ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)

Vol 12, Issue 01, January 2025

SignSpectra: Sign Language Translator

[1] Shailaja Udtewar, [2] Tanmay Sawant, [3] Pratiksha Sarvankar, [4] Mahek Shaikh

[1] Professor, Xavier Institute of Engineering, Department of Electronics & Telecommunication, Mumbai, India
[2] [3] [4] Xavier Institute of Engineering, Department of Electronics & Telecommunication, Mumbai, India
Corresponding Author Email: [1] shailiayush@gmail.com, [2] tvsawant22119@gmail.com, [3] pratiksha.s.1702@gmail.com, [4] smahek2108@gmail.com

Abstract*— SignSpectra: Sign Language Translator is a multi-phase AI/ML project dedicated to bridging the communication gap between the deaf/mute community and the hearing population. This system utilizes Indian Sign Language (ISL) as the primary medium for translation, transforming sign language into spoken English, thereby fostering inclusive interactions. The project is organized into three developmental phases: Phase 1 (P1) involves training models on a dataset containing alphabets and numbers; Phase 2 (P2) expands on this foundation with a dataset of commonly used words; and Phase 3 (P3) further extends the system's functionality by incorporating sentence structures. In Phase 1, AlexNet and VGG19 architectures were employed to classify alphabets and numbers. For Phase 2, AlexNet was used exclusively, leveraging its performance with word-level classification. Phase 3 adopts a more complex approach, utilizing EfficientNet B0 and LSTM architectures to handle sentence level translations, achieving a higher degree of contextual accuracy for natural communication flow. Throughout the project, careful attention was paid to model overfitting and accuracy optimization, with techniques such as early stopping, splitting the dataset into train, test, validation, and limiting the number of files trained per epoch. The entire system was developed and tested across multiple IDEs, including Spyder, Google Colab, and Kaggle. Each phase has been completed successfully, demonstrating the system's capability to translate ISL into spoken English, thereby enhancing accessibility for the deaf/mute community.

***Index Terms*— Accessible Communication, Deep Learning, Indian Sign Language, Sign-to-Speech Translation.**

I. INTRODUCTION

The SignSpectra: Sign Language Translator project is an important initiative that bridges a critical communication gap between the deaf and hearing communities, while at the same time providing significant social and technological benefits. By translating Indian Sign Language (ISL) into spoken English, the system enables people who communicate using sign language to interact more freely with others, thereby promoting inclusion and enriching the quality of life. The project's structured approach is divided into phases that focus on translating ISL alphabets, numbers, words, and sentences, ensuring a comprehensive understanding and accurate representation of linguistic elements. This systematic approach helps the model develop a solid understanding of the fundamental yet complex components of communication, providing a solid foundation for practical application. The project uses advanced machine learning architectures such as AlexNet, VGG19, EfficientNet B0, and LSTM networks to improve the accuracy and contextual awareness of the translation system. These architectures enable the system to process and interpret sign language with high accuracy, ensuring that gestures are correctly recognized and converted into coherent spoken English. Integrating these models allows SignSpectra to maintain a balance between recognizing the detailed features of sign language and understanding the broader context that is essential for natural, fluid communication. This functionality makes the system applicable in a variety of contexts, including the workplace, educational institutions and public service environments, where smooth communication will significantly improve access to information and promote social and professional participation. By enabling accurate translation, the project will help remove the barriers that people who use sign language often face. In addition to its direct impact, SignSpectra promotes accessibility and contributes to technological innovation that creates precedents in the future development. This system offers a foundation that expands the possibility of translating gestures into English, and opens a way to a more comprehensive communication solution on a global scale. Through this initiative, this project has contributed to global initiatives aimed at creating a more fair and understandable society.

II. RELATED WORKS

The authors [1] introduce a model based on Vision Transformer (ViT) for the static recognition of Indian Sign Language (ISL), utilizing an internal attention mechanism that has proven effective in natural language processing to achieve high recognition accuracy. The dataset consists of 36 gesture classes encompassing ISL numbers (0-9) and the alphabet, with over 1,000 RGB images for each class enhanced for diversity through methods such as rotation, flipping, and brightness adjustment. Images are resized to 72×72 pixels and segmented into 144 patches (6×6 each) to generate input sequences for the transformer. The model architecture utilizes ViT to convert images into spatially relevant patches, processed through six internal attention layers using Multi-Head Self-Attention (MHSA) for effective feature extraction. Gesture classification is carried out by a four-layer MLP classifier employing ReLU activation. The model performs exceptionally well with various backgrounds and lighting conditions, achieving 99. 29% accuracy across 36 gesture classes in just five epochs, validated by precision, recall, and F1 scores. It surpasses CNN-based models such as ResNet in accuracy and training efficiency, effectively handling large datasets without the need for extensive preprocessing. The study concludes that Vision Transformers with multi-head attention greatly enhance static gesture recognition while tackling challenges like background variations and computational efficiency. Future work anticipates extending the model to dynamic and continuous sign languages with temporal coding for video-based recognition and including multimodal inputs such as facial expressions for a thorough interpretation of gestures.

The authors in [2] introduce a system aimed at bridging the communication divide between sign language users and those who do not use sign language. The system provides two main features: a sign language learning tool that allows users to practice and assess signs, and a real-time translation feature that captures live video gestures and converts them into text. Rooted in Indian Sign Language (ISL), the system utilizes image processing methods with OpenCV and a convolutional neural network (CNN) for gesture recognition. The dataset comprises specially selected gestures focusing on arm movements, processed to highlight key areas. Preprocessing includes skin segmentation, dilation, and erosion techniques to separate hand gestures from the background. The CNN model is trained on these processed images for accurate gesture classification. Issues such as the variability of hand gestures, achieving real-time performance, and ensuring accessibility for users were effectively resolved. The methodology involves data gathering, preprocessing, feature extraction, and CNN-based classification to present recognized gestures as text. The system achieves real-time recognition with high precision, integrating sophisticated image processing and deep learning to surpass previous static recognition systems. Future plans include broadening support for various sign languages, enhancing the recognition of intricate gestures, and adding speech output for two-way communication to increase versatility and inclusiveness.

The authors [3] introduce a technique for real-time recognition of static hand gestures using fine-tuned Convolutional Neural Networks (CNNs) to enhance accuracy and tackle issues like dataset restrictions and variability in hand gestures. They employed the Massey University (MU) dataset containing 2515 images across 36 classes and the HUST-ASL dataset with 5440 samples, both marked by variations in hand shapes, lighting, and background noise. Pre-trained CNN models AlexNet and VGG-16 were fine-tuned for feature extraction and classification, modifying weights for the specific gesture recognition task. The challenges included small dataset sizes, gesture similarity under different conditions, and background noise, addressed by using pre-trained models and a score-level fusion approach that merges results with optimal weights. The methodology involved data preprocessing, model fine-tuning, and evaluation through leave-one-subject-out (LOO) and regular cross-validation. Results from the MU dataset yielded accuracies of 90. 26% (LOO) and 98. 14% (regular), while the HUST-ASL dataset recorded 56. 18% and 64. 55%, respectively. The system processed gestures in 0. 52 seconds, indicating practical applicability despite challenges in misclassification between similar gestures. The study concludes that fine-tuned CNNs with score-level fusion improve resilience to noise and gesture variation, recommending future investigation into shape-based feature extraction and larger datasets for better model performance.

