Inteligencia Artificial 27(74), 181-213
doi: 10.4114/intartif.vol27iss74pp181-213

INTELIGENCIA ARTIFICIAL

http://journal.iberamia.org/

Building an Integrative System: Enriching Gesture Language Video Recognition through the Multi-Stream of Hybrid and Improved Deep Learning Models with an Adaptive Decision System

Anwar Mira[1,2], Olaf Hellwich[1]
[1] Technische Universität Berlin, Computer Vision & Remote Sensing, Germany
[2] University of Babylon, College of Information Technology, Iraq
anwar.jaafar@uobabylon.edu.iq

Abstract. The recognition of hand gestures is of growing importance in developing human-machine interfaces that rely on hand motions for communication. However, recognizing hand gesture motions poses challenges due to overlapping gestures from different categories that share similar hand poses. Temporal information has proven to be more effective in distinguishing sequences of hand gestures. To address these challenges, this research presents an innovative adaptive decision-making system that aim to enhance gesture recognition within the identical category have been introduced. The system capitalizes on the potential for variations in recognition outcomes derived from a diverse model of time-sharing neural networks, each employing different neural networks and trained on distinct input features. By incorporating such diverse input features, the system significantly boosts the robustness of recognition decisions, enabling it to effectively capture even the most subtle disparities within internal video representations. To achieve our research objective, we extensively investigate deep convolutional neural networks specifically trained on videos for hand gesture recognition. We also incorporate enhanced features from deep CNN using standard neural networks, namely Self Organizing Network and Radial Basis Function Network. By combining these features in various configurations, we develop novel frame-wise features based on the enhanced CNN features. These frame-wise features enable the training of diverse sets of recurrent neural network models, resulting in novel ensembles of composite models derived from various recurrent neural networks with diverse configurations. Some models are trained using multiple streams, while others utilize a single stream. To ensure the effective integration of these models, we implement a novel adaptive decision system mechanism that improves performance for weak prediction models and enhances overall recognition capability by taking a collective prediction decision. Experimental results demonstrate the significance of each proposed recurrent neural network model and the effectiveness of the new frame-wise features in enabling accurate decisions. This research achieves state-of-the-art performance in hand gesture recognition, highlighting the potential of combining different neural network architectures and feature representations to achieve superior outcomes.

Keywords: Recurrent Neural Networks, Self Organizing Network, Radial Basis Function Network, Video Gesture Recognition, Adaptive Decision System

1 Introduction

System integration is a vital aspect of machine learning that combines multiple models or algorithms to enhance prediction accuracy. Various approaches have been developed for this purpose. One such approach is the Generalized Weighted Ensemble with Internally Tuned Hyperparameters (GEM-ITH), which is a weighted averaging stacking method introduced in [1]. GEM-ITH optimizes ensemble weight optimization by tuning the hyperparameters of each base learner, resulting in improved performance and accurate capture of the underlying data distribution. The optimization process is accelerated through Bayesian search, and a heuristic generates diverse and high-performing base learners.

Ensemble methods and stacking classifiers are also widely used in medical image analysis, particularly in the diagnosis of pediatric pneumonia using chest X-rays [2]. In this application, a proposed approach incorporates contrast-limited Adaptive Histogram Equalization for image enhancement. Deep learning-based features extracted from models such as MobileNet, DenseNet121, DenseNet169, and DenseNet201 are concatenated and fed into a stacked ensemble classifier, leading to accurate classification.

Another effective integration technique is stacking, where predictions from multiple models serve as input features for a meta-model that makes the final prediction. This technique has shown promise in precise Breast Cancer (BC) diagnosis [3]. StackBC, a deep learning-based stacking method, combines Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM), and Gated Recurrent Unit (GRU) models to classify Invasive Ductal Carcinoma (IDC), achieving improved predictive outputs through the stacking technique. Ensemble learning and stacking frameworks have also been employed to enhance wind power forecasting accuracy and stability [4]. An ensemble model is constructed by pre-processing wind power data using various decomposition techniques and selecting optimal methods. The first-layer forecasting model consists of base learners with low correlation and strong predictive ability, while a higher-order neural network (HNN) serves as the second-layer prediction model.

Hierarchical approaches provide an alternative method for system integration by assigning different models to distinct prediction levels. This hierarchical structure enables specialized tasks such as low-level feature extraction, object classification, and decision-making based on object classifications. Researchers have proposed various hierarchical approaches to address specific challenges. For example, the Hierarchical Scale-Aware Vision-Language Transformer (HSVLT) introduced in [5] recognizes multiple objects within an image. HSVLT incorporates a hierarchical multi-scale architecture and a Cross-Scale Aggregation module, enabling the recognition of objects with varying sizes and appearances. Similarly, an end-to-end trainable hierarchical deep learning model is proposed in [6] for autonomously localizing putty terminal points in the wall putty scraping process using construction robots. This model combines visual attention and the inception module to enhance feature extraction capabilities through multi-scale attention feature fusion.

Fusion techniques play a crucial role in system integration by combining information from different sources or modalities to improve predictions. In multi-modal computer vision tasks, fusion techniques facilitate the fusion of features from various sensors or modalities, such as images, depth maps, or audio. Fusion can be achieved through techniques like early fusion, which combines features at the input level, or late fusion, which combines predictions at the output level. For instance, the dual-decoding hierarchical fusion network (DHFNet) described in [7] effectively utilizes RGB and thermal data in multi-modal computer vision tasks. DHFNet incorporates a two-layer decoder with boundary refinement and boundary-guided foreground/background enhancement modules to preserve more information during decoding. It also includes an adaptive attention-filtering fusion module to extract complementary information from RGB and thermal modalities, a graph convolutional network, and an atrous spatial pyramid pooling module to capture multiscale features and enhance semantic information.

In the context of time series forecasting [8], a reinforcement learning-based model combination (RLMC) framework is proposed to address the challenge of accurate modeling and forecasting of time series data. RLMC treats model selection as a sequential decision-making problem and learns a deterministic policy through reinforcement learning to output dynamic model weights suitable for non-stationary time series data. Deep learning is incorporated to extract hidden features from raw time series data, enabling quick adaptation to changing data distributions.

Additionally, self-supervised representation learning algorithms like Predictions of Bootstrapped Latents (PBL) [9] capture structured information about environment dynamics by utilizing multistep predictive representations of future observations. PBL trains the representation by predicting latent embeddings of future observations, facilitating the learning of crucial aspects of the environment dynamics. PBL offers flexibility by defining prediction tasks in the latent space and supporting the use of multimodal observations. Furthermore, a dynamic ensemble wind speed prediction model based on deep reinforcement learning is proposed in [10]. This model considers the time-varying characteristics of wind speed series and incorporates ensemble learning, multi-objective optimization, and deep reinforcement learning to ensure effectiveness.

The choice of integration method depends on the specific problem and the characteristics of the models involved. Evaluating and comparing different integration techniques is important to determine the most effective approach for a given computer vision task.

2 Related Works

Recognizing hand gestures accurately is often challenging due to the similarities in hand poses observed within video gestures, leading to overlapping. Moreover, variations in movement speed and the importance of sequential events further compound the difficulties in video recognition. Addressing these challenges necessitates the recognition of variations inherent in hand gestures. To address these challenges, researchers have explored various approaches in hand gesture recognition, leveraging the integration between machine learning techniques and considering the temporal aspects of gesture sequences.

In the work by [11], the focus is on dynamic gesture recognition in human-machine interaction. The proposed model consists of two sub-networks: a transformer and an ordered-neuron long-short-term memory (ON-LSTM) based recurrent neural network (RNN). These sub-networks are trained using skeleton joint information to recognize gestures. The distinct architectures of the sub-networks enable knowledge sharing between them. Through knowledge distillation, the features and predictions from each sub-network are fused into a new fusion classifier. Additionally, a cyclical learning rate is employed to generate an ensemble of models, enhancing generalizability, and improving prediction performance.

The paper by [12] proposes an ensemble of CNN-based approaches for hand gesture recognition. The process involves gesture detection using background separation, contour extraction, and hand region segmentation. The images are then resized and fed into three individual CNN models, trained in parallel. The output scores of the CNN models are averaged to create an optimal ensemble model for the final prediction. In [13], a deep learning model is proposed for accurate surface electromyography (EMG) decoding of hand gestures in human-computer interaction. The model incorporates attention mechanisms and transfer learning. It includes a feature extractor, label classifier, and gesture estimator. Attention modules enhance feature extraction by emphasizing important information and transfer learning selectively transfers parameters from a pre-trained model, improving performance. The proposed model outperforms baseline models in terms of estimation accuracy on the Myo and NinaPro datasets, even with limited retraining data.

Another study [14] focuses on improving the control of wearable mechatronic devices used in assisted therapy by developing a user-independent gesture classification method. Through sensor fusion, the method combines electromyography (EMG) data and inertial measurement unit (IMU) data. The study involved healthy participants wearing the Myo Armband and performing seven gestures in different arm positions. Three classification methods were employed to achieve accurate gesture recognition.

Skeleton-based action recognition is explored in [15], simultaneously addressing temporal and spatial aspects. Different representations of skeletal data are proposed and evaluated, incorporating attention mechanisms to enhance training efficiency. The proposed representations demonstrate comparable or improved accuracy compared to previous work on UT-Kinect and Kinect Activity Recognition Dataset (KARD) datasets, highlighting their practicality and effectiveness for action recognition.

The work by [16] introduces SDViT, a stacked distilled vision transformer model for hand gesture recognition (HGR). SDViT addresses similar hand gestures, real-time performance, and model generalization challenges. It utilizes a pre-trained vision transformer (ViT) with self-attention to capture intricate connections among image patches. Knowledge distillation compresses the ViT model and enhances generalization. Multiple distilled ViTs are stacked to improve predictive performance and reduce overfitting.

MoVNect, presented in [17], is a lightweight deep neural network designed for 3D human pose estimation using a single RGB camera. The model's performance is improved by applying knowledge distillation through teacher-student learning. Real-time post-processing ensures temporally stable 3D skeletal information, enabling direct application in various scenarios. The authors demonstrate the network's capabilities by implementing a real-time 3D avatar application on mobile devices, showcasing high accuracy and fast inference time.
Moreover, the study [18] demonstrates the effectiveness of a late fusion approach in sign language recognition. Two deep neural networks optimized for vision and Leap Motion data are fused, resulting in improved performance compared to individual approaches. The multimodal approach outperforms single-sensor methods when applied to unseen data. Transfer learning with British Sign Language weights further enhances the models' performance in classifying American Sign Language, with the transfer learning multimodality approach achieving the highest accuracy.
These related works collectively contribute to the advancements in hand gesture and motion interaction in machine learning, providing insights and techniques for improved recognition and application in various domains.
The research presented in this study builds upon the work by [19] and introduces several significant contributions that advance the field of hand gesture recognition. It is important to discuss the importance of these contributions to the current state-of-the-art methods and highlight the significance of detecting and identifying gestures.
Firstly, this study introduces novel training pipelines for different modalities of neural networks, aiming to improve predictability. By developing specific training approaches of different modalities, such as single-stream and multi-stream of time-shared recurrent neural networks, to address the challenges of accurately recognizing hand gestures in diverse input types. This contribution is crucial in improving the performance of hand gesture recognition systems, as it optimizes the training process for each modality and enhances the overall predictability of the networks.
Secondly, the research focuses on the extraction of diverse new features with varying performance levels. By identifying and incorporating a wide range of features that capture different aspects of hand gestures, the study enhances the discriminative power of the models. These diverse features play a crucial role in improving the recognition performance of both proposed models. This contribution enables a more comprehensive and robust representation of hand gestures, leading to more accurate and reliable recognition. Furthermore, the study proposes an adaptive decision system that effectively enhances the accuracy of predictions while reducing execution time. By intelligently combining the outputs of various training modality pipelines, the research optimizes hand gesture recognition performance. This contribution is particularly valuable in real-time applications, where quick and accurate gesture recognition is essential for efficient human-computer interaction.
In light of the current state-of-the-art methods, the contributions of this research significantly advance the field of hand gesture recognition. The novel training pipelines, diverse feature extraction, and adaptive decision systems collectively contribute to improving the recognition and predictability of neural networks. By addressing the challenges related to different modalities, enhancing the feature representation, and optimizing decision-making, the proposed methods push the boundaries of gesture recognition technology.
It is crucial to recognize the substantial implications of gesture detection and identification across various domains. Hand gestures serve as a natural and intuitive form of communication, particularly in sign language learning and human-computer interaction. The accurate and efficient recognition of gestures plays a pivotal role in facilitating effective communication among individuals with different communication abilities, enhancing accessibility in technology interfaces, and unlocking new opportunities for interactive applications. As a result, the contributions of this research hold significant value in advancing gesture recognition techniques, benefiting fields such as education, accessibility, and human-computer interaction.
However, it is important to consider that the implementation of diverse neural network architectures and ensembles of models may introduce computational complexity. This complexity can potentially impact real-time performance and practical feasibility, particularly in resource-constrained environments or on low-power devices. Nevertheless,

the effectiveness of the proposed pipeline on the hand gesture video dataset has been empirically demonstrated, specifically in the context of sign language learning.
The effectiveness of the new pipeline on the hand gesture video dataset has been proven for sign language learning. The model structure proposed in this study encompasses four distinct types of neural networks, which are extensively categorized and elucidated in Section (3). Section (4) provides a comprehensive overview of the research methodology, including a detailed explanation of the proposed models and their precise implementation mechanisms. Furthermore, Section (5) is dedicated to presenting the results obtained from the implementation of the proposed work.

