Sign Language Translation Using Deep Convolutional Neural Networks

Rahib H.Abiyev[1*], Murat Arslan[1] and John Bush Idoko[1]
1Applied Artificial Intelligence Research Centre, Department of Computer Engineering, Near East University, North Cyprus, Mersin-10, Turkey
[e-mail: rahib.abiyev@neu.edu.tr, murat.arslan@neu.edu.tr, john.bush@neu.edu.tr]
*Corresponding author: Rahib H.Abiyev

Received June 3, 2019; revised Septebmer 3, 2019; accepted October 6, 2019; published February 29, 2020

Abstract

Sign language is a natural, visually oriented and non-verbal communication channel between people that facilitates communication through facial/bodily expressions, postures and a set of gestures. It is basically used for communication with people who are deaf or hard of hearing. In order to understand such communication quickly and accurately, the design of a successful sign language translation system is considered in this paper. The proposed system includes object detection and classification stages. Firstly, Single Shot Multi Box Detection (SSD) architecture is utilized for hand detection, then a deep learning structure based on the Inception v3 plus Support Vector Machine (SVM) that combines feature extraction and classification stages is proposed to constructively translate the detected hand gestures. A sign language fingerspelling dataset is used for the design of the proposed model. The obtained results and comparative analysis demonstrate the efficiency of using the proposed hybrid structure in sign language translation.

Keywords: Sign language, single short multi-box detector, convolutional neural networks, support vector machine

1. Introduction

Sign language is a communication medium that uses facial/bodily expressions, postures and a set of gestures in human-human communication, as well as through TV and social networks. Sign language is utilized as the first language by millions of deaf people (hearing impaired people), in addition, hard-of-hearing people, and people who have various speaking difficulties. In accordance with the investigation conducted by the British Deaf Association, it is recorded that about 151,000 people use sign language as a means of communication [1]. There is no universal sign language and almost every country has its own national sign language and fingerspelling alphabet. They use a combination of manual gestures with facial mimics and lips articulation. These sign languages have a special grammar that has fundamental differences to speech-based spoken languages.

One of the popular sign languages is American Sign Language (ASL), which has its own rules and grammar. There are sign systems such as Signed English that borrow signs from the ASL, but utilize them in English language order [2]. As sign language involves both reading the signs (receptive skills) and rendering the signs (expressive skills), it is a two-way process. Translation and recognition of sign language is an important research area because it integrates hearing impaired people into the general public and provides equal opportunities for the entire population. The development of a human-machine interface that has the capability to enhance the common correspondence amongst healthy and hearing impede individuals is a significantly important problem, targeted at supplanting the third human factor (translator). The sign language recognition problem is often limited to the translation of fingerspelled words to text, where recognition of the Sign Language alphabet is the major task [3]. Characterized by their own rules and grammar, Sign Languages are comprised of a dynamic configuration of a set of palm and hand gestures positions, body movements, and finally, facial expressions [4]. For most if not all known natural dialects/languages, there are different signs.

Sign language as a visual form communication of information between people is detailed and rapid. Both spelling and accurate translation of thoughts and feelings in a short time are very important. Since some people do not understand sign language, and some people usually find it very difficult to understand, it has become important to design a vision-based sign language translator. The design of such a system allows the communication barrier between people to be significantly reduced. There are two major methods for sign language translation. The first one is vision-based methods which utilize installed camera(s) to capture the target images which are in turn fed into the image analysis module [5-8]. The second approach is a glove-based approach which utilizes sensors and gloves for implementation. Here, the additional hardware (glove) is employed to mitigate the challenges of the traditional vision-based methods. Even though signers/users often find glove-based approaches to be burdening and obtrusive, they give more accurate and precise results [9, 8]. In this research, we propose a vision-based sign language translation method which uses a single video camera to dynamically capture hand gestures.

In the literature, several approaches have been presented for the purpose of sign language gesture identification. In the early 2000s, sensor-based approaches with neural networks and Bayesian networks were explored [10-12]. Cheap wearable technologies such as wearable sensor gloves are utilized to get the relative gesture of hands and fingers in order to predict sign language [12]. The use of constrained grammars and coloured gloves produced low error rates on both training and test data [13]. Using sensor devices, a multimodal framework is

applied for isolated sign language translation [14]. The sensors are used to capture finger, palm positions and then the Hidden Markov Model (HMM) and Bidirectional Long Short-Term Memory Neural Network (BLSTM-NN) are used for classification purpose. In-depth knowledge of sign language can lead to a better understanding of the gesture classification problem. In this regard, Bheda et al. [15] addressed the gesture classification problem using a deep convolutional neural network. In some studies, the colour and depth of images are utilized for recognition purposes. Here, Ameen et al. classified ASL using a convolutional neural network with the colour and depth of images, and in their experiments, they obtained 80% recall and 82% precision rates [16]. Another widely explored classifier for gesture and posture is the linear classifier. The structure is relatively simple as compared to Bayesian networks, and they frequently produce high accuracies [7]. The paper [18] presents a sign language recognition system based on segmentation, tracking, feature extraction and classification of hand gestures. Euclidian distance is applied for the classification of features. In [19], the "likelihood of hidden Markov model" is presented for sign language translation. In addition to HMM, the paper [20] used an independent Bayesian classification combination for improving recognition accuracy. In [21], facial expressions are recognised and used in sign language communication. Probabilistic Principal Component Analysis model is integrated with the recursive tracking schemes for feature extraction. The recognition of tracked results is performed using HMM and SVM. The paper uses skin colour and texture attributes with neural networks to separate the hand from the background [22]. KNN and SVM classifiers are applied for recognition purposes. The construction of a mobile application using a speech-based system to translate text from Indian Sign Language is described in [23]. Here, the authors implemented the model using a pre-built domain of images stored locally on a machine and queried it during execution [23].