The authors [4] introduce a dual-function system aimed at closing the communication gap for those with hearing and speech challenges by converting Indian Sign Language (ISL) gestures into English text and transforming spoken English into the equivalent ISL gestures. By employing machine learning models, image processing methods, and Google's speech recognition API, the system achieves notable accuracy and user-friendliness. The dataset consists of 42 gesture classes, encompassing 50,391 images, with thorough pre-processing for consistency and noise reduction. Among the models evaluated, Support Vector Machine (SVM) paired with K-means clustering and Bag of Visual Words (BoV) reached an impressive accuracy of 99. 5%, surpassing CNNs (88. 89%) and RNNs (82. 3%) in terms of accuracy and reliability. Real-time recognition was accomplished with a latency of 0. 04 seconds per frame, supporting live use cases. For gesture-to-text conversion, pre-processed image data was categorized using SVM, while speech-to-gesture mapping relied on Google's speech API to align spoken text with ISL gestures instantaneously. Challenges included variability in gestures, extensive standardization of the dataset, and the need for real-time processing. The system exhibited enhanced performance compared to earlier ISL recognition research, with its dual-functionality increasing applicability in practical situations. Future endeavors will focus on enlarging the dataset, refining sentence recognition, incorporating text-to-audio components, and creating a more comprehensive communication framework.

The authors [5] introduce a reliable system for recognizing Indian Sign Language (ISL) from live video feeds, enhancing communication between hearing-impaired individuals and others. By employing a hybrid model that integrates Convolutional Neural Networks (CNNs) and Long Short-Term Memory Networks (LSTMs), the system proficiently captures both spatial and temporal features of gestures, achieving an impressive accuracy of 95. 99%. Tested on a varied Castalk-ISL dataset containing 5,000 video samples of 50 ISL words demonstrated by 10 individuals, the model generalizes effectively across different users and settings. The architecture utilizes Inception V3 for extracting spatial features, with a Global Average Pooling layer that minimizes dimensionality, and incorporates LSTM for analyzing sequential data to capture dependencies across video frames. A SoftMax layer accurately classifies gestures, with precision, recall, and F1-score values of 96. 10%, 95. 99%, and 96. 05%, respectively. Surpassing ResNet and 3D-CNN-based methods, especially in recognizing dynamic gestures, the system exhibits exceptional real-time performance. Error analysis reveals difficulties with gestures that have similar movements, such as "hello" and "good evening. " It uses an hybrid model approach for training its dataset. Future efforts will focus on expanding the dataset for sentence-level recognition, decreasing computational demands, and enhancing real-time processing for greater accessibility.

The authors [6] present a groundbreaking vision-based method to close the communication gap between the deaf community and the broader society. The primary aim of the system is to convert Indian Sign Language (ISL) gestures into text and speech utilizing an advanced neural network architecture. The system merges a 3D Convolutional Neural Network (3D-CNN), which analyzes spatial and depth data from gesture images, with a Long Short-Term Memory (LSTM) network that handles the temporal elements of gesture sequences, allowing accurate identification of both static and dynamic gestures. The training dataset includes images of the ISL alphabet, numbers, sentences, and emerging words. To enhance the quality of the input data, pre-processing techniques such as background isolation and feature extraction were implemented. Faster R-CNN in the Gesture Area Acquisition module guarantees accurate hand region detection, even amidst background noise or motion blur, eliminating reliance on glove-based systems that are susceptible to hardware malfunctions. Issues like managing dynamic gestures and diverse backgrounds are tackled through effective feature extraction and LSTM-based temporal evaluation. The suggested approach consists of three modules: Gesture Area Acquisition through Faster R-CNN for hand detection, Feature Extraction via 3D-CNN for spatial and depth assessment, and LSTM Encoding to link gestures to text and speech. An intuitive interface facilitates real-time gesture capturing and conversion into speech. Performance metrics indicate a high level of accuracy in recognizing both static and dynamic gestures, surpassing traditional glove-based and 2D image processing systems, while providing an affordable and accessible 3D-based alternative. The article concludes by highlighting the potential to broaden datasets to encompass complex gestures, enhance dynamic gesture recognition, and incorporate regional language support for improved scalability and inclusivity.

The author [7] highlights important challenges in enabling communication between the hearing impaired and the banking sector using Indian Sign Language (ISL) gestures. This study is significant for its investigation into converting ISL gestures related to banking into text. The authors encountered difficulties such as a small self-recorded dataset, differences in video length, and the intricacy of ISL gestures involving complex hand movements and body interactions. The dataset, made up of 1,100 self-recorded videos at a resolution of 1080x1920 pixels and 40 fps, was divided 80-20 for training and testing. The authors used a two-part deep learning model: Sparkle Neural Networks (CNN) for feature extraction and Long Short-Term Memory (LSTM) networks for gesture classification. The V3 base architecture was utilized to extract image features, which were then processed by the LSTM model to interpret gestures as text. Recognizing dynamic signs was especially challenging due to their sequential characteristics, and variations in gestures added further complexity. The limited dataset heightened the risk of overfitting and diminished generalization. The structured approach consisted of converting videos to still images, extracting features with a CNN, and classifying them using LSTM. The results demonstrated an 85% accuracy for banking gestures and 92% for everyday gestures, yielding an overall precision of 81%. The model performed effectively, particularly for certain gestures like "Debit Card" and "EXPIRY," but occasionally confused similar gestures such as "balance" and "working hours," a situation linked to the small dataset and feature overlap. The study illustrates the potential of deep learning for ISL recognition in banking but suggests a need for expanding the dataset, recognizing sentences, and integrating text-to-speech functionalities for enhanced interaction and comprehensive solutions.

The authors [8] explore a novel system aimed at closing the communication divide between sign language users and those who do not utilize it. This system converts Indian Sign Language (ISL) gestures into spoken English instantaneously, enabling effective communication between individuals with hearing and speech challenges and the wider community. A key component of this solution is the application of Convolutional Neural Networks (CNN) to recognize gestures, alongside a Text-to-Speech (TTS) translator that turns recognized gestures into audio. The system underwent testing using a dataset of stationary ISL gestures, including the alphabet and fundamental words, recorded under diverse conditions to create a comprehensive training set. The CNN analyzes input images to detect and categorize hand movements, implemented via Keras on TensorFlow for greater scalability. The TTS library, Google TTS (gTTS), is utilized for generating speech. A primary challenge encountered was addressing user variability, which encompasses distinct signing styles, body language, and physical traits, while also ensuring real-time processing with minimal latency. The methodology was organized into stages, with Phase 1 concentrating on identifying stationary gestures, iterating the model over several epochs to enhance accuracy, and converting gestures into immediate audio output. The model attained 85-95% accuracy on training data and 75-85% on validation data after 50 epochs, demonstrating steady performance improvement. The system's capacity to interpret stationary gestures and generate real-time speech was validated, using confidence intervals to assess prediction dependability. In contrast to conventional glove-based systems, this method employs CNN and TTS libraries, providing a hardware-independent solution and focusing on ISL, which has received less attention compared to ASL or BSL. The study concludes that the combination of CNN and TTS offers an efficient approach for real-time ISL translation, setting the stage for future advancements such as dynamic gesture recognition and sentence-level translation. Future enhancements may involve incorporating sentiment analysis through facial expressions and body language, as well as broadening support for additional dialects of sign and spoken languages to further enhance accessibility.