3 Neural Networks

Neural networks are a subset of machine learning that involve learning techniques such as classification and regression. They offer a variety of algorithms and designs that can be used for supervised and unsupervised learning, among other things. In our proposed work, we have employed several neural network algorithms, and we will provide a detailed description related to the working mechanism with videos of each one that has been utilized in this study.

3.1 Convolutional Neural Network CNN

Deep learning networks, particularly Convolutional Neural Networks (CNNs), have gained significant prominence in recent years as a formidable approach to neural network modeling. CNNs consist of multiple layers, each serving a specific purpose in the classification process. Convolutional layers utilize filters of various lengths to extract essential features from the input while pooling layers reduce the dimensionality of the input pattern. Non-linear layers are strategically interspersed to create a deep network that can be trained in a feed-forward manner. This training optimizes the weights for each layer, enhancing the network's performance. To train CNNs, a gradient descent algorithm is employed with a constant learning rate (R) and descent momentum value (Mom), as depicted in equation (1). This iterative process allows for the optimization of weights in each layer, enabling effective video image classification in the proposed work.

$$W_L=W_{L-1}+R_L*Mom_L \quad (1)$$

Through extensive research and experimentation, CNNs have demonstrated their potential in various domains. Their ability to handle large-sized images during training and mitigate overfitting issues sets them apart from other neural networks. This makes CNNs particularly suitable for identifying video images of hand gestures and, thus, their feature extraction, as they offer superior performance and robustness.

In our study, we employed the video image features of hand gestures extracted via CNN training as essential inputs in training other neural networks. The primary objective behind this approach was to enhance recognition performance significantly. Subsequently, we will delve into a comprehensive exploration of these networks in the subsequent sections.

3.2 Radial Basis Function Neural Network

In the domain of neural networks, the Radial Basis Function (RBF) network stands out as a specific type of feed-forward neural network. The RBF network extends beyond the input layer, incorporating two additional layers - a hidden layer and an output layer. The hidden layer in an RBFN plays a pivotal role in clustering and feature extraction by adjusting its parameters, particularly the center points of the radial basis functions, to effectively capture the data distribution. Meanwhile, the output layer performs a linear transformation of the hidden layer activations to produce the final output [20]. Within the hidden layer, each neuron calculates its sigma (σ) value using expression (2). This value represents the average distance between the training set F, consisting of m video image features from a specific category, and the center μ. To achieve optimal clustering, K-means is employed to cluster the training image features around the most suitable image feature center μ. Expression (3) reveals the computation of Beta (β) for each hidden neuron. Beta determines the extent of influence of the activation function based on the calculated sigma value. Expression (4) demonstrates how the activation function Φ is determined for each input neuron that quantifies the similarity between an input image feature and the feature vector of cluster centers across all categories. The activation function leverages an exponential function to measure the similarity. The output layer employs the activation function to compute the final decision for each output node, as depicted in expression (4),(5). This computation involves the generalized inverse for all activated image frames, multiplied by the corresponding binary label (L) of the same length as the set of training image frames. Binary labels are assigned as follows: 1 for the same category and 0 for others.

$$\text{Sigma}=\frac{1}{m}\sum_{j=1}^{f}\sum_{i=1}^{m}\left\|F_i-\mu_j\right\|^2 \tag{2}$$

$$\beta=\frac{1}{2sigma^2} \tag{3}$$

$$\Phi(\text{F})=e^{\beta||F_i-\mu||^2} \tag{4}$$

$$\theta=(\varphi(\text{F})^T\varphi(\text{F}).\,\varphi(\text{F})^T*L_c \tag{5}$$

As obvious, Radial Basis Function Neural Network offers a powerful framework for video similarity analysis. By leveraging the architecture's hidden layer computations, clustering, and feature extraction, the network can effectively extract and analyze similarity relationships within video categories. Additionally, training the network on all video sets enhances the activation function of the hidden layer, enabling a more comprehensive understanding of similarity relationships across the entire video dataset. This approach contributes to improved video recognition and classification outcomes, facilitating more accurate and efficient analysis of video similarity.

3.3 Self-Organizing Maps Network SOM

In the realm of neural networks, the Self-Organizing Maps (SOM) network stands out with its unique architecture. As described in [21], the SOM network comprises two layers: the input layer and the output layer. The output layer consists of a square grid of neurons, each initialized with a weight W. These weights are trained to compete with neighboring neurons to identify the matching input video image feature O of hand gesture. The winning neuron is determined based on its distance D from the input video image feature. To update the weights, neighboring neurons in two directions are taken into account, utilizing the Gaussian function N that dependes on the distance between the considered neuron i and the winning neuron j in the output neuron grid of length n*m. The neighborhood function is higher for neurons closer to the winner, as expressed in equations (7) and (8).

Through the process of weight adjustment, the weights W converge to optimal values that are proportional to each input video image feature, as depicted in equation (9), where L represents a small learning rate. The SOM network operates as an unsupervised learner, as it does not rely on the actual values of desired classes. Competitive learning is employed to extract adjacent topological features in the image training set, assigning each input video image feature to the closest weight. In the SOM network's output layer, it is evident that features from input video images of a single category can correspond to multiple neurons. This characteristic enables the collection of data about similar features at multiple locations, thereby aiding in the classification of complex data within video images. The network's ability to gather data about multiple neurons makes it easier to distinguish overlapping images from distinct categories.

$$\mathrm{Dj} = \sqrt{\sum_{i=1}^{n} \left\| O_i - w_{ij} \right\|^2} \quad (6)$$

$$N_{i,j}(O) = \exp\left(-\frac{D_j^2}{2\sigma^2}\right) \quad (7)$$

$$\sigma = \sqrt{n^2 * m^2}/2 \quad (8)$$

$$W_{ij}(t+1) = W_{ij}(t) + L.N_{i,j}(t)\left[O_i(t) - W_{ij}(t)\right] \quad (9)$$

The utilization of Self-Organizing Maps (SOM) in video image classification provides valuable insights and enhanced classification capabilities. Through neuron competition and weight adjustment, the network extracts topological features and enables the classification of complex feature present in video images. By gathering data from multiple neurons, the SOM network improves the classification of overlapping images from different categories. The SOM network's architecture and learning process contribute to more accurate and comprehensive video image classification, facilitating various applications in video recognition and analysis.

3.4 Temporal Sharing Neural Networks

To fully comprehend the true meaning behind a video, it is essential to connect the visual elements in the chronological order in which they appear to uncover their true significance. Recurrent neural networks have demonstrated remarkable effectiveness in this regard; they can be classified into three different types based on the mathematical operations and relationships involved in training the network. In our proposed work, we will explore the applications of these recurrent neural network types and analyze their performance in hand gesture video analysis and understanding tasks.

3.4.1 Recurrent Neural Network (RNN)

Recurrent neural network (RNN) is well-suited for processing historical image frames in video due to its ability to share weights cyclically across time through the hidden layer states. The learning process is mathematically formulated in expressions (10 to 13). Since RNN can accumulate and calculate weights over time during the training, it's important to consider a series of events occurring over a certain period to fully understand a significant movement inside a video. This allows for accurate analysis and capture of significant movements within the video.

$$h^{(t)} = b_h + W_{sh}\, S^{(t-1)} + W_{Fh}\, F^{(t)} \quad (10)$$

$$S^{(t)} = \Phi\, (h^{(t)}) \quad (11)$$

$$C^{(t)} = b_C + W_{Ch}\, h^{(t)} \quad (12)$$

$$y^{(t)} = \mathrm{Loss}(C^{(t)}) \quad (13)$$

In these equations, 'F' symbolizes an individual image frame feature within a hand gesture video clip, where each image frame is influenced by the preceding one. 'W' denotes the weight vector associated with the inputs from the video clip across various time steps't' spanning both the hidden and output layers. Consequently, 'h' and 'S' correspond to the image frames within a single video clip at a particular time step inside the hidden layer. Furthermore, 'C' and 'y' signify the predictions generated by the output layer at a specific time point within one video clip. During forward propagation in the network, each layer processes multiple image frames that collectively form a video clip. The input frames are sequentially passed through the hidden layer states, one batch at a time, and activated using the nonlinear activation function Φ. The first two equations (10, 11) correspond to the hidden layer, while the last two equations (12, 13) pertain to the output (prediction) layer. The prediction (output) layer incorporates a multi-loss function, allowing for predictions to be represented as maximizing neuron-wise values across multiple image frames. The neuron-wise layer with the highest dominance yields the best result, thus characterizing the RNN topology as "many-to-many" (refer to Figure 1).

Overall, RNN process involves the sequential processing of video image frame features, the utilization of nonlinear activation functions, and the application of a multi-loss function for prediction in the output layer. This is followed by backward to adjusting the weights.

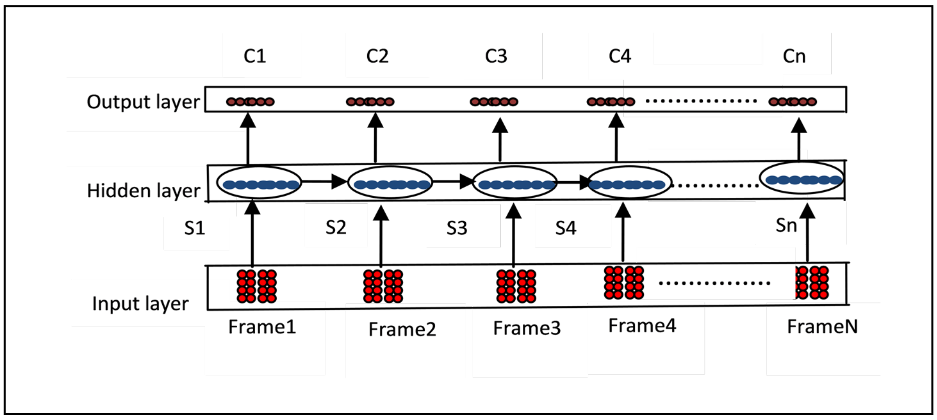


Figure 1: proposed RNN layers.

3.4.2 Long short-term memory (LSTM)

The network uses the same primary algorithm as RNN training, with an additional hidden layer Figuare2. Each layer has its own activation function Φ for each state, the first layer using a sigmoid function (14) and the second layer using double from the activation function of tanh (15). The output layer is represented as (16) and (17). As every layer is a feed-forward neural network with a nonlinear function, the loss function chosen for the output layer is cross-entropy log function explained in (22).

$$I = \Phi(\ W_{FI}F^{t} + W_{hI}\, h^{t-1} + b_{I}) \tag{14}$$
$$S = \Phi(\Phi(W_{Fs}F^{t} + W_{hs}\, h^{t-1} + b_{s})) \tag{15}$$
$$C^{t} = b_{C} + W_{C}\, S^{t} \tag{16}$$
$$Y^{t} = \text{Loss}\,(C^{t}) \tag{17}$$

In a video clip of hand gesture images, the current image frame feature F depends upon the previous image frame feature h. Within the hidden layer, a weight vector W is linked to each image frame feature in the video clip. The output layer is represented as C and y. The network undergoes a forward pass, followed by a backward pass, to fine-tune the weights.