The classical approach used for the recognition of sign language is fundamentally based on feature extraction and classification. In the paper, these two stages are combined in a convolutional neural network (CNN) for designing the sign language translation system. The presented approach simplifies the implementation of the sign language recognition system. Today, CNN is actively used for solving different problems. These are human activity recognition [24], vehicle detection in aerial images [25], detection of smoke as a moving object [26], detection of network intrusion [27], and identification of tomato nutrition disorders [28].

In the paper, the proposed sign language translation (SLT) includes three basic modules: object detection, feature extraction and classification. To solve these problems, the integration of three powerful models- SSD, Inception 3 and SVM is proposed. These algorithms are applied for object detection, feature extraction and classification purpose. The criteria presented for the system were robustness, accuracy, high speed.

There have been designed a set of techniques for object detection. The more used techniques are Viola-Jones algorithm [29], histograms of oriented gradients (HOG) [30], recognition using regions [31], R-CNN [32,33], "You only look once" (YOLO) [34,35], single-shot multibox detection (SSD) [36] techniques. Viola-Jones algorithm [29] is based on Haar feature selection used for different parts of images. The algorithm use Adaboost training and cascade clustering architecture. The algorithm has good feature selection properties. One of the disadvantages of the algorithm is that it is sensitive to lighting conditions and possibly detects different degree of the exact object due to subwindows overlapping. The algorithm is not effective in detecting titled or turned images. Next algorithm HOG [30] significantly outperformed Viola-Jones algorithm in this task. The algorithm uses handcoded features. For every pixel, the surrounding pixels are selected and the direction (arrow) showing change of

colour of darker region is determined. For each pixel this process which is called gradient is repeated. The whole image is replaced with the arrows (directions) which are characterized by histogram of gradient (HOG). Despite being good in many applications, it still used handcoded features which failed in a more generalized setting with much noise and obstacles in the background.

The conventional object detection methods are built on handcrafted features and shallow trainable architectures. They have difficulties in constructing more complex systems combining multiple low- level image features with high-level context. One powerful approach that is able to learn semantic, high-level and deeper features is the development of deep learning structures for object detection. Recently, deep learning-based methods such as R-CNN, Faster R-CNN, YOLO, SSD algorithms [32-37] are also applied for object detection. R-CNN uses Selective Search to create bounding boxes (or region proposals) [38]. The selective search takes the image of different sizes, and for each size tries to group together adjacent pixels using texture, colour, or intensity in order to identify objects. Then for each bounding box using CNN, image classification is performed. The algorithm has some disadvantages. Used selective search is fixed algorithm that does not use learning and this may lead to the generation of bad candidate region proposals. The algorithm also takes a huge amount of time for training of network that classifies many regions of proposals and because of this, the algorithm cannot be implemented in real-time. Later, a faster version of the R-CNN algorithm [33] that uses CNN instead of selective search is designed in order to solve above-mentioned problems. But faster version requires many passes (systems) through a single image in order to extract all the objects. The performance of the system depends on how the previous system is performed.

Next algorithm YOLO (You Only Look Once) [34] actually looks at the image just once but in a clever way. The algorithm divides up the image into a grid of *S*x*S* cells. Each of these cells is responsible for predicting *m* bounding boxes that enclose some objects. For each bounding box, the cell also predicts a class. The predicting of bounding boxes is performed by calculating the confidence score. The architecture of YOLO is based on CNNs. An input image given to the YOLO is processed by the convolutional network in a single pass, and at the end of the network, a tensor describing the bounding boxes for the grid cells are derived. After determining the final scores for the bounding boxes the outputs are determined. YOLO is a simple and fast algorithm. One limitation of YOLO is that it struggles with small objects within image, there may be difficulties in detecting a floc of birds. This is due to the spatial constraints of the algorithm. Later, a faster version of YOLO algorithm was developed, but it is less accurate than the first version.

SSD [36] is based on CNN that produces a fixed-size collection of bounding boxes. In these boxes, by scoring the presence of object class instances the final detection of objects is implemented. The network is the object detector that classifies the detected objects. The network uses sub-components such as Multibox, Priors, Fixed priors. In the network structure, a set of new layers (SSD) and new faster R-CNN modules or some of their combination are used to replace Convolution/Pooling layers. Using SSD a better balance between swiftness and precision is achieved. By running a convolutional network only one time, SSD determines a feature map of the input image. SSD also uses anchor boxes at a variety of aspect ratio comparable to Faster-RCNN and learns the off-set to a certain extent than learning the box [36]. After multiple convolutional layers, SSD predicts the bounding boxes in order to hold the scale. Since every convolutional layer functions