III. DESIGN METHODOLOGY

A. AlexNet Architecture

AlexNet is a profound convolutional neural organize (CNN) that accomplished noteworthy victory in the field of computer vision. Created by Geoffrey Hinton and his group, it won the 2012 ImageNet Huge Scale Visual Acknowledgment Challenge by accomplishing a huge edge over past strategies. The design comprises of 8 layers, counting 5 convolutional layers and 3 completely associated layers, with the essential objective of moving forward picture classification execution. It consolidates a few novel methods such as ReLU enactment, Neighborhood Reaction Normalization (LRN), and Dropout, which essentially contributed to its success. The arrange is outlined to handle high-resolution pictures with a few components that center on highlight extraction, non-linearity presentation, and regularization.

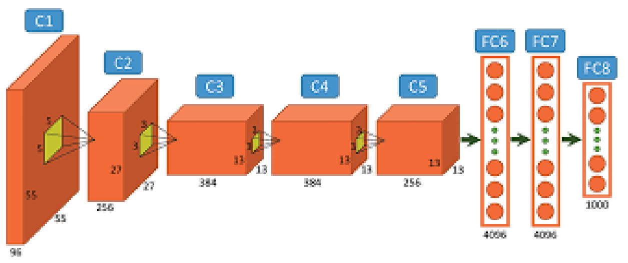


Fig. 1. AlexNet Architecture

Input Layer: The input layer acknowledges pictures with measurements 227x227x3, where 227x227 speaks to the stature and width of the input picture, and 3 indicates the three RGB color channels. The input picture is pre-processed by normalizing the pixel values (regularly subtracting the cruel of the dataset), which makes a difference to progress the model's learning handle amid training.

Convolutional Layer: AlexNet employments a add up to of 5 convolutional layers, each outlined to capture progressively complex highlights in the picture. The subtle elements of each convolutional layer are as follows: First Convolutional Layer: The to begin with layer employments 96 channels of estimate 11x11 with a walk of 4. This captures low-level highlights such as edges and textures. Second Convolutional Layer: This layer employments 256 channels of estimate 5x5 to capture more complex designs and textures. Third, Fourth, and Fifth Convolutional Layers: These layers utilize 384 channels of measure 3x3. As the arrange develops, these layers capture high-level designs, such as parts of objects and theoretical features.

ReLU: Each convolutional operation is taken after by the ReLU (Corrected Direct Unit) actuation work. ReLU presents non-linearity into the organize, permitting it to learn complex designs. For each positive esteem in the input, ReLU returns the esteem itself, and for all negative values, it returns zero. This makes a difference the arrange to maintain a strategic distance from the vanishing angle issue and quickens preparing, making it a favored choice in profound learning architectures.

LRN: Local Reaction Normalization (LRN) is an critical procedure utilized in AlexNet, connected after the ReLU enactment in the to begin with two convolutional layers. LRN makes a difference to normalize the actuations inside a neighborhood neighborhood by emphasizing the bigger actuations and smothering littler ones. This strategy energizes competition among neurons, which progresses the model's generalization capacity and strength. LRN is especially valuable for upgrading execution in large-scale neural systems like AlexNet.

Max Pooling Layer: Max pooling is connected after a few convolutional layers to decrease the spatial measurements of the include maps. This operation diminishes the computational complexity and makes a difference to center on the most noteworthy highlights whereas diminishing the number of parameters. AlexNet employments a 3x3 max pooling channel with a walk of 2, which holds the most imperative highlights whereas down-sampling the include maps.

Fully Associated Layer: After the convolutional and pooling layers, the organize straightens the highlight maps into a 1D vector. This vector is passed through 3 completely associated layers: FC1 and FC2: These layers comprise of 4096 neurons each. ReLU actuation is connected in both layers to learn theoretical and high-level representations of the highlights extricated from the past layers. FC3: The last completely associated layer comprises of 1000 neurons, comparing to the 1000 classes in the ImageNet classification assignment. The softmax enactment work is utilized here to compute a likelihood dissemination over the classes. The lesson with the most noteworthy likelihood is chosen as the output.

Dropout Layer: To anticipate overfitting, AlexNet applies dropout amid preparing in the completely associated layers. Dropout arbitrarily cripples a division of neurons amid each emphasis, constraining the arrange to learn repetitive representations. This makes a difference to make strides the model's generalization capability.

Softmax Layer: The softmax layer is the last layer of the arrange. It takes the yield from the final completely associated layer and changes over it into a likelihood dissemination, guaranteeing that the whole of probabilities over all yield classes rises to 1. The lesson with the most noteworthy likelihood is chosen as the anticipated lesson.

B. VGG19 Architecture

VGG19 is a significant convolutional neural orchestrate (CNN) laid out for picture classification assignments. It highlights a include up to of 19 layers, checking 16 convolutional layers and 3 totally related layers. The building takes after a fundamental be that as it may exceedingly beneficial structure that grants it to capture different leveled highlights from pictures utilizing a consistent design.

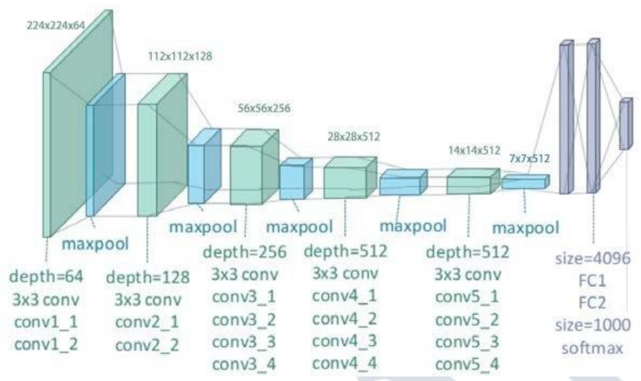


Fig. 2. VGG19 Architecture

Input Layer: The illustrate takes an input picture of gauge 224x224 pixels with 3 color channels (RGB). Customarily, the picture is normalized a few time as of late being fed into the orchestrate to advance appear execution in the midst of training.

Convolutional Layers: VGG19 utilizes 3x3 convolutional channels in its layers to learn fine-grained highlights such as edges, surfaces, and plans. These channels allow the organize to distinguish distinctive points of view of the picture. Each convolution operation is taken after by the application of the ReLU (Revised Straight Unit) sanctioning work, which presents non-linearity. This makes a distinction the appear learn complex plans in the data and makes strides its capacity to generalize to present day inputs. The dependable utilize of 3x3 channels all through the organize streamlines its arrange though effectively capturing spatial associations in the images.

Max Pooling Layers: After the convolutional layers, max pooling operations are associated to reduce the spatial estimations of the incorporate maps and lessen the computational stack. A 2x2 max pooling layer with a walk of 2 downsamples the incorporate maps, ensuring that the appear holds as it were the most crucial highlights though reducing the chance of overfitting.

ReLU: The ReLU sanctioning work is utilized after each convolutional layer to show non-linearity into the orchestrate. This work returns the input regard for positive inputs and zero for negative inputs. ReLU is significantly reasonable in making a distinction the organize learn complex plans and speeding up the planning process.

Fully Related Layers: Once the highlight maps have been arranged by the convolutional and pooling layers, they are fixed into a 1D vector. This vector is at that point passed through the totally related layers, which perform the final decision-making. The to start with two totally related layers each include of 4096 neurons, applying ReLU actuations to plan the highlights. The final totally related layer has 1000 neurons, comparing to the 1000 conceivable surrender classes, and businesses the softmax incitation work to convey the model’s predictions.