3.4.3 Gated recurrent units (Gru)

Notably, the GRU network shares the same design and training algorithm with the LSTM network. As depicted in Figure 2. It consists of two hidden layers, as shown in equations (18) and (19). The main difference lies in the second layer, where a single activation function Φ of tanh is employed, as illustrated in equation (19).

The output layer is represented by equations (20) and (21), which utilize the cross-entropy log as the chosen loss function as depicted in (22).

$$I = \Phi\ (\ W_{IF}F^t + W_{hI}\, h^{t-1} + b_I) \quad (18)$$

$$S = \Phi\ (W_{Fs}F^t + W_{hs}\, h^{t-1} + b_s) \quad (19)$$

$$C^t = b_C + W_C\ S^t \quad (20)$$

$$Y^t = \text{Loss}\ (C^t) \quad (21)$$

In the context of a single video clip capturing hand gestures, much like the principles explained in LSTM, the current video image frame feature F depends on the previous video image frame feature h. Weight vectors W are associated with each image frame feature within the video clip across hidden layer. The output layer is represented by C and y. The network undergoes a feed forward process, followed by a backward pass to adjust the weights.

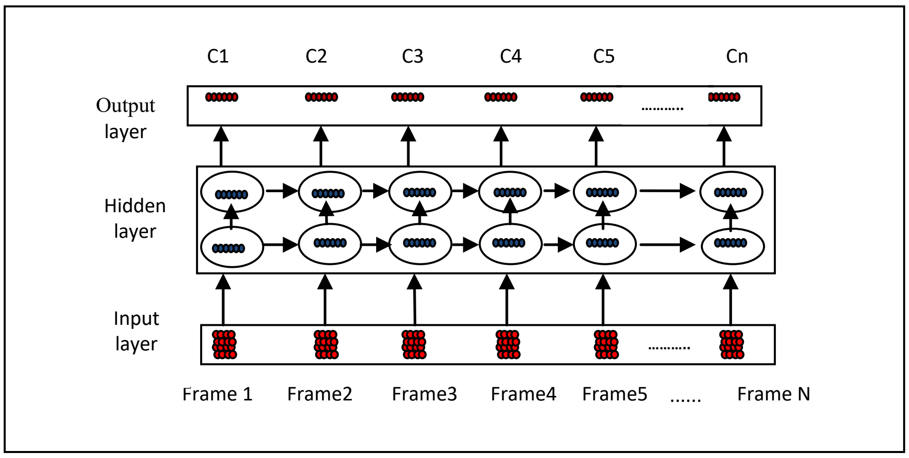


Figure 2: proposed LSTM and GRU layers.

4 Proposed Methodology

The general methodology consists of four stages, illustrated in Figure 3. The first stage involves organizing and preparing the data to format the videos to fit the proposed models. This is followed by training the CNN network and extracting the necessary features, which prepares the system to handle the two proposed neural network models. The final stage involves making the final decision by sharing the predicted outcomes from each proposed model, using the decision system of the proposed adaptive AdaBoost.

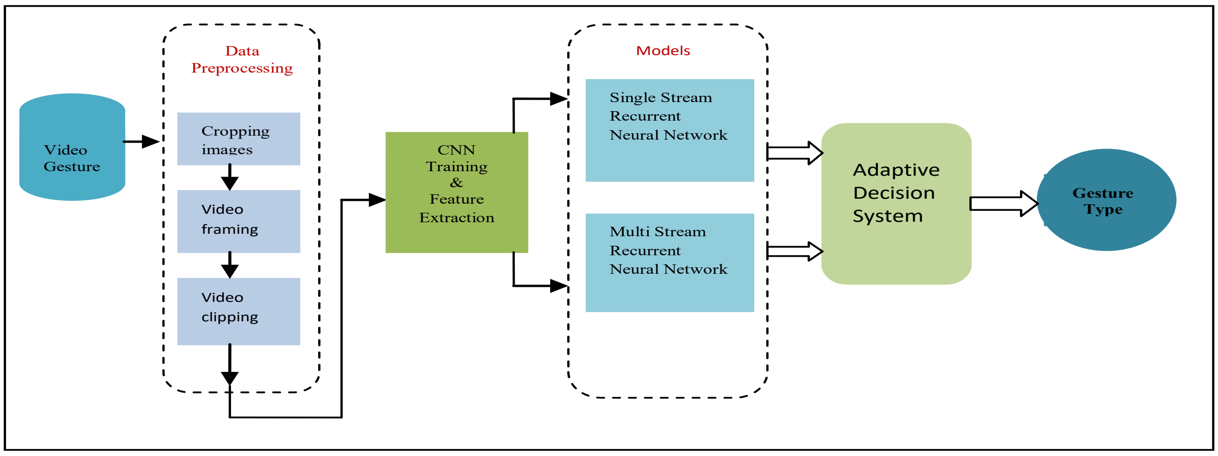


Figure 3: The Proposed Methodology Architecture.

4.1 Video Gesture description

We analyzed two multimodal datasets of sign language videos: Chalearn 2014 [22] and ChalearnIsoGD [23]. Chalearn 2014 contains 14,000 continuous samples of single signers of 20 Italian gestures performed by 27 people, while ChalearnIsoGD contains 35,878 samples of 249 hand gestures representing words from 30 different lexicons, gestures performed by 21 different signers. Both datasets are in RGB and depth video format, with a resolution of 640 x 480. For each signer of Chalearn 2014 [22] , skeleton data is provided.

Notably, the testing videos of Chalearn 2014 (3,555 videos) may include signers who also appear in the training set, exhibiting different hand gestures. In contrast, the testing videos of ChalearnIsoGD (6,271 videos) do not contain repeated actions from individual signers; instead, new signers perform the gestures, challenging the dataset's recognition performance. We preprocessed both datasets by segmenting continuous sample videos into separated videos containing meaningful hand movements, omitting static movements. We also resized the dimensions of each frame image in the Chalearn 2014 dataset to match the corresponding skeletal structure, minimizing the effect of background confusion during classification.

4.2 CNN Training and Feature Extraction

The architecture of the employed CNN is composed of four blocks, each consisting of two convolutional layers with a 3x3 kernel filter. An RLU activation function is applied after each convolutional layer. Following each block, max-pooling is performed. The filter sizes for the four blocks are 16, 32, 64, and 128, respectively. Two fully connected layers with 2048 neurons each follow the convolutional layers. A dropout rate of 0.5 is applied to reduce overfitting. The network is trained using the cross-entropy log loss function, which calculates the difference between the predicted output and the ground truth (22).

$$-\sum g(x) \log p(x) \tag{22}$$

The membership probability of the target class x concerning other classes y and weights w is estimated by maximizing the joint probability of x and y given w, which can be represented as P(x, y|w). The likelihood of the estimation is maximized using stochastic gradient descent with momentum and a learning rate as optimization methods to adjust the weights. The weight initialization method of Xavier is used, which determines the initialization scale based on the filter length (n), as shown in (23).

$$W=\sqrt{\frac{2}{n}} \tag{23}$$

For training CNN network, a single frame is used as a linear process to adjust the weights. The video frames have been scanned sequentially to obtain the inputs for training CNN network. The training procedure in this scenario is represented by the linear operation in (24) for each convolutional layer. The video images were scanned one after another.

$$O_k=I_{k-1}(x_1,y_1,c,) * W_k (x,y,f) + b_k \tag{24}$$

Here, W represents the learning weight associated with a kernel of size (x, y) of (f) filters, b are the biases on the epoch time k with input image I of dimension (x1, y1) and number of chanels c.
Multiframe is the second scenario for training the CNN network. The first convolution layer of the previously mentioned network design is changed to 3dconv across 32 frames (the recommended number of frames to recognize one gesture movement) by feeding it across the dimension of channel. Hence, the linear operation for the whole process can be represented as follows:

$$O_k=I_{k-1} (x,y,t) * W_k (xx , yy,f) +b_k \tag{25}$$

In this context, (I) stands for the input image frame composed of temporal shared image frames across time (t) with image dimensions x and y. These frames are multiplied by training weights (W) corresponding to a kernel filter of size (xx, yy) for (f) filter. Additionally, a bias term (b) is introduced as a constant to ensure data balance during the epoch number (k).

Regarding CNN feature extraction, the features are obtained from the last layer of the well-trained CNN on the training image set. This layer provides a dense representation of the visual features for image data classification. The process involves extracting features from image X using the well-trained CNN network, applying the optimally learned weights W to all convolutional layers of the deep network up to the last flattening layer, without performing weight adjustment. This results in the extraction of feature F, as depicted in equation (26).

$$F_k=X_{k-1} * W_k \tag{26}$$

As a result, the output of the last fully connected layer is a feature vector of size 2048, following the network design described earlier. This feature vector serves as a dense representation of the visual feature.

4.3 Proposed Models

In this study, several neural network models have been proposed to generate new frame-wise features by enhancing the performance of CNN features. The SOM network and RBF network were trained on CNN features, composite features were constructed by leveraging the strengths of each participating neural network. These composite features were utilized to train new networks with various designs for time-sharing recurrent networks. The proposed models can be classified into two categories based on the nature of the connection between the temporal-sharing neural networks, as outlined below:

4.3.1 Single Stream Recurrent Neural Network

This recognition model utilizes an RNN to train a video clip, leveraging a comprehensive set of diverse features for each image frame. The training process involves extracting features from a CNN, and both SOM and RBFN networks are employed to utilize and enhance these CNN features. Subsequently, an RNN network is employed to train the video clip. The process incorporates frame-level features extracted from standard enhancing networks (SOM and RBF) and combines them by concatenation with the CNN features in a unified layer, forming a comprehensive representation of each image frame. Furthermore, the enhanced features extracted from each standard network are organized into separate layers, creating different types of dense features for each image frame. This enables the RNN network to effectively train on the video clip using various processing pipelines, as illustrated in Figure 4. The overall process yields seven distinct types of dense features (frame-wise features) extracted from different training pipelines, as summarized in Table (1), with each type representing a single image frame. As a result, the RNN is trained using the diverse features obtained from multiple pipelines, following the procedures outlined in (1).

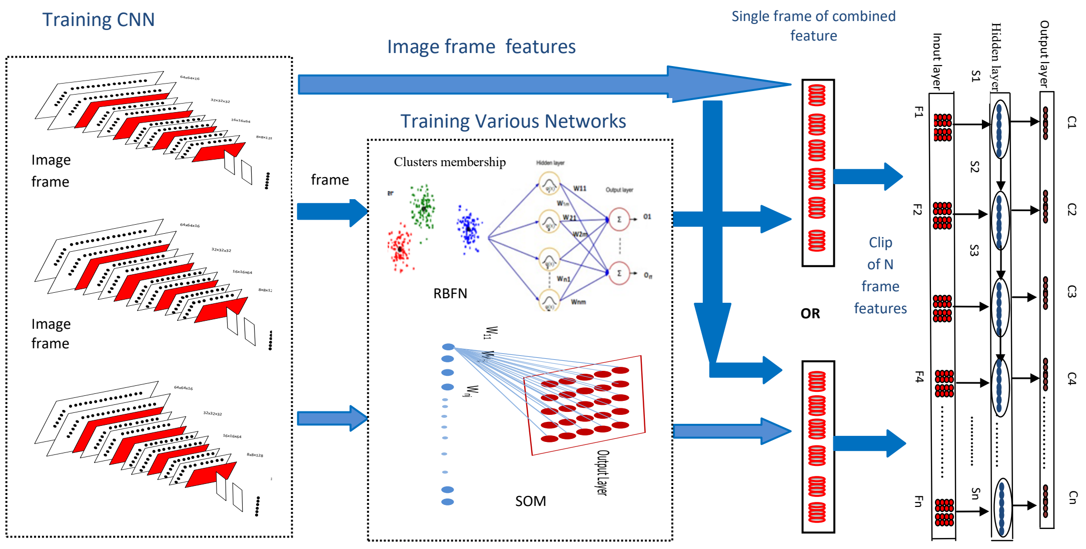


Figure 4: Structure of One Stream Recurrent Neural Network Model.

