhou, Wen-gang., Li, Houqiang., & Li, Weiping. (2019). Attention-Based 3D-CNNs for Large-Vocabulary Sign Language Recognition. IEEE Transactions on Circuits and Systems for Video Technology , 29 , 2822-2832. [doi.org/10.1109/TCSVT.2018.2870740]

[9]. Sasikala, N., Swathipriya, V., Ashwini, M., Preethi, V., Pranavi, A., and Ranjith, M. Feature extraction of real-time image using sift algorithm. *European Journal of Electrical Engineering and Computer Science* *4*, 3 (2020). [doi.org/10.24018/ejece.2020.4.3.206.]

[10]. Dalal, N., and Triggs, B. Histograms of oriented gradients for human detection. In 2005 *IEEE computer society conference on computer vision and pattern recognition* (CVPR'05) (2005), vol. 1, Ieee, pp. 886893. [doi 10.1109/CVPR.2005.177]

[11]. Rekha, J., Bhattacharya, J., and Majumder, S. Shape, texture and local movement hand gesture features for Indian sign language recognition. In 3rd international conference on trends in information sciences & computing (TISC2011) (2011), IEEE, pp. 3035. [dx.doi.org/10.1109/tisc.2011.6169079]

[12]. Ram, P., and Padmavathi, S. Analysis of harris corner detection for color images. In 2016 International Conference on Signal Processing, Communication, Power and Embedded System (SCOPES) (2016), IEEE, pp. 405410. [doi**:** 10.1109/SCOPES.2016.7955862]

[13]. Chang, F., and Chen, C.-J. A component labelling algorithm using contour tracing technique. In Seventh International Conference on Document Analysis and Recognition, 2003. Proceedings. (2003), vol. 3, Citeseer, pp. 741741. [doi:10.1109/ICDAR.2003.1227760]
https://miro.medium.com/v2/resize:fit:1400/format:webp/1*Ha7EfcfB5mY2RIKsXaTRkA.png