Softmax Layer: The final softmax layer of the VGG19 plan changes the surrender of the last totally related layer into a probability scattering over the 1000 classes. This ensures that the aggregate of all lesson probabilities rises to 1, with the lesson having the most raised probability being chosen as the expected abdicate.

C. EfficientNet B0 Architecture

EfficientNet B0 is used to extract features from each input data point. It is a highly efficient convolutional neural network that analyzes the input data through its convolutional layers. These layers extract essential features such as shapes, textures, or spatial patterns present in the data. Its output is a feature vector that represents the data point in a lower-dimensional space, retaining only the most relevant information for downstream processing.

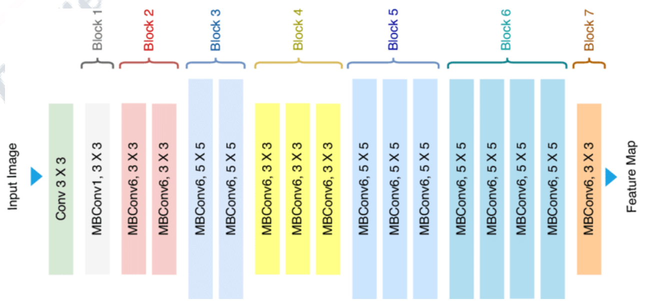


Fig. 3. EfficientNet B0 Architecture

Input Layer: The input layer acknowledges pictures of estimate 224x224x3, where the to begin with two measurements (224x224) speak to the tallness and width of the input picture, and the "3" speaks to the three RGB color channels. This is the normal input estimate for EfficientNet B0 and other picture classification models.

Convolutional Layers: Conv1 (7x7 channels): The to begin with convolutional layer employments 32 channels of estimate 7x7, and this makes a difference capture low-level highlights such as edges and surfaces. A ReLU actuation work is regularly utilized after each convolution to present non-linearity. Be that as it may, EfficientNet B0 employments the Wash actuation work, which performs superior than ReLU in a few cases. Conv2, Conv3, Conv4, etc.: The organize proceeds with dynamically more complex convolutional layers. The number of channels increments with profundity, empowering the arrange to capture progressively theoretical highlights, like shapes and designs. The channel sizes regularly diminish as the arrange deepens.

Depthwise Distinct Convolutions: One of the key developments of EfficientNet is the utilize of depthwise distinct convolutions, which essentially decrease the number of computations compared to standard convolutions. In a depthwise distinct convolution, the convolution operation is part into two stages: a depthwise convolution (applies channels to person channels) taken after by a pointwise convolution (1x1 convolutions to combine the yield of depthwise convolutions). This decreases the number of parameters and computational fetched whereas keeping up performance.

Batch Normalization: Batch normalization is connected after each convolutional operation. This normalizes the actuations inside each mini-batch, making a difference to stabilize the learning handle. It too speeds up joining and makes a difference anticipate overfitting, making the demonstrate more robust.

Activation Work: EfficientNet B0 employments Wash (a variation of ReLU) as the actuation work. Wash has been appeared to beat ReLU in certain errands by permitting the arrange to learn more complex designs and move forward the by and large performance.

Pooling Layer: Instead of utilizing conventional pooling layers like max-pooling, EfficientNet B0 employments Worldwide Normal Pooling (Hole), which midpoints the spatial measurements (tallness and width) of the highlight outline for each channel. This decreases the include outline to a 1D vector, which is at that point passed to the last classification layer. Crevice decreases the spatial measurements whereas holding critical highlights, driving to less parameters and a more compact model.

Fully Associated (FC) Layer: After the pooling layer, the yield is passed through a completely associated (thick) layer that performs classification. In EfficientNet B0, this completely associated layer decreases the dimensionality of the extricated highlights to coordinate the number of yield classes (e.g., 1000 classes for ImageNet classification).

D. LSTM Architecture

LSTM is designed to handle sequential data, making it ideal for tasks where the order and context of elements in a sequence are important. The LSTM network processes each feature vector sequentially, maintaining hidden states that capture the relationships between consecutive data points. This allows the model to understand how each element in the sequence relates to previous and future elements, learning the temporal dependencies of the data.

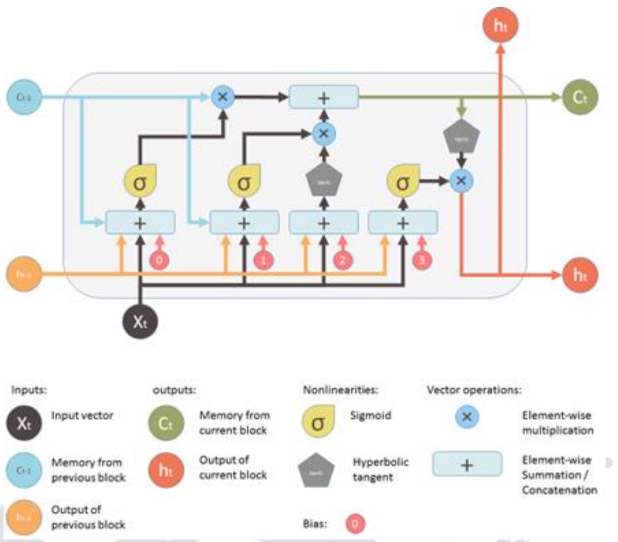


Fig. 4. LSTM Architecture

Input Layer: The input layer gets the arrangement of information focuses at each time step. Each input at a time step is ordinarily a vector that speaks to the current information (e.g., a highlight vector speaking to an picture or a word inserting). The arrangement is handled step by step through the LSTM unit.

LSTM Cell: The center of an LSTM is the LSTM cell, which contains different components that control the stream of data. Each LSTM cell comprises of doors that direct the information passed through the organize and offer assistance it hold long-term conditions. These entryways include:

Forget Gate: The disregard door decides which data from the past cell state ought to be disposed of. It takes the current input and the past covered up state as inputs, passes them through a sigmoid work, and yields a esteem between 0 and 1. This esteem chooses how much of the past memory ought to be "forgotten."

Input Gate: The input entryway controls how much of the current input ought to be included to the memory. It comprises of two parts: A sigmoid layer, which chooses which values to overhaul (yields between 0 and 1). A tanh layer, which makes candidate values that may be included to the state. The values are at that point combined and included to the cell state.

Cell State: The cell state is the memory of the LSTM and speaks to the long-term data that is passed from one time step to the following. The disregard entryway and input door work together to adjust this memory, permitting the show to keep in mind or disregard certain information.

Output Gate: The yield entryway controls what data from the cell state ought to be passed to the following time step. It takes the current input and past covered up state, passes them through a sigmoid work to decide which parts of the cell state

ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)

Vol 12, Issue 01, January 2025

are imperative, and applies a tanh enactment to scale the output.

Hidden State: The covered up state, moreover known as the yield of the LSTM unit, is computed at each time step. The covered up state is upgraded based on the yield door and is passed along with the cell state to the another time step. It contains the data that is pertinent for expectations at that specific time step.

Final Output: After preparing all time steps in the grouping, the LSTM creates an yield. In numerous applications (like sequence-to-sequence assignments), the last covered up state or the grouping of covered up states is passed to ensuing layers (such as a thick layer or softmax layer) for advance preparing or forecast.