Procedure 1 Training of single Stream Model: SSM (X_n ,Y_n)

```
/*X is a set of CNN features along with the corresponding labels Y for n images */

1.Retrieve set of B and S Features from training each of RBF networks and SOM for n images.

    1.1    $B_n$=train RBF ($X_n$ ,$Y_n$)
    1.2    $S_n$=train SOM ($X_n$)

2. Extract set of vectors V of length n from different features for n images .

    2.1    $V1_n= B_n \cup S_n$
    2.2    $V2_n= X_n \cup B_n$
    2.3    $V3_n= X_n \cup S_n$
    2.4    $V4_n= X_n \cup S_n \cup B_n$

3. Train the RNN using various sets of feature vectors to obtain corresponding sets of prediction values P
   and their associated scores S for n images.

    3.1    Return ( $P1_n$ ,$S1_n$) = Rnn ($X_n$ ,$Y_n$)
    3.2    Return ( $P2_n$ ,$S2_n$) = Rnn ($B_n$,$Y_n$)
    3.3    Return ( $P3_n$ ,$S3_n$) = Rnn ($S_n$,$Y_n$)
    3.4    Return ( $P4_n$ ,$S4_n$) = Rnn ($V1_n$,$Y_n$)
    3.5    Return ( $P5_n$ ,$S5_n$) = Rnn ($V2_n$,$Y_n$)
    3.6    Return ( $P6_n$ ,$S6_n$) = Rnn ($V3_n$,$Y_n$)
    3.7    Return ( $P7_n$ ,$S7_n$) = Rnn ($V4_n$,$Y_n$)
End
```

The feature vector extracted from the CNN is then utilized to train both the RBFN and SOM models to obtain new features. These new features, along with the feature frames, are fed side by side to train the RNN.

For the standard RBF network, the features F from the image in the training set are extracted by matching the optimal weights W obtained from the previously trained RBF with the outputs of the activation function φ. The activation function φ represents the distance between the ideal beta and the locations of the ideal cluster centers of all image features in the training set. This can be expressed as in expression (27). On the other hand, the features of the SOM network are determined by considering the label of the closest distance D between each image feature (i) in the image training set T and the optimal weights W in the SOM output layer. This is illustrated in expression (28):

$$F = \varphi * W \tag{27}$$

$$D = \sqrt{\sum_{i=1}^{n} \left\| T_i - w_{ij} \right\|^2} \tag{28}$$

These extracted features from both the RBFN and SOM networks are then used in conjunction with the feature frames to train the RNN model.

4.3.2 Multi Stream Recurrent Neural Network

In this phase, a time-shared recognizer is trained by utilizing diverse predictors derived from other time-shared recognizers that have been trained with different enhanced features. The proposed network model is a deeper design that incorporates features of recurrent multi-stream networks. In this model, the features extracted from RNN training, utilizing both CNN features and the enhanced features from training each of the RBFN and SOM networks separately, are reintroduced into another temporal sharing network. Specifically, only the isolated training pipelines from the previously proposed models are fed into an additional recurrent temporal sharing network. Since we did not obtain significant differences from the other types of the previously extracted features, we chose to include these types only. This step is performed after integrating all the extracted features from the isolated trained pipelines into a newly created concatenated feature layer to represent a single image frame. Consequently, an LSTM, GRU, or RNN network is trained on the video clip using the newly constructed density of image frame features. This training process utilizes the modified density of image frame features (frame-wise features) obtained from the concatenation process. The detailed process is elucidated in Procedure (2), while the corresponding flow of the process is depicted in Figure 5.

```
Procedure2 Training of Multi Stream Recurrent Model: MSRM (Xn,Yn)

/*X is a set of CNN features along with the corresponding labels Y for n images*/

  1  Retrieve set of B and S Features from training each of RBF and SOM for n images.

       1.1   $B_n$= RBF ($X_n$ , $Y_n$)
       1.1   $S_n$= SOM ($X_n$)

  2  Derive Various Features, denoted as F for n images, from training of RNN using different input
     features .
       2.1   $F1_n$ = RNN ($X_n$, $Y_n$)
       2.2   $F2_n$ = RNN ($B_n$, $Y_n$)
       2.3   $F3_n$ = RNN ($S_n$, $Y_n$)

  3  concatenate features in V1 for n images.

       3.1   $V1_n = F1_n \cup F2_n \cup F3_n$

  4  Find set of prediction values P with the corresponding scores S for n images from training each of
     LSTM and GRU.
       4.1  Return ($P8_n$ , $S8_n$) = LSTM ($V1_n$,$Y_n$)
       4.2  Return ($P9_n$ , $S9_n$) = GRU ($V2_n$,$Y_n$)
       4.3  Return ($P10_n$ , $S10_n$) = GRU ($V3_n$,$Y_n$)
End
```

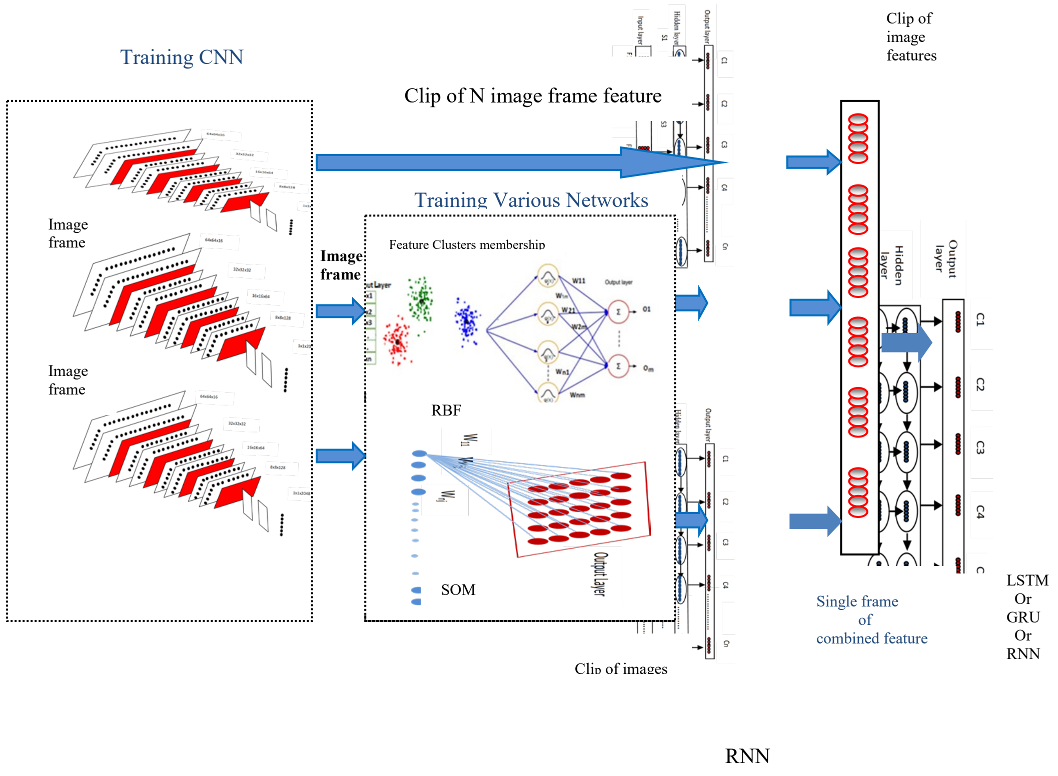


Figure 5: Structure of Multi Stream Recurrent Neural Network Model.

In the previous section, we discussed the extraction of image features from CNN, RBF, and SOM models on the training image sets. The process of extracting image features from each kind of recurrent neural network involves applying the optimal weights W from all the predictors P of a well-trained network on the time-shared image features over time t. This process is described by equation (29), where X represents a clip of image frames during training. These recurrent neural networks are capable of extracting features that represent the entire image clip in the video. The overall length of the extracted features is determined by the length of the predictor, which corresponds to the number of classes, multiplied by the length of the clip images.

$$F=X(t) * W_P(t) \tag{29}$$

4.4 Decision System Mechanism

In order to assess the effectiveness of the proposed system, the pipelines developed from each neural network model presented in this study were applied to unseen data, known as the test dataset. Following the same approach for each pipeline of the neural network model on the test dataset, the optimal weights obtained from training were utilized without requiring feedback for adjustments. Therefore, the mechanism of the decision system consists of two stages: the individual predictions generated by each proposed network model, known as the prediction phase, followed by the integration stage, where the prediction results from all the proposed neural network models are combined.

I. Prediction Phase

The process in temporal sharing networks, as mentioned earlier, follows a many-to-many approach, where a series of image frames predicts a series of image frames. Consequently, each image frame corresponds to a set of prediction nodes, with the number of nodes equal to the total number of classes in the problem. The objective of recurrent networks is to maximize their performance, and thus the winning neuron that classifies a frame is the node with the highest prediction value. Therefore, the winning frame of a clip is the frame with the highest prediction value. Since all the proposed models in this work employ a recurrent network with temporal sharing within a clip, the prediction for each model needs to be performed at the clip level.

Hence, the prediction phase in a presented model can be divided into two steps. Firstly, the average of the corresponding prediction values across each prediction layer in the recurrent network is computed. Equation (30) represents the prediction average, resulting in a single prediction layer that encompasses the entire clip.

$$\mathbf{P} = \sum_{i=1}^{m} \sum_{j=1}^{n} \frac{F_{ij}}{n} \qquad \mathbf{(30)}$$

Here, P represents a vector of n prediction nodes for one clip, where F is a set of vectors from m frame layers for that clip, each containing n prediction nodes.

Secondly, to obtain the overall model prediction score, the average function is followed by the maximization function, as indicated in (31).

$$\mathbf{Score} = \forall_{n} \mathbf{Argmax}(\boldsymbol{P_n}) \qquad \mathbf{(31)}$$

In this equation, P represents the vector of prediction layers with a length of n for one model, and Score represents the final score prediction for that model's clip.

II. Integration Phase

The integration process of all the presented network models across different pipelines aims to leverage the prediction properties of each model, ultimately improving the final class decision outcome. This process incorporates adaptive AdaBoost to enhance the overall decision-making of the ten network models discussed earlier. To facilitate integration among all the proposed network models in this work, a four-stage decision system has been devised to ensure accuracy and improve the weak predictions from each model. The initial stage of integration involves a refinement process that capitalizes on the highest repeated prediction score **Highest(Score)** among all the previously presented models (32).

$S = \{s_1, s_2, \ldots, s_m\}$ be the set of m scores predicted by the multimodels.

Rep(si) be the function that counts the number of times each score s_i appears in S.

BestModel(si) be the function that identifies the model with the best performance associated with score (s_i).

The mathematical expression for the process is as follows:

$$\text{Highest(S)} = \begin{cases} \text{argmax } (s_i \in \text{DuplicateScores})(s_i), \text{if DuplicateScores} \neq 0 \\ \text{argmax } (s_i \in S)(\text{BestModel } (s_i)), \text{if DuplicateScores } = 0 \end{cases} \qquad \mathbf{(32)}$$

Where:

DuplicateScores = $\{s_i \mid \text{Rep}(s_i) > 1\}$ is the set of scores with duplicates.

argmax ($s_i \in$ DuplicateScores)(s_i) selects the score with the highest prediction value among the scores in DuplicateScores. The expression "argmax (si ϵ S) BestModel (si))" selects the score associated with the model that exhibits the best overall performance, indicating the corresponding maximum prediction value among all scores in S when there are no duplicates.

In the second stage, the acquired predicted values from all the proposed models have been organized into two distinct categories: weak and strong. This categorization process has been facilitated through the utilization of the 'Isolate' function, which relies on the accuracy values extracted from the scores derived from the 'Highest' function. As a result, the classes that demonstrate the highest accuracy levels are designated as strong, while those that do not meet this threshold are categorized as weak. Moving on to the subsequent third stage, the weak categories undergo Adaboost training, as outlined in Algorithm 1, while deliberately excluding the strong categories from the training process.