E. Indian Sign Language (ISL):

The project was carried out in three specific phases, each intended to tackle increasingly complex aspects of sign language translation, from basic components to more sophisticated constructs. Phase 1 concentrated on translating alphabets (A-Z) and numbers (0-9), utilizing a dataset of 36 classes represented by 62,745 images. This phase established the groundwork by focusing on essential static gestures. Building on this, Phase 2 aimed at translating words, markedly increasing the dataset to 119 classes with a total of 1,84,263 images, enabling the system to recognize a wider and more practical vocabulary. Phase 3 presented the challenge of translating sentences, shifting from static images to dynamic video data. Initially, the dataset consisted of 126 videos covering 21 classes, which were subsequently divided into individual images, resulting in a vast dataset of 2,76,487 segmented images. To improve the robustness and variability of the model, the dataset was enhanced at each phase through techniques such as rotation, flipping, scaling, and brightness adjustment, ensuring better generalization and performance under varied conditions. This phase highlighted dynamic gesture recognition and the contextual understanding required for sentence-level translation, signifying a crucial advancement toward establishing a comprehensive and scalable sign language translation system.

IV. IMPLEMENTATION

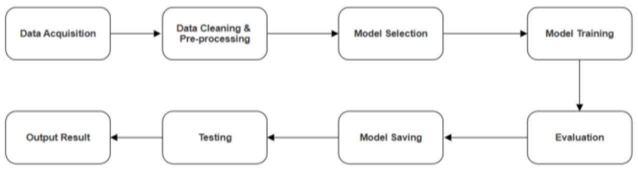


Fig. 5. Block Diagram of Proposed Methodology

The efficacy of machine learning models, particularly in the realm of sign language recognition, hinges significantly on the quality and appropriateness of the datasets employed. This section elucidates the methodologies employed in the collection and preparation of datasets, ensuring their suitability for the various phases of research project.

Data Acquisition and Suitability for Phases: The datasets for this research were meticulously selected based on specific criteria that align with the goals of each phase of the project. The primary sources for these datasets were Kaggle, a renowned platform that hosts a multitude of datasets across different domains, including those pertinent to machine learning tasks such as image and video recognition. The selection process was characterized by a rigorous evaluation of datasets based on the following criteria:

A) Comprehensive Coverage: The datasets needed to encompass a wide range of gestures representative of Indian Sign Language (ISL). This was crucial to ensure the models could generalize well across various signing styles and contexts.

B) Phase-Specific Requirements: For alphabets and number foundational phase, labeled datasets containing static images of each letter and digit were utilized. The focus was on ensuring clarity and visibility of signs to facilitate accurate recognition by the model. In word phase employed a labeled image dataset specifically designed to capture static gestures corresponding to individual words in Indian Sign Language (ISL). Each image in the dataset was meticulously labeled to accurately represent the gestures associated with specific words. The selection process prioritized high-resolution images, ensuring that the details of the hand signs were clearly visible. The final sentence phase necessitated the use of longer video sequences that depicted complete sentences in ISL. This phase aimed to teach the model not just to recognize individual signs but to understand the context and flow of language.

Data Fragmentation and Augmentation: To optimize the utility of the acquired datasets, a combination of data fragmentation and augmentation techniques was implemented. These strategies were designed to enhance the model's learning capabilities while ensuring computational efficiency.

A) Data Fragmentation: Data fragmentation involved segmenting the datasets into smaller, manageable subsets tailored for specific training objectives. This fragmentation allowed for targeted training and validation processes. The datasets were divided based on the recognition phase, ensuring that the training data for alphabets, words, and sentences were distinctly categorized. This structure enabled focused learning, allowing the model to specialize in recognizing distinct elements of sign language. When dividing the datasets into training, validation, and testing sets, stratification was employed to ensure that each subset maintained the same distribution of classes. This practice is essential to avoid any biases that may arise from an imbalanced dataset.

B) Data Augmentation: Data augmentation techniques were employed to artificially increase the size of the training datasets, thereby improving the model's ability to generalize from the training data to unseen data. Images and video frames underwent random transformations, including rotations, translations, and flips. This variability helped the model learn to recognize signs from different orientations and perspectives. For the video datasets, temporal augmentation techniques were implemented, such as frame skipping (selecting every nth frame) and temporal shifting (adding or removing frames from the sequence). These techniques enabled the model to learn from variations in signing speed and motion.

Data Preprocessing: Data preprocessing is a critical step in preparing the datasets for model training. This phase ensures that the data is in an appropriate format and scale for effective learning. The following preprocessing techniques were applied:

A) Normalization: All images and video frames were resized to a consistent dimension, ensuring uniformity across the dataset. Pixel values were normalized to a range of 0 to 1, facilitating faster convergence during training.

B) Label Encoding: Each sign, represented by either an alphabet, number, or word, was encoded into a numerical format. This transformation was vital for enabling the model to interpret categorical data effectively.

C) Data Splitting: The datasets were divided into training, validation, and test sets using a standard split ratio. Typically, 70 percent of the data was allocated for training, 15 percent for validation, and 15 percent for testing. This division is essential for assessing the model's performance and ensuring that the model is not evaluated on data it has seen during training.

The selection and training of machine learning models are pivotal in achieving high accuracy and efficiency in sign language recognition. This section outlines the strategies employed in model selection, training, and evaluation across the different phases of the project.

Phase 1: Alphabets and Numbers: The first phase concentrated on the recognition of individual alphabets and numbers in Indian Sign Language (ISL). A Convolutional Neural Network (CNN) architecture was deemed suitable for this task due to its proficiency in image classification.

A) Model Architecture: In this phase, two prominent CNN architectures, AlexNet and VGG19, were employed to evaluate their performance on the recognition task:

AlexNet: This architecture effectively captured spatial hierarchies in the images, enabling the model to extract essential features representing the shapes of the alphabets and numbers.

VGG19: Knownfor its depth and use of small convolutional filters, VGG19 provided a robust framework for recognizing more complex features in the signs, facilitating improved accuracy.

Both models were trained on a labeled dataset of static images, allowing for a comparative analysis of their performance. The outputs from each model were evaluated, providing insights into their strengths and weaknesses.

B) Training Process: The model training involved a structured approach, including the following key components

Loss Function: A categorical cross-entropy loss function was employed, suitable for multi-class classification problems, allowing for effective optimization of the models.

Optimizer: The Adam optimizer was selected for its efficiency in dynamically adjusting learning rates, aiding the models in converging effectively.

Batch Training: The training was conducted in batches to enhance computational efficiency and manage memory usage effectively.

Early Stopping: An early stopping criterion was implemented to monitor validation loss during training. If the validation loss did not improve for a predetermined number of epochs, training was halted to prevent overfitting.

This phase established a solid foundation for advancing to the next level of complexity, demonstrating the effectiveness of both AlexNet and VGG19 in recognizing alphabets and numbers.

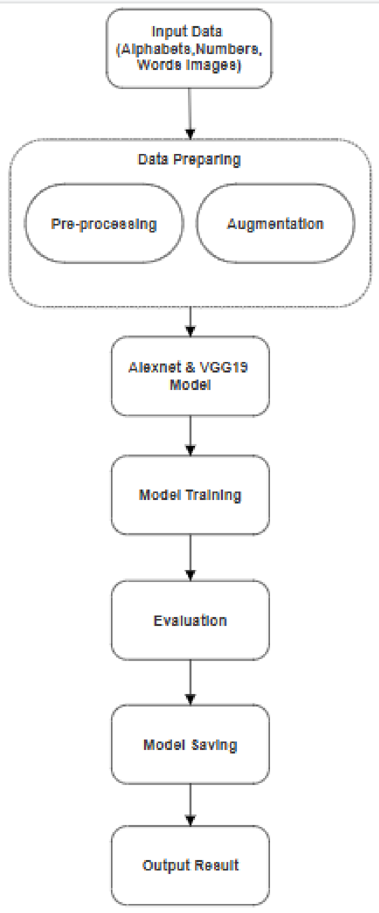


Fig. 6. Block Diagram of Phase 1 and 2

Phase 2: Word Recognition: With the basic recognition of alphabets and numbers successfully established, the second phase aimed at recognizing complete words in ISL. This phase required a model capable of interpreting dynamic gestures, which was accomplished using AlexNet exclusively.

A) Model Architecture: The architecture employed in this phase utilized:

AlexNet: The model focused on spatial feature extraction from labeled images representing specific words. Its deep structure was effective in capturing the nuances of the gestures associated with individual words.

B) Training Process: The training process for this phase involved:

Input Preparation: The labeled dataset consisted of images that clearly depicted individual signs corresponding to various words. This approach ensured that the model could learn from distinct examples.

C) Performance Evaluation: The model's performance was evaluated using precision, recall, and F1-score metrics, providing a comprehensive understanding of its effectiveness in recognizing words in ISL. The results indicated significant improvements in recognition accuracy compared to the previous phase. Through careful tuning and training, the model demonstrated a strong capability in recognizing words in ISL, leveraging the strengths of the AlexNet architecture.

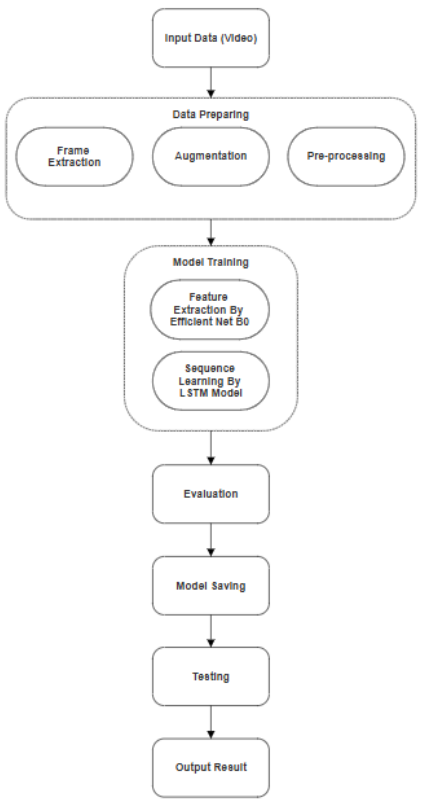


Fig. 7. Block Diagram of Phase 3

Phase 3: Sentence-Level Recognition: The final phase focused on recognizing complete sentences in ISL, an intricate task requiring a deeper understanding of language context and flow.

A) Model Architecture: For this phase, a more sophisticated architecture was implemented, featuring:

Hybrid Model of LSTM and EfficientNet B0: The integration of EfficientNet B0 for spatial feature extraction and LSTM for temporal modeling allowed the model to analyze the spatial and sequential aspects of signing simultaneously.

EfficientNet B0: This model enhanced the feature extraction process through its efficiency in handling various input resolutions while maintaining high accuracy.

LSTM: This architecture was critical for capturing the temporal dynamics of the signing gestures, enabling the model to remember previous states and learn dependencies between signs.

B) Training Process: Training in this phase was conducted on sequences of video frames, focusing on the nuances of signing duration and fluidity. Key elements of the training process included:

Complex Data Handling: The model was trained on a comprehensive dataset that included variations in sentence structure, allowing it to learn from a wide array of examples.

Continuous Monitoring: Throughout the training, the model's performance was continuously monitored, with adjustments made as necessary to improve accuracy and reduce loss. The successful implementation of this phase culminated in a model capable of recognizing complete sentences in ISL, paving the way for practical applications in real-time sign language translation.

Training and Early Stopping: Across all phases of training, early stopping was a pivotal strategy utilized to enhance model performance and prevent overfitting. By continuously monitoring validation loss, training could be halted at optimal points, ensuring the model retained its ability to generalize to new data. This comprehensive approach to model selection and training facilitated the effective recognition of alphabets, words, and sentences in Indian Sign Language.

V. EXPERIMENTAL RESULTS

In three stages, the "SignSpectra: Sign Language Translator" was thoroughly evaluated, with each phase focusing on distinct linguistic difficulties with Indian Sign Language (ISL). This section offers a thorough performance analysis of the system, demonstrating its scalability and resilience with the help of statistical and visual insights.

Phase 1: Alphabets and Numbers using AlexNet

ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)

Vol 12, Issue 01, January 2025

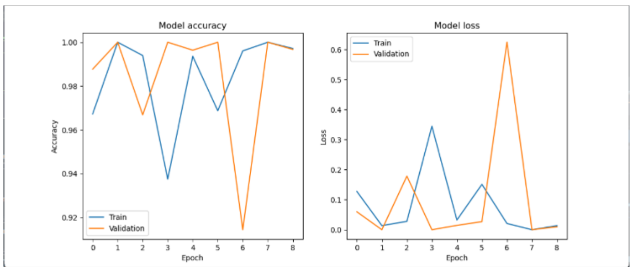


Fig. 8. Without Early-Stopping using AlexNet for Phase-1

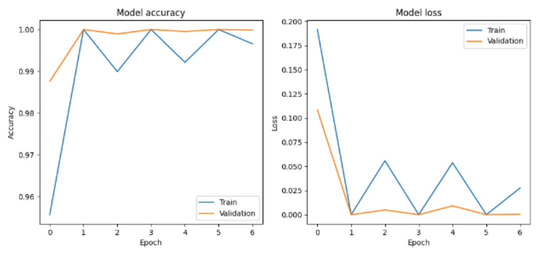
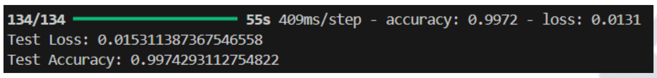
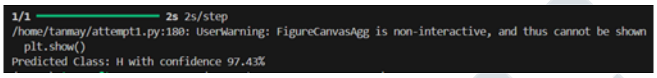


Fig. 9. With Early-Stopping using AlexNet for Phase-1



```
134/134 ━━━━━━━━━━━━━━━━━━━━ 55s 409ms/step - accuracy: 0.9972 - loss: 0.0131
Test Loss: 0.015311387367546558
Test Accuracy: 0.9974293112754822
```

Fig. 10. Test Accuracy and Loss using AlexNet for Phase-1



```
1/1 ━━━━━━━━━━━━━━━━━━━━ 2s 2s/step
/home/tanmay/attempt1.py:180: UserWarning: FigureCanvasAgg is non-interactive, and thus cannot be shown
  plt.show()
Predicted Class: H with confidence 97.43%
```

Fig. 11. Test with input image using AlexNet for Phase-1

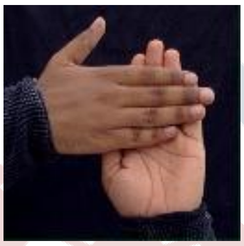


Fig. 12. Input Image for Phase-1 using AlexNet

Evaluation: During this phase, the English alphabet (A-Z) and digits (0-9) were among the 36 different classes that were recognized. To assess how well the system classified static signs, two models—AlexNet and VGG19—were used. To ensure balanced learning across all categories, the dataset's 62,745 photos were divided into training (70%), validation (15%), and testing (15%) sets. With modest testing and training losses of 0.015 and 0.0131, respectively, AlexNet showed excellent performance, with testing accuracy of 99.74% and training accuracy of 99.72%. Within the first 6 epochs, the accuracy graph in shows quick convergence and stabilizes at about 100%. Strong generalization abilities are demonstrated by the loss graph in, which shows a smooth fall with little difference between training and validation losses. As seen in, for example, the model successfully identified the letter "H" with 97.43% confidence, demonstrating its capacity to properly capture spatial hierarchies.