During the final stage, the previously separated Weak and Strong categorises are harmoniously merged through a combination function, leading to an optimized training experience for both categories. This innovative approach significantly reduces the overall training time required for AdaBoost, enhancing its efficiency.

AdaBoost [24] is an ensemble learning technique that leverages the output predicted classes from the proposed multiple models to construct a strong classifier. Through a sequence of iterations, it focuses on models previously misclassified from the Weak categories, by adjusting their prediction weights to reduce learning errors. The final classification decision is made by substituting the prediction values of these misclassified with a new different weight. This process is carried out for each model in the multimodel problem, resulting in a set of trained AdaBoost models. Each proposed model is independently trained to distinguish one model class from the others in a one-vs-all approach, employing positive and negative representations. The model with the highest learning class value is selected as the winner, enhancing its predictive capability (p) among the K models, as explained in (33).

$$F(P) = \text{Argmax}\,[P(K)] \tag{33}$$

```
Algorithm1: Model Integration (P, Sc)
Input:   P is a set of L network model prediction values for N images, and Sc is a corresponding set of scores.
Output:  FP The final prediction scores for N images.
Begin
• DScore $\leftarrow$ Highest(Sc)
  - Find 'DScore', the score with the highest repetition as defined in expression (32) for N images.
• Acc $\leftarrow$ DScore /* find the accuracy 'Acc' for each category*/
• Isolate(P, Sc, Acc)
     - Using 'Acc' isolate 'P' and 'Sc' into two sets:
    - 'W(P)' for prediction values of Weak categories and 'S(P)' for Strong categories among the L network
       models with a length of L1.
    - 'W(Sc)' for scores of Weak categories and 'S(Sc)' for Strong categories among the L network models
       with a length of L2.
   For C from 1 to L   /* Iterate over L network models*/
    1.   Terror $\leftarrow$ 0   /* Initialize the total error to zero*/
    2.   $W_C$(Sc) $\leftarrow$ W(Sc)
              - Score transformation: where the scores of the "C" model are replaced with a value of (+1),
                while the scores of all  other models   are substituted by (-1).
    3.   W $\leftarrow$ 1/length($W_C$(Sc)) /* Initialize the weight 'W'  */

        For j from 1 to T /* Perform T iterations*/
              a.  [$E_C$, Err] $\leftarrow$ classifier(W(P), $W_C$(Sc),W)
                    - Apply a threshold classifier to 'W(P)' after assigns (+) for model C and (-) for other models.
                    - '$E_C$' is the estimated model, and 'Err' represents the initial error.
              b.  Alpha $\leftarrow$ 1/2 * (log(1-Err) / Err) /* Calculate 'Alpha'*/
              c.  $W_{j+1} \leftarrow (W_j * e^{Alpha*W_C(Sc)*E_C}) / \sum_{i=1}^{n}(W_j * e^{Alpha*W_C(Sc)*E_C})$   /* Adjust weights 'W'*/
              d.  ET $\leftarrow$ Sign $(\sum_{i=1}^{n} E_C * Alpha)$
                   - Calculate the total estimation 'ET'.
                   - 'sign'function returns 1 for output greater than 0 and -1 for output less than 0.
              e.  Terror $\leftarrow$ Terror+ $\sum_{i=1}^{n} \frac{(ET \neq W(P))}{length(W(P))}$   /* Update the total error 'Terror'*/
             f.  If Terror = 0 then break /* Stop if the error becomes zero.*/
        End
    4.   $Model_C \leftarrow sign\ (\sum_{i=1}^{n} Alpha * E_C)$
             -  accumulate all predictions from '$Model_C$' across L models, each with a length of L1, as the
                training follows a one-vs-all approach.
             - 'sign'function returns 1 for output greater than 0 and -1 for output less than 0.
 End
• PP $\leftarrow$ Max ($Model_C$)   /* Select the Final prediction score 'PP' for Weak categories of length L1*/
• FP $\leftarrow$ Combine (PP, S(Sc))
    /* Combine scores from strong 'S' categories with 'PP' to produce the final score prediction 'FP' for N images*/
End
```

The practical implementation diagram of our proposed method is illustrated in Figure 6, showcasing the overall process. Additionally, Algorithm 2 provides a comprehensive outline of the steps involved. In contrast, for a detailed and step-by-step evaluation process of the entire workflow, refer to Algorithm 1. The decision system functions in two distinct phases: the first phase focuses on decision-making for each proposed network model, while the second phase involves the integration of decisions from all the proposed models.

```
Algorithm 2: Overall steps for the proposed work.
Input:  CNN features of training image sets T= {($X_1$,$Y_1$),($X_2$,$Y_2$),…($X_n$, $Y_n$ )},
        where y is the label of feature X of length n.
Output:  a set of optimal scores O of length n images.
Begin
    1 [$P_{mn}$ , $S_{mn}$ ] $\longleftarrow$ SSM ($X_n$, $Y_n$)
      - a set of (m =7) prediction values P, along with their corresponding scores S, for all images of length n
        as outlined in procedure 1 is out  .
    2 [$P_{kn}$ , $S_{kn}$] $\longleftarrow$ MSRM ($X_n$, $Y_n$)
      - a set of (K=2) prediction values P, along with their corresponding scores S, for all images of length n
        as outlined in procedure 2.
    3 $XX_n$ $\longleftarrow$ Concat ($P_{kn}$,$P_{mn}$) /* set 'XX' of length (h=10) as a concatenation for all set of the prediction values */
    4 $Class_n$ $\longleftarrow$ Concat ($S_k$,$S_m$ )  /* set 'Class' of length (h=10) as a concatenation for all set of Scores */
    5 $O_n$ $\longleftarrow$ Models Integration ($XX_n$, $Class_n$) /*Apply Adaptive Decision as in Algorithm1 for n images*/
End
```

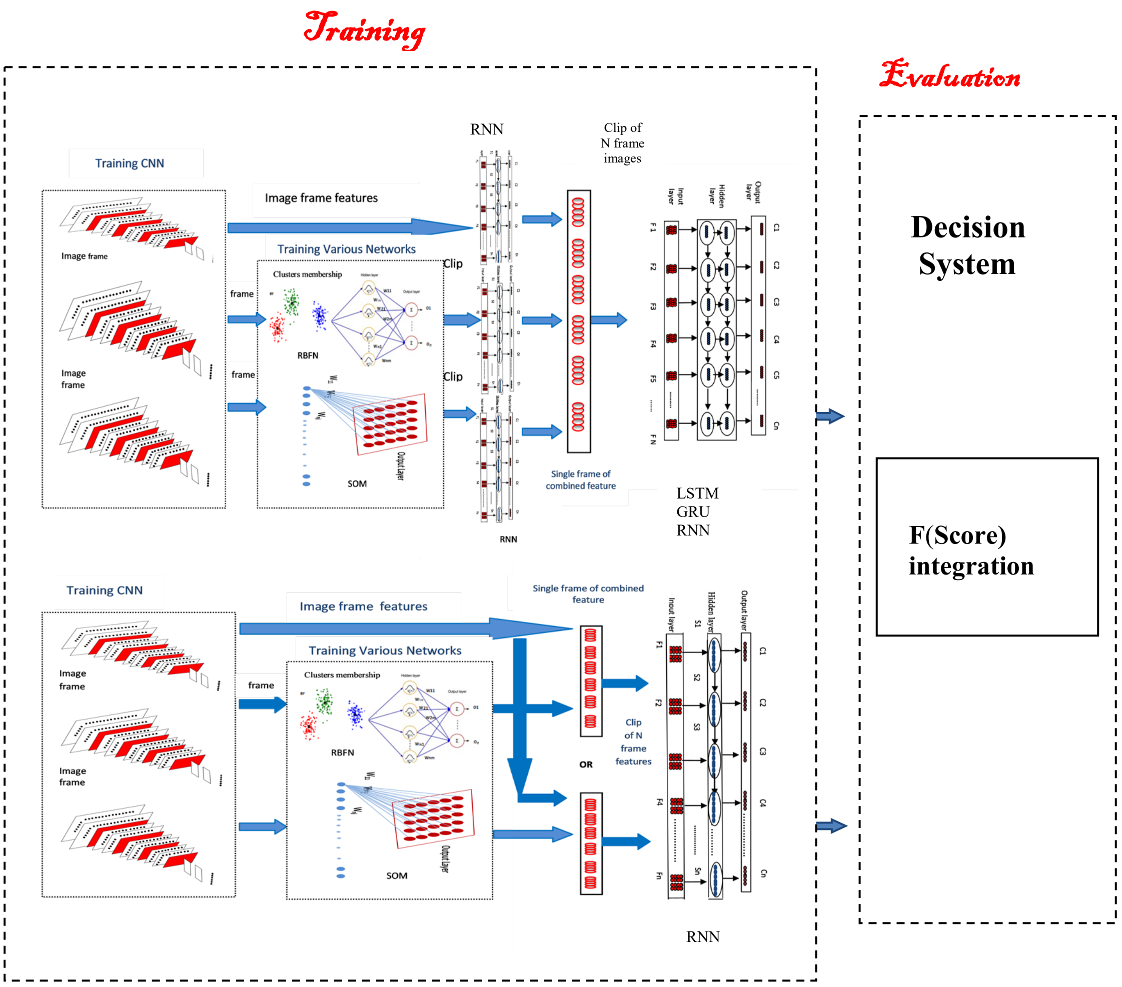


Figure 6: The overall structure of the proposed work, depicting the networking pipelines.

4.5 Evaluation Metrics

In this study, we have employed evaluation metrics to effectively assess the performance of the proposed models. The accuracy metric, as defined in equation (34), has been utilized to quantify the proportion of correctly predicted gesture videos (G) out of the total number of gesture videos (T) in the test dataset. The accuracy is presented as a percentage:

$$\text{Accuracy}=\frac{\mathrm{G}}{\mathrm{T}}\ X\ 100 \qquad (34)$$

To facilitate a meaningful comparison of our results with prior research, we also computed the Jaccard index (35). This index measures the degree of similarity within the tested sample videos by evaluating the binary intersection of predicted values (O) and ground truth labels (T), divided by the binary union of these prediction results and ground truth labels across all video samples (V). In each video sample, an individual signer performs a continuous array of hand gestures belonging to various gesture classes (C).

$$\text{Jaccard} =\frac{1}{VC}\sum_{v=1}^{V}\sum_{c=1}^{C}\left|\frac{T_{v,c}\cap O_{v,c}}{T_{v,c}\cup O_{v,c}}\right| \qquad (35)$$

5 Experimental Results

The primary objective of this research is to demonstrate the effectiveness of the proposed adaptive AdaBoost method on hand gesture recognition videos. To evaluate our approach, we have employed the evaluation protocols designed for continuous gesture recognition on the test dataset, which mainly utilizes the Jaccard index as the evaluation metric (where higher values indicate better performance) in addition to accuracy. The results of these experiments have been detailed and analyzed to assess the effectiveness of the proposed methods.

5.1 Training of Neural Networks

This section delves into the training of numerous neural networks, each characterized by its own unique set of parameters and methods. A comprehensive discussion of these networks is presented below.

5.1.1 Convolutional Neural Network (CNN) Training

During the training phase, the Convolutional Neural Network (CNN) was trained using two approaches on the Chalearn 2014 dataset. This dataset consisted of 268,517 individual image frames extracted from 6,400 video gestures. Additionally, the CNN was trained on the ChalearnIsoGD dataset, of which only the first thirty categories dealt, including mutual, Italian, and Chinese numbers. The ChalearnIsoGD dataset comprised 267,329 single image frames. The evaluation was conducted on the corresponding testing videos, featuring categories 1-30.

The first approach involved training the CNN on single frames of resized color images, each sized at 224x224 pixels. The second approach focused on training the network using multiple frames or clips, with each clip containing 32 frames of resized color images with dimensions of 64x64 pixels. In total, this resulted in 171,600 clips of frames for Chalearn 2014 and 159,449 clips for ChalearnIsoGD. For ChalearnIsoGD, the weights for training the network were fine-tuned from the pre-trained AlexNet [25] network on ImageNet. An additional fully connected layer of 2048 neurons, preceded by a RELU activation layer, was added to match the dimensions of the network used for the first dataset. This approach aimed to gather meaningful features from the fine-tuning process to train the second network, thus avoiding overfitting.