Phase 1: Alphabets and Numbers using VGG19

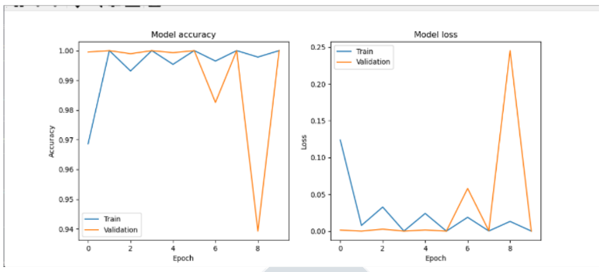


Fig. 13. Without Early-Stopping using VGG19 for Phase-1

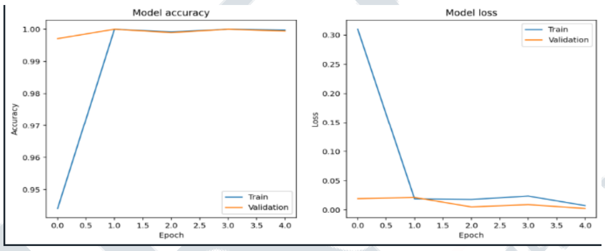
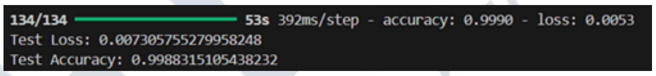
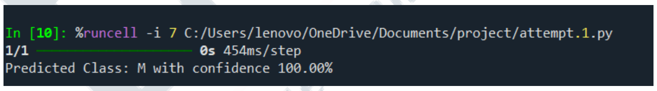


Fig. 14. With Early-Stopping using VGG19 for Phase-1



```
134/134 ━━━━━━━━━━━━━━━━━━━━ 53s 392ms/step - accuracy: 0.9990 - loss: 0.0053
Test Loss: 0.007305755279958248
Test Accuracy: 0.9988315105438232
```

Fig. 15. Test Accuracy and Loss using VGG19 for Phase-1



```
In [10]: %runcell -i 7 C:/Users/lenovo/OneDrive/Documents/project/attempt.1.py
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 454ms/step
Predicted Class: M with confidence 100.00%
```

Fig. 16. Test with input image using VGG19 for Phase-1

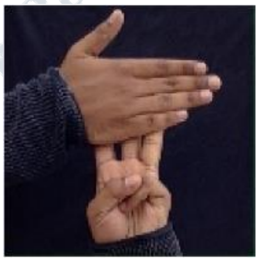


Fig. 17. Input Image for Phase-1 using VGG19

Evaluation: During this phase, the English alphabet (A-Z) and digits (0-9) were among the 36 different classes that were recognized. To assess how well the system classified static signs, two models—AlexNet and VGG19—were used. To ensure balanced learning across all categories, the dataset's 62,745 photos were divided into training (70%), validation (15%), and testing (15%) sets. By utilizing its deeper architecture, VGG19 also surpassed AlexNet, with testing accuracy of 99.88% and training accuracy of 99.90% with testing and training losses of 0.007 and 0.0053, respectively. The below figures illustrate the trends in accuracy and loss, respectively, and both show impressive consistency, with validation performance closely resembling training performance. The accurate categorization of the letter "M" with 100% confidence, as illustrated in, demonstrates how the model's capacity to extract finer image information greatly increased its precision. Overall, VGG19 outperformed

AlexNet on this job thanks to its deeper layers, demonstrating its aptitude for highly accurate static sign recognition.

Phase 2: Words using AlexNet

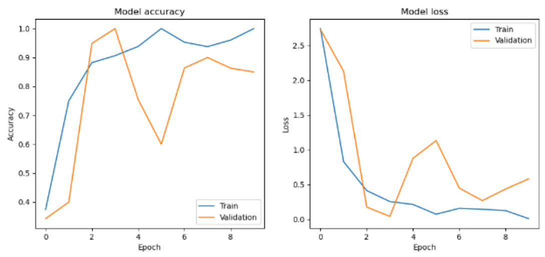


Fig. 18. Without Early-Stopping using AlexNet for Phase-2

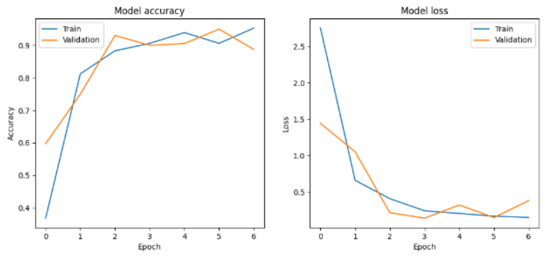
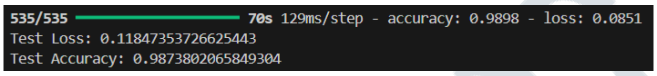
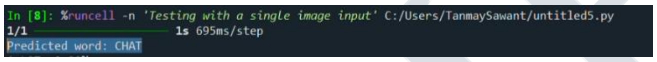


Fig. 19. With Early-Stopping using AlexNet for Phase-2



```
535/535 ━━━━━━━━━━━━━━━━━━━━ 70s 129ms/step - accuracy: 0.9898 - loss: 0.0851
Test Loss: 0.11847353726625443
Test Accuracy: 0.9873802065849304
```

Fig. 20. Test Accuracy and Loss using AlexNet for Phase-2



```
In [8]: %runcell -n 'Testing with a single image input' C:/Users/TanmaySawant/untitled5.py
1/1 ━━━━━━━━━━━━━━━━━━━━ 1s 695ms/step
Predicted word: CHAT
```

Fig. 21. Test with input image using AlexNet for Phase-2

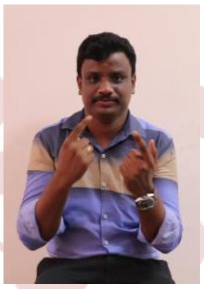


Fig. 22. Input Image for Phase-2 using AlexNet

Evaluation: In the second stage, word-level classification was the main focus, and AlexNet was the only tool used to assess how well it could adjust to changing gestures. The 1,84,263 photos in the dataset were split into three sets: 70% for training, 15% for validation, and 15% for testing. The dataset was made more diverse by include differences in hand location, lighting, and other real-world difficulties. AlexNet demonstrated strong performance in spite of these difficulties, attaining a testing accuracy of 98.73% and a training accuracy of 98.98%. With a slight variation in later epochs, perhaps due to dataset variability, the training and validation loss values were found to be 0.0851 and 0.118, respectively. The accuracy trends for this phase, which demonstrate a consistent improvement with validation accuracy closely following training accuracy. The loss graph highlights the model's ability to generalize well in spite of the dataset's complexity. As seen below, the model was able to correctly identify the word "Chat," proving its applicability for word-level identification. The outcomes of this stage demonstrate how well the model can handle static word-level gestures, even in difficult situations.

Phase 3: Sentences using EfficientNet B0 and LSTM

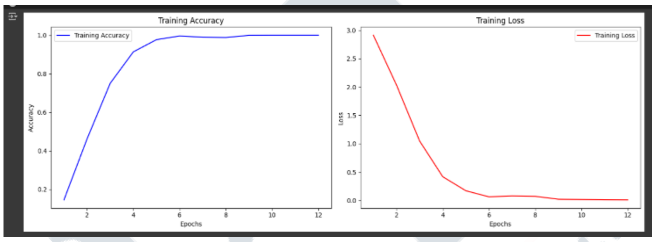
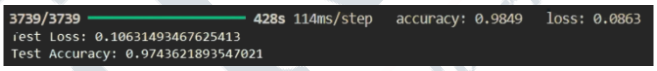
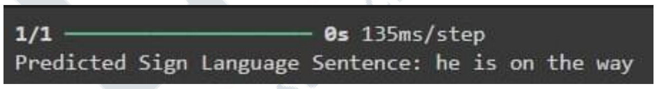


Fig. 23. With Early Stopping using EfficientNet B0 and LSTM



```
3739/3739 ━━━━━━━━━━━━━━━━━━━━ 428s 114ms/step   accuracy: 0.9849   loss: 0.0863
Test Loss: 0.10631493467625413
Test Accuracy: 0.9743621893547021
```

Fig. 24. Test Accuracy and Loss using EfficientNet B0 and LSTM



```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 135ms/step
Predicted Sign Language Sentence: he is on the way
```

Fig. 25. Input video using EfficientNet B0 and LSTM

Evaluation: The last stage combined LSTM for temporal modeling with EfficientNet B0 for spatial feature extraction to tackle the difficulty of sentence-level translation. A dataset of 2,76,487 video frames, divided into training (70%), validation (15%), and testing (15%) sets, was used to assess this hybrid architecture. The model has to understand both the temporal and spatial elements of signing motions since the dataset contained sequences with a variety of syntax and semantics. The hybrid model recorded training and testing losses of 0.0863 and 0.106, respectively, and attained training accuracy of 98.49% and testing accuracy of 97.43%. The accuracy trend, which shows a steady increase and demonstrates how well the model can learn temporal relationships. The linguistic complexity of the dataset is responsible for the little variations in validation loss, while the loss graph exhibits steady convergence. As seen, the system's contextual awareness and sequential data processing skills were put to the test when it correctly translated the line "He is on the way" from a sample video. This stage demonstrates the model's ability to manage complex sentence-level translations, which makes it a useful tool for ISL interpretation in real time.

Comparative Perspectives: The models' remarkable accuracy and low loss throughout all stages confirmed their efficacy in handling a range of ISL identification tasks. With

IFERP
Explore Your Research Journey...
ISSN (Online) 2394-6849
International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)
Vol 12, Issue 01, January 2025

each model tailored to its specific role, the graph analyses offer further insights about the system's resilience. While the EfficientNet B0 and LSTM hybrid model performed very well when handling sequential phrase patterns, AlexNet and VGG19 shown efficacy in static gesture detection.

Comparative Performance Table:

Phase	Model	Training Accuracy	Testing Accuracy	Training Loss	Testing Loss
1	AlexNet	99.72%	99.74%	0.0131	0.015
1	VGG19	99.90%	99.88%	0.0053	0.007
2	AlexNet	98.98%	98.73%	0.0851	0.118
3	Efficient Net B0 + LSTM	98.49%	97.43%	0.0863	0.106

Table. 1. Comparative Performance

VI. CONCLUSION

The SignSpectra project has successfully developed a robust AI-driven system for translating Indian Sign Language (ISL) into spoken English, marking a significant step toward inclusive communication for the deaf/mute community. The project's phased approach, spanning alphabets, words, and sentences, provided a progressive structure for handling linguistic complexities at multiple levels. Through the use of advanced architectures AlexNet, VGG19, EfficientNet B0, and LSTM the system demonstrated high accuracy and contextual understanding, with effective strategies such as early stopping to mitigate overfitting. Notably, in Phase 3, the model's sentence-level accuracy underscored its capability in interpreting and translating complex expressions, as reflected in the sample testing of phrases like "He is on the way." With final training accuracy at 98.49 percent and testing accuracy at 97.43 percent, SignSpectra is positioned as a highly accurate and reliable tool for real-time ISL translation. By enabling seamless interactions across diverse settings, this project not only advances accessibility in India but also sets a foundation for future expansions in sign language translation, helping to bridge societal communication gaps and fostering an inclusive future for all.

Future Scope

- Expand to support dynamic and regional sign languages for broader communication coverage.
- Adding real time feature capturing
- Integrate multi-modal inputs like facial expressions and body movements for enhanced accuracy.
- Develop multilingual capabilities to translate sign language into various global languages.
- Utilize lightweight models for deployment on mobile and portable devices.

REFERENCES

[1] D. R. Kothadiya, C. M. Bhatt, T. Saba, A. Rehman and S. A. Bahaj, "SIGN FORMER: DeepVision Transformer for Sign Language Recognition," in IEEE Access, vol. 11, pp. 4730-4739, 2023. https://doi.org/10.1109/ACCESS.2022.3231130

[2] A. Rasheed, Farsana K. A., M. Z. Ikbal, N. Rajesh, and M. Shefeek, "Sign Language Translator," International Journal of Scientific Development and Research (IJSDR), Vol. 8, Issue 5, pp. 1857-1861, 2023.

[3] J. P. Sahoo, A. J. Prakash, P. Pławiak, and S. Samantray, "Real-Time Hand Gesture Recognition Using Fine-Tuned Convolutional Neural Network," Sensors 2022, 22, 706, 2022. https://doi.org/10.3390/s22030706.

[4] D. S, K. H. K B, A. M, S. M, D. S and K. V, "An Efficient Approach for Interpretation of Indian Sign Language using Machine Learning," 2021 3rd International Conference on Signal Processing and Communication (ICPSC), Coimbatore, India, pp. 130-133 2023, 2021. https://doi.org/10.1109/ICSPC51351.2021.9451692

[5] Muthu M. H. and Gomathi V., "Indian Sign Language Recognition through Hybrid ConvNet-LSTM Networks," EMITTER International Journal of Engineering Technology, Vol. 9, No. 1, pp. 182-203, 2021. https://doi.org/10.24003/emitter.v9i1.613

[6] A. Chavan, J. Ghorpade-Aher, A. Bhat, A. Raj and S. Mishra, "Interpretation of Hand Spelled Banking Helpdesk Terms for Deaf and Dumb Using Deep Learning," 2021 IEEE Pune Section International Conference (PuneCon), Pune, India, pp. 1-5, 2021. https://doi.org/10.1109/PuneCon52575.2021.9686514

[7] G. Jayadeep, Vishnupriya N. V., V. Venugopal, Vishnu S., and Geetha M., "Mudra: Convolutional Neural Network based Indian Sign Language Translator for Banks," 2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS), Madurai, India, pp. 1228-1232, 2020. https://doi.org/10.1109/ICICCS48265.2020.9121144

[8] A. Sharma, S. Panda and S. Verma, "Sign Language to Speech Translation," 2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT), Kharagpur, India, pp. 1-8, 2020. https://doi.org/10.1109/ICCCNT49239.2020.9225422