A comparison between these two methods is presented in Figure (7). Regardless of the training approach, the CNN was trained with a learning rate of 0.001, weight decay of 0.0005, and momentum of 0.9, using SGD as the optimizer. The training was conducted using a batch size of 36. The MatConvNet MATLAB toolbox, which is a part of the VLFeat open-source library for computer vision, was used for training the deep convolutional neural network [26].

5.1.2 Radial Basis Function Network (RBFN) Training

The Radial Basis Function Network (RBFN) underwent specialized training on the features extracted from the CNN to improve image frame recognition. To determine the number of initial centers for each category in RBFN clustering, we divided the length of the category by 120. The initial center values were selected at random within their respective categories. Notably, our clustering process was optimized by disregarding zero clusters that might be generated. This approach resulted in a variable-length sigma for each class, leading to a varying length of Beta, which is an accumulation of the sigmas from each class. Consequently, the RBFN achieved its peak output feature length of (5415) for Chalearn2014 and (6213) for ChalearnIsoGD. We employed MATLAB and a tutorial provided by Chris McCronick[27] for the manipulation of the RBF network, ensuring robust training and performance optimization.

5.1.3 Self-Organizing Map (SOM) Training

The Self-Organizing Map (SOM) was trained on CNN features to enhance image frame recognition. For optimal performance, the SOM network was configured with 51x51 output neurons for both datasets, encompassing a total of 105 iterations. The determination of the iteration count and output layer neurons followed the methodology established by [28], which takes into account the overall dataset length. The initial learning rate was set at 0.01.

To determine the winning neuron on the output layer, we examined the weights of the neuron on the SOM output layer that best matched each data feature. As a result, the output feature length precisely matched the input feature length, which was 1x2048. For the training of the SOM network, we leveraged a MATLAB project designed for classifying handwritten digits, as provided by [29], on the same notebook previously mentioned.

Considering the features obtained from training the CNN on multiple frames couldn't be directly employed for training standard neural networks (RBF and SOM), we opted to use the features extracted from training the CNN frame by frame in all our proposed models. This decision was influenced by the relatively short duration of the videos, each representing individual gestures.

5.1.4 Temporal Sharing Neural Network Training

In this section, we present a detailed explanation of the training methodology employed for the Recurrent Neural Networks (RNNs) in our research. The RNN architecture utilized in our study follows a many-to-many design, where multiple input frames are processed, resulting in the generation of multiple corresponding outputs. This architecture is visually depicted in Figure 3. The hidden layer within the Recurrent Neural Networks is constructed as a multi-layer perceptron for each state. It facilitates the connection between the multiple input nodes and the hidden state nodes. To capture the intricate temporal dependencies present in the data, we incorporate the hyperbolic tangent (tanh) activation function. Notably, this activation function is applied alongside the sigmoid function, but only for the first hidden layer in each of the LSTM and GRU architectures. This combination of activation functions enhances the network's ability to capture and model complex temporal relationships. Furthermore, an essential aspect of the hidden layer is weight sharing with previous hidden states. This feature enables the network to effectively leverage temporal information by retaining and utilizing knowledge from previous time steps. By sharing weights, the RNN can capture and propagate relevant information throughout the sequence, enhancing its ability to learn and make accurate predictions. The output layer of the RNNs operates as a multi-predictor for a clip of frames, establishing connections between the hidden layer and the final output. This layer plays a crucial role in mapping the learned representations from the hidden layer to the desired output format. To make predictions, we employ the softmax loss function, as defined in equation (13), which effectively predicts 'n' values for each state. In our specific case, 'n' corresponds to the 20 classes on which the network has been trained. The softmax loss function aids in producing probability distributions over the classes, enabling the network to make confident predictions based on the learned features. By utilizing this training methodology, the Recurrent Neural Networks in our research are capable of effectively processing multiple input frames, capturing temporal dependencies, and generating accurate predictions for the specific classes they have been trained on.

The input for our Recurrent Neural Networks comprises a sequence of image frames, each with a composite of neural network features. To effectively handle this input, we employ the sliding window technique on every gesture video, which systematically extracts sequential clips with a slight overlap within each clip containing only one frame. Notably, each clip is composed of 32 image frames, and we consider a single clip as an individual temporal sharing networks input. In instances where the last clip in a video falls short of the standard clip length, we pad it to match the required length. As a result, our presented recurrent neural networks are composed of 32 prediction layers. Each prediction layer is comprised of 20 predictor nodes for Chalearn2014 and 30 predictor nodes for ChalearnIsoGD , corresponding to the number of classes on which the network has been trained. This architecture allows for the accurate classification of visual video gesture within the dataset.

Additionally, we presented a performance comparison in Figure 7. This comparison is provided to explain the improved CNN features. For clarity, we've denoted the network training paths with underscores (_) in the names of each pipeline, and the features from each network are concatenated in an additional layer, marked with a plus sign (+).

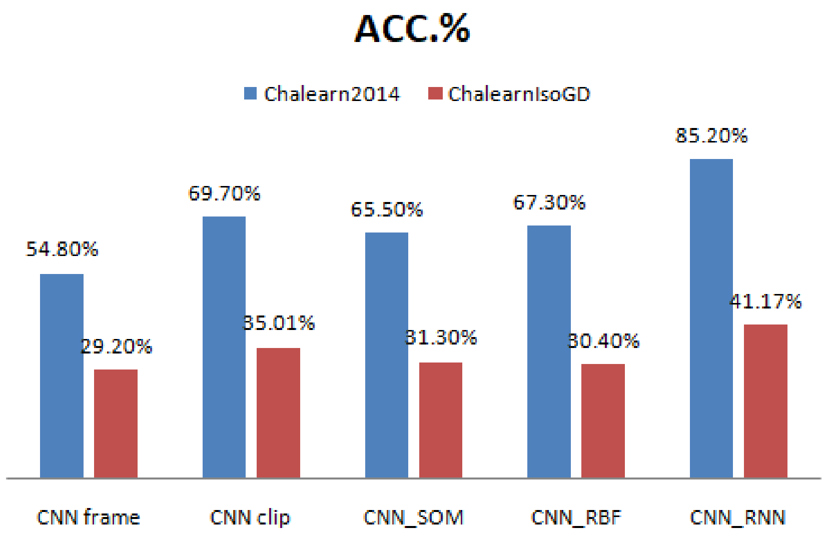


Figure 7: Accuracy of different enhanced CNN features.

In Figure 7, we presented a comprehensive comparison among various structural network that participated in the served models based on the accuracy score expression (34). Our observations reveal that the network evaluation varies depending on the type of network and its performance. Moreover, the RNN network has shown superiority over other networks in its training performance.
All networks contributing to the models served in this work were trained on an Asus notebook equipped with an Intel(R) Core(TM) i5-7300HQ CPU running at 2.50 GHz, without utilizing GPU acceleration, and with 16GB of RAM.

5.2 Training the proposed Models

In this section, we provide detailed insights into the training procedures for the models proposed in this study, all of which are meticulously designed to elevate the performance of hand gesture recognition. Different pipelines on the model of single stream recurrent neural network have generated a layer of feature vectors of various lengths. Therefore, we presented Table (1) explains the relationship between model pipelines and the total length of the corresponding feature vector layer of one frame on which an RNN was trained.

Table 1: network pipeline and length of the feature vector layer relationship.

Name of Pipeline	**Feature layer length**	
	Chalearn2014	ChalearnIsoGD
CNN_RNN	2048	2048
CNN_SOM_RNN	2048	2048
CNN_RBFN_RNN	5415	6213
(CNN+ CNN_SOM)_RNN	4096	4096
(CNN+CNN_RBFN)_RNN	7463	8,261
(CNN_SOM +CNN_RBFN)_RNN	7463	8,261
(CNN+CNN_RBFN+CNN_SOM)_RNN	9511	10,309

The length of the CNN image frame feature was 2048, as previously mentioned, corresponding to the output from the final fully connected layer. In contrast, the length of the Radial Basis Function Network (RBFN) feature for a single image frame was 5415 in Chalearn 2014 and 6213 in ChalearnIsoGD . This length exhibits variability due to several factors, including the number of initial centers chosen for clustering and the variability of beta. In our specific case, the output feature was an aggregation of the centers for each category, resulting in a length equal to the category's data size divided by 120.
Furthermore, the feature length of the Self-Organizing Map (SOM) for one image frame matched the length of the network's input image frame features, which was 2048. This alignment was due to the approach of determining the winning neuron's weights at the output layer by identifying the closest neuron weights trained in the SOM output layer for each feature of the input image frame.

The predominance of the output results generated by RNN training has been observed, as depicted in Figure 7. To further illustrate the efficacy of RNN as a model for temporal sharing neural networks, we have presented an accuracy comparison using expression (34), as shown in Figure 8. This figure illustrates the efficiency of RNN when applied to each extracted feature from the pipeline of the first proposed model.

In this regard, RNN has been trained on various types of features extracted from the previously proposed networks, either directly or through combinations in different arrangements, as explained in Figure 4. The training of RNN has been conducted on the extracted features detailed in Table (1).

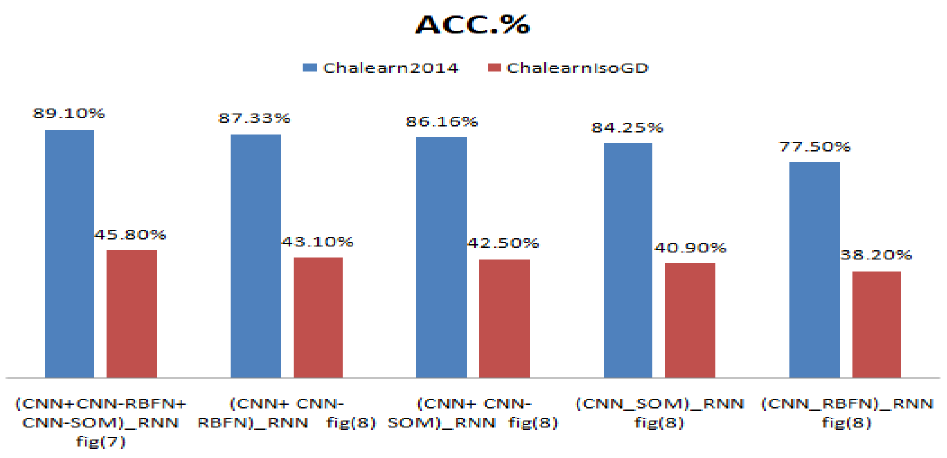


Figure 8: Accuracy of one stream recurrent neural network.

Figure 8 shows that the RNN trained on the features of multiple networks, each trained separately, outperformed the other models. This suggests that the RNN can recognize CNN features more accurately than CNN-enhanced features. However, a larger feature layer size can help the RNN achieve better recognition accuracy, even though the CNN variables are smaller than the improved ones.

Thus, work has progressed to present another temporal sharing neural network, so that the inputs that has trained the final network are a multiple stream from a set of time sharing recurrent networks obtained from the previous stage, as explained before in the proposed models of Section (4.3.2), where the features have been extracted from the output layer of each recurrent network by accumulating each prediction output layer for all participating image frames, so that the length of the features was 640, since 20 predictions were accumulated for each frame with 32 common output frames for Chalearn2014 and 960 for ChalearnIsoGD, since it has 30 category. The final length of the combining vector of the only two RNN streams, which were either LSTM or GRU, was (1280, 1920)for each of Chalearn2014 and ChalearnIsoGD respictivly, while the final length vector of the three RNN streams was (1920, 2,880) for each of Chalearn2014 and ChalearnIsoGD respictivly, and so on. It is worth noting that all kinds of temporal sharing neural networks were trained with a batch size of 100 and cells of 30 neurons on each state of the hidden and output layers, respectively, utilizing the optimizer RMSprop explained in equations (22, 23, and 24), and four iterations have been considered in training. The Matlab version from [30] was utilized to train each of RNN, LSTM and Gru networks. Table 2 shows the accuracy performance for the multistream network pipeline.

Table 2: Multistream network model in two datasets

Network Pipeline	**Chlearn2014**	**ChalearnIsoGD**

	ACC.%	Feature Layer Length	ACC.%	Feature Layer Length
(CNN_RNN+CNN_RBFN_RNN+CNN_SOM_RNN)_LSTM Fig. 5	94.41	1920	52.1	2,880
(CNN_RNN+CNN_RBFN_RNN+CNN_SOM_RNN)_GRU Fig.5	93.91	1920	51.8	2,880
(CNN_RNN+CNN_RBFN_RNN+CNN_SOM_RNN)_RNN Fig.5	93.41	1920	51.5	2,880

Although the multi-network model's feature layer is shorter than the feature layers of previous models, table 2 shows that it achieves better recognition accuracy.

5.3 Decision System and Integrating the Proposed Models

The integration model has been implemented through the evaluation phase, as explained in Section (4.4). All features from each pipeline model that ended with training a recurrent neural network have been shared in the decision system, in a total of ten models. As have been mentioned earlier, decision system of highest repetition score has been proposed before training AdaBoost to reduce the time of training, by isolating the full accurate predicted classes and making training focuses on the rest predicted classes of Weak evaluation, thus a better increasing in the right evaluation values, additional to, increasing the overall accuracy will be obtained with less training time. Figure 9 gives samples of Weak prediction and Strong predictions after applying the expression as in (32) of highest repetition score phase in decision system.

(A) Clip of Strong category “fame” (B) Clip of Weak category “vattene” (C) Clip of Weak category “perfetto”

(D) Clip of Strong category “2” (E)Clip of Weak category “11” (F) Clip of Weak category “12”

Figure 9: Examples of strong clip **A** and weak clips **B & C** from Chalearn2014 and strong clip **D** and weak clips **E & F** from **ChalearnIsoGD**

The w

t role image frames overlapping plays in confusing recognition. Despite the considerable differences in the construction of prediction systems for recognition in this work, the similarity between image frames in terms of hand poses still contributes to the confusion observed. Many image frames share a great resemblance either within the same clip or outside, leading to potential recognition errors. Although the temporal sequence is considered, the risk of confusion remains high due to the abundance of similar frames. Consequently, developing a decision system becomes necessary to aid in making accurate recognition decisions, particularly when multiple recognition decisions are involved. Training the system to determine the final winning decision effectively resolves this confusion. AdaBoost has demonstrated its quality in achieving the desired recognition accuracy, as indicated by the final accuracy results.

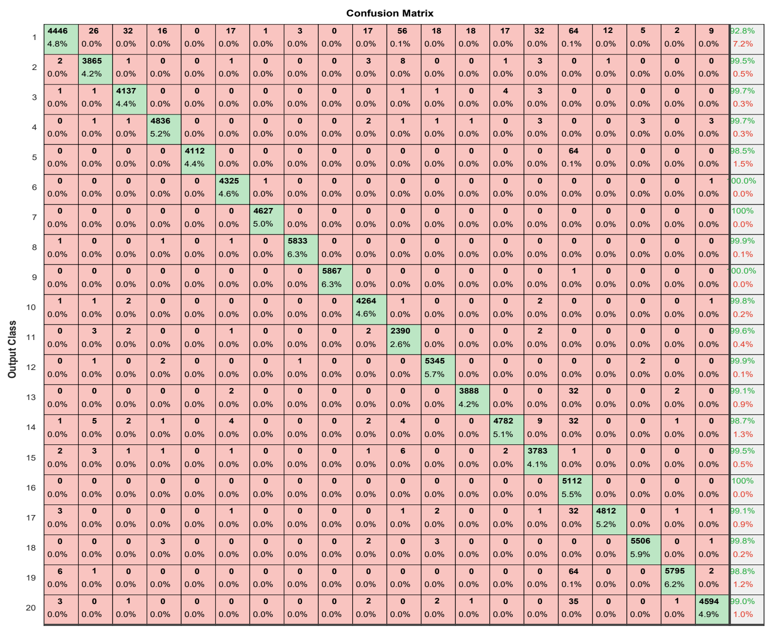
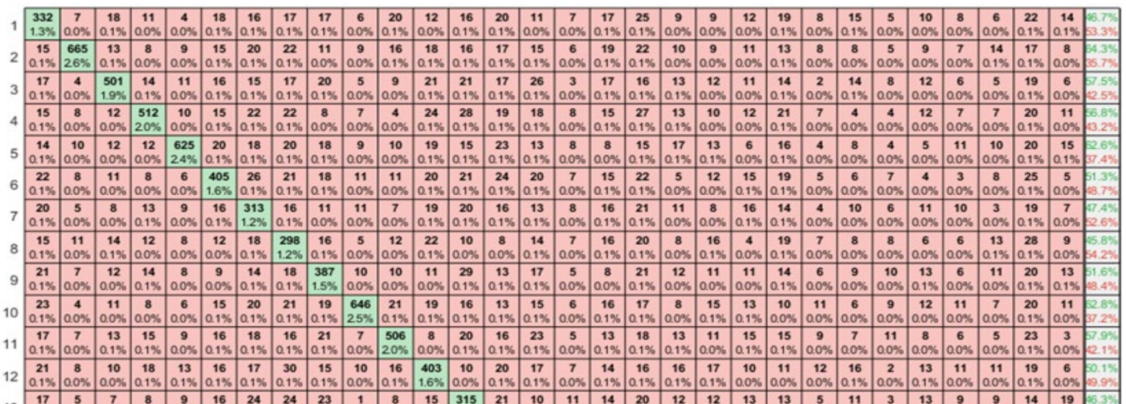


Figure 10: Recognition Confusion matrix of Chalearn2014 using adaptive AdaBoost.

After excluding the class that achieved full accuracy (one out of 20 classes in Chalearn2014), we applied adaptive AdaBoost to the remaining ten models and achieved a maximum accuracy of 99.2% after 4,000 iterations (see the confusion matrix in Figure 10). This metric provides insights into the effectiveness of the proposed system across all twenty hand gesture classes. Notably, the first class (representing the "Fame" gesture) achieved full recognition performance, while the remaining classes had lower accuracy. In ChalearnIsoGD, we applied adaptive AdaBoost without excluding any class, even though we did not find any class with full accuracy. After 4,000 iterations, we achieved a final accuracy of 56.9%, as shown in the confusion matrix of Figure 11.



.

Figure 11: Rercognition Confusion matrix of IsoGD using adaptive AdaBoost.

Considering the adaptive nature of the AdaBoost method proposed in this study, which aims to improve recognition speed, we have presented Table 3, showcasing the execution time for data recognition. The table includes results obtained by using all classes, as well as results obtained by excluding the class with the highest accuracy performance. This comparison was conducted on both datasets utilized in this research.

Table 3: Effect of the adaptive AdaBoost in the two datasets.

No. of shared classes in training the adaptive AdaBoost	Execution time of Chalearn2014 in sec.	Execution time of ChalearnIsoGD in sec.
All	34.83 Sec.	36.61 Sec.
Excluding one class of the highest accuracy	30.01 Sec.	34.20 Sec.
Excluding two classes of the highest accuracy	28.21 Sec.	31.32 Sec.

Table 3 shows that excluding the highest accuracy class from AdaBoost training speeds up the algorithm. Furthermore, a Jaccard index comparison (35) with previously published work on the test data of Chalearn2014 was presented in Table (4), along with the proposed network models of a multi-stream of different recurrent neural networks. In addition, the application of the decision system that integrates all the proposed models is done in two steps: first, using the decision system with the highest repetition score (32), and second, using the full adaptive AdaBoost of all models.

Table 4: proposed method comparison on Chalearn2014 with other works.

Method name	Jaccard
(CNN_RNN+CNN_RBFN_RNN+CNN_SOM_RNN)_LSTM Fig. 5	0.9211
(CNN_RNN+CNN_RBFN_RNN+CNN_SOM_RNN)_GRU Fig.5	0.9191
Decision system of highest repetition score Fig.6, expression(32)	0.9303
Adaptive Decision system Proposed work Fig.6,Algorithm (2)	**0.937**
Motion Dynamic+Fution [31]	0.923
Temp Conv + LSTM [32]	0.906
(Multi-Scale DNN) [33]	0.880
(AdaBoost, HoG) [34]	0.822
(MRF, KNN, PCA, HoG) [35]	0.826

Table 4 in this study presents a comprehensive comparison of individual models and the integrated approach, demonstrating the effectiveness of our proposed method in addressing the challenge of overlapping hand gesture recognition. The table also provides a performance comparison with other established methods. However, it is important to clarify that our primary objective was not solely focused on direct competition with existing works. Instead, our main emphasis was on highlighting and validating the effectiveness of the distinctive features presented in our approach.

Specifically, our approach targets the issue of overlap between similar image frames within a single clip for hand gestures belonging to different categories. This specific challenge had not been adequately addressed by any of the methods included in the comparison. Therefore, our focus was on introducing novel techniques to overcome this particular limitation and demonstrate their efficacy in improving overall recognition accuracy. By highlighting the unique contributions of our approach, we aimed to provide valuable insights into the field of hand gesture recognition and showcase the effectiveness of our proposed techniques.

Conclusions and Future Works

In this study, we successfully integrated multiple recurrent neural network predictors in an adaptive decision system that significantly enhances the performances of hand gesture recognition in videos. By training a recurrent neural network with composite features derived from video gestures, we demonstrated effective improvement in recognition performance. The choice of features utilized in constructing these composite features played a crucial role in achieving the overall recognition accuracy.

Moreover, we presented models of recurrent neural networks that share composite features to generate new frame-wise features by leveraging enhanced CNN features recovered from RBF and SOM networks. These network models exhibited great potential in enhancing the performance of features obtained through CNN training. Additionally, we discovered that combining CNN features with the improved extracted features can further enhance recognition by training a recurrent neural network capable of sharing the time with the features extracted from both CNN and the proposed pipeline models of training different neural networks. Our observations revealed that the spatiotemporal responses of RNN to the presented pipelines of enhanced features have diverse and significant effects on video recognition.

Our research findings indicate that different training streams of time-sharing neural networks can greatly improve recognition performance. We demonstrated this through various models of multi-stream recurrent networks, where time-sharing networks were scaled up by sharing a set of training pipelines from other time-sharing neural networks. Moreover, the integration of prediction outcomes of all proposed models by the introduced decision system showcased the efficient improvement in recognition performance. This is achieved by utilizing multiple training stream networks with other different single stream networks, especially in overlapped hand gestures.

While no significant differences were observed in the training performances of each type of temporal sharing recurrent neural network (RNN, LSTM, or GRU), our experimental results suggest that these subtle differences can still be leveraged to enhance the performance of the proposed decision system. The varied and distinct prediction values generated by each model contributed to the successful training of the proposed adaptive decision system of reduced execution time.

The significance of this research extends beyond controlled environments, as it addresses the real-world applicability of the developed system and its performance in diverse, practical scenarios. By effectively overcoming the challenges associated with hand gesture recognition, our work opens new avenues for the development of advanced human-machine interfaces and paves the way for improved communication through hand gestures. The adaptive decision-making system and the incorporation of diverse neural network architectures enhance the system's ability to perform accurately in real-world situations, where hand gestures may vary in complexity, context, and environmental conditions. This research contributes to the practical implementation of robust hand gesture recognition systems that can be applied in a wide range of applications, including interactive technologies, virtual reality, augmented reality, and assistive devices.

One significant challenge encountered during the implementation of this work was the limited availability of physical resources, particularly working memory. This issue primarily arose during the training phase when training neural networks with a single batch of data, as observed in the training of both the Self-Organizing Map (SOM) and Radial Basis Function (RBF) networks. Training these networks with a single batch of data often led to the depletion of the computer's workspace allocated for training. In contrast, training in the form of data packets, as seen in CNN training, provided better utilization of storage space for training purposes. Furthermore, it was noted that while the

CNN network's recognition performance improved when trained on a complete sequence of images rather than a single image frame, the RNN network exhibited greater effectiveness when trained to leverage the temporal properties extracted from the CNN network trained on an image frame. This observation can be attributed to the relatively short length of the video capturing the single hand gesture used in this study. To address the limitations faced in this work, we can focus on improving the performance of each neural network involved through extensive research and development. The existing wealth of research offers numerous avenues for enhancing and enriching the capabilities of these networks. Furthermore, in this work, the proposed time-shared neural network models, including both single and multi-stream models, were trained using composite features. These composite features were obtained by concatenating a set of features from different networks. The study found that increasing the size of the composite feature component resulted in improved recognition performance in the network. However, this increase in size also led to higher storage space and workspace consumption. To address this challenge, it is recommended to explore methods for compressing or reducing the size of the resulting composite feature. By finding ways to install or represent these features more efficiently, the overall size of the resulting feature can be shortened. This would help optimize storage space and workspace requirements without compromising the recognition performance of the network.

In conclusion, this research highlights the importance of a comprehensive training pipeline and the integration of diverse neural network models to achieve substantial improvements in hand gesture recognition performance.
These findings provide valuable insights for future research in this field, guiding the development of more effective and robust recognition systems either by discovering other ensemble learning techniques that combine multiple weak classifiers to create a strong classifier or by exploring other neural network of diverse design. Additionally, this research contributes to our understanding of sign language as a unique form of communication that relies on a combination of signs, facial expressions, and body language to convey meaning. This poses challenges for sign annotation and translation. The insights gained from this study can aid in addressing these challenges and advancing sign language recognition technologies.
The discoveries made in this research have broad implications for various realms of human-machine interactions. They can be applied in gesture-based control systems, virtual reality, augmented reality, sign language recognition, and human-robot interactions. By enhancing the accuracy and robustness of hand gesture recognition, these findings open up new possibilities for more intuitive and efficient communication between humans and machines, enabling seamless human-machine interfaces. Overall, this research not only contributes to the field of hand gesture recognition but also has the potential to revolutionize how humans interact with machines, paving the way for innovative applications and advancements in diverse areas of human-machine interaction.

Declaration of competing interest

Authors certify that they have no affiliations with or involvement in any organization or entity with any financial interest or non-financial interest in the subject matter or materials discussed in this manuscript.

Declaration of Funding

The authors did not receive support from any organization for the submitted work.

Authors' contributions

Anwar Mira, under the supervision of Prof. Olaf Hellwich, was responsible for conceiving, designing, and programming the experiments, performing the experiments, analyzing and interpreting the data, and writing the paper.

Availability of data and materials

The original data used to support the findings of this study is available on Kaggle (www.kaggle.com), a renowned online platforme for data scientists and machine learning engineers. Interested individuals can request access to the data from the community through the platform.

Acknowledgements

We extend our sincere gratitude to Prof. Dr. Eman Salih Al-Shamery from the University of Babylon for her remarkable support and unwavering dedication during the analysis of the data set and proposal development process. Her invaluable expertise and insightful contributions have played a pivotal role in ensuring the alignment and coherence of our work.

References

1. Shahhosseini, Mohsen, Guiping Hu, and Hieu Pham. "Optimizing ensemble weights and hyperparameters of machine learning models for regression problems." *Machine Learning with Applications* 7 (2022): 100251.https://doi.org/10.1016/j.mlwa.2022.100251
2. Arun Prakash, J., Asswin, C., Ravi, V. *et al.* Pediatric pneumonia diagnosis using stacked ensemble learning on multi-model deep CNN architectures. *Multimed Tools Appl* **82**, 21311–21351 (2023). https://doi.org/10.1007/s11042-022-13844-6
3. Haq, Amin Ul, Jian Ping Li, Zafar Ali, Inayat Khan, Ajab Khan, M. Irfan Uddin, Bless Lord Y. Agbley, and Riaz Ullah Khan. "Stacking approach for accurate invasive ductal carcinoma classification." *Computers and Electrical Engineering* 100 (2022): 107937.
4. Dong, Yingchao, Hongli Zhang, Cong Wang, and Xiaojun Zhou. "Wind power forecasting based on stacking ensemble model, decomposition and intelligent optimization algorithm." *Neurocomputing* 462 (2021): 169-184.
5. Ouyang, Shuyi, Hongyi Wang, Ziwei Niu, Zhenjia Bai, Shiao Xie, Yingying Xu, Ruofeng Tong, Yen-Wei Chen, and Lanfen Lin. "HSVLT: Hierarchical Scale-Aware Vision-Language Transformer for Multi-Label Image Classification." In *Proceedings of the 31st ACM International Conference on Multimedia*, pp. 4768-4777. 2023.https://doi.org/10.1145/3581783.3612159
6. M. A. Eldosoky, J. P. Li, A. Ul Haq, and F. Zeng, "Hierarchical Multi-scale Visual Attention Module for Wall Putty Bulge Terminals Detection," 2023 IEEE 6th International Conference on Pattern Recognition and Artificial Intelligence (PRAI), Haikou, China, 2023, pp. 638-643, doi: 10.1109/PRAI59366.2023.10332002.
7. Cai, Yuqi, Wujie Zhou, Liting Zhang, Lu Yu, and Ting Luo. "DHFNet: Dual-decoding hierarchical fusion network for RGB-thermal semantic segmentation." *The Visual Computer* (2023): 1-11.
8. Fu, Y., Wu, D., & Boulet, B. (2022). Reinforcement Learning Based Dynamic Model Combination for Time Series Forecasting. *Proceedings of the AAAI Conference on Artificial Intelligence*, *36*(6), 6639-6647. https://doi.org/10.1609/aaai.v36i6.20618
9. Guo, Zhaohan Daniel, Bernardo Avila Pires, Bilal Piot, Jean-Bastien Grill, Florent Altché, Rémi Munos, and Mohammad Gheshlaghi Azar. "Bootstrap latent-predictive representations for multitasking reinforcement learning." In *International Conference on Machine Learning*, pp. 3875-3886. PMLR, 2020.

10. Chen, Chao, and Hui Liu. "Dynamic ensemble wind speed prediction model based on hybrid deep reinforcement learning." *Advanced Engineering Informatics* 48 (2021): 101290.https://doi.org/10.1016/j.aei.2021.101290.
11. K. Lai and S. Yanushkevich, "An Ensemble of Knowledge Sharing Models for Dynamic Hand Gesture Recognition," *2020 International Joint Conference on Neural Networks (IJCNN)*, Glasgow, UK, 2020, pp. 1-7, doi: 10.1109/IJCNN48605.2020.9207601.
12. Sen, A., Mishra, T.K. & Dash, R. A novel hand gesture detection and recognition system based on ensemble-based convolutional neural network. *Multimed Tools Appl* **81**, 40043–40066 (2022). https://doi.org/10.1007/s11042-022-11909-0.
13. Wang, Yanyu, Pengfei Zhao, and Zhen Zhang. "A deep learning approach using attention mechanism and transfer learning for electromyographic hand gesture estimation." *Expert Systems with Applications* 234 (2023): 121055.doi.org/10.1016/j.eswa.2023.121055.
14. Colli Alfaro, Jose Guillermo, and Ana Luisa Trejos. 2022. "User-Independent Hand Gesture Recognition Classification Models Using Sensor Fusion" *Sensors* 22, no. 4: 1321. https://doi.org/10.3390/s22041321
15. Rouali, M.L., Boulahia, S.Y. & Amamra, A. Structure and Sequencing Preserving Representations for Skeleton-based Action Recognition Relying on Attention Mechanisms. *J Sign Process Syst* **95**, 1003–1019 (2023). https://doi.org/10.1007/s11265-023-01892-6
16. Tan, Chun Keat, Kian Ming Lim, Chin Poo Lee, Roy Kwang Yang Chang, and Ali Alqahtani. 2023. "SDViT: Stacking of Distilled Vision Transformers for Hand Gesture Recognition" *Applied Sciences* 13, no. 22: 12204. https://doi.org/10.3390/app132212204
17. Hwang, Dong-Hyun, Suntae Kim, Nicolas Monet, Hideki Koike, and Soonmin Bae. "Lightweight 3d human pose estimation network training using teacher-student learning." In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 479-488. 2020.
18. Bird, Jordan J., Anikó Ekárt, and Diego R. Faria. 2020. "British Sign Language Recognition via Late Fusion of Computer Vision and Leap Motion with Transfer Learning to American Sign Language" *Sensors* 20, no. 18: 5151. https://doi.org/10.3390/s20185151.
19. Anwar Mira,Olaf Hellwich. "An Embedded Neural Network Approach for Reinforcing Deep Learning: Advancing Hand Gesture Recognition",accepted for publishing in Journal of universal Computing (2024).
20. Orr, M. J. L. :Introduction to radial basis neural networks. Center for cognitive science, Edinburgh University, Scotland. (1996). UK. http://anc. ed. ac. uk/rbf.
21. Kohonen, T.: The self-organizing map. Proceedings of the IEEE, 78(9), 1464-1480. (1990)
22. Escalera, Sergio, Xavier Baró, Jordi Gonzalez, Miguel A. Bautista, Meysam Madadi, Miguel Reyes, Víctor Ponce-López, Hugo J. Escalante, Jamie Shotton, and Isabelle Guyon. "Chalearn looking at people challenge 2014: Dataset and results." In Computer Vision-ECCV 2014 Workshops: Zurich, Switzerland, September 6-7 and 12, 2014, Proceedings, Part I 13, pp. 459-473. Springer International Publishing, 2015.https://doi.org/10.1007/978-3-319-16178-5_32.
23. Wan, Jun, Yibing Zhao, Shuai Zhou, Isabelle Guyon, Sergio Escalera, and Stan Z. Li. "Chalearn looking at people rgb-d isolated and continuous datasets for gesture recognition." In Proceedings of the IEEE conference on computer vision and pattern recognition workshops, pp. 56-64. 2016.
24. Hastie, Trevor, Saharon Rosset, Ji Zhu, and Hui Zou. :Multi-class adaboost. Statistics and its Interface 2, no. 3 .349-360. (2009).
25. Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." Advances in neural information processing systems 25 (2012).
26. Andrea Vedaldi and Karel Lenc. Matconvnet: Convolutional neural networks for matlab. In Proceedings Of the 23rd Annual ACM Conference on Multimedia Conference, pages 689–692. ACM, 2015.
27. Chris McCormick,(15 Aug 2013) , Radial Basis Function Network (RBFN) Tutorial https://mccormickml.com/2013/08/15/radial-basis-function-network-rbfn-tutorial/.
28. Tian, Jing, Michael H. Azarian, and Michael Pecht. "Anomaly detection using self-organizing maps-based

k-nearest neighbor algorithm." In PHM society European conference, vol. 2, no. 1. 2014.

29. Mareva Brixy ,2017, Self-organising Map for handwritten number classification .GitHub https://github.com/marevab/SOM
30. Chengxi Ye, Chen Zhao, Yezhou Yang, Cornelia Fermüller, and Yiannis Aloimonos. Lightnet: A versatile, standalone matlab-based environment for deep learning. In Proceedings of the 2016 ACM on Multimedia Conference, pages 1156–1159. ACM, 2016.
31. Wang, Huogen. 2021. "Two Stage Continuous Gesture Recognition Based on Deep Learning " Electronics 10, no. 5: 534. https://doi.org/10.3390/electronics10050534
32. Pigou, L., Van Den Oord, A., Dieleman, S., Van Herreweghe, M., & Dambre, J. : Beyond temporal pooling: Recurrence and temporal convolutions for gesture recognition in video. International Journal of Computer Vision, 126(2), 430-439. (2018).
33. Neverova, N., Wolf, C., Taylor, G.W., Nebout, F. Multi-scale Deep Learning for Gesture Detection and Localization. In: Agapito, L., Bronstein, M., Rother, C. (eds) Computer Vision - ECCV 2014 Workshops. ECCV 2014. Lecture Notes in Computer Science(), vol 8925. Springer, Cham. https://doi.org/10.1007/978-3-319-16178-5_33. (2015).
34. Monnier, Camille, Stan German, and Andrey Ost. :A multi-scale boosted detector for efficient and robust gesture recognition. In Computer Vision-ECCV 2014 Workshops: Zurich, Switzerland, September 6-7 and 12, 2014, Proceedings, Part I 13, pp. 491-502. Springer International Publishing.(2015).
35. Chang, Ju Yong. :Nonparametric gesture labeling from multi-modal data. In Computer Vision-ECCV 2014 Workshops: Zurich, Switzerland, September 6-7 and 12, 2014, Proceedings, Part I, pp. 503-517. Cham: Springer International Publishing. (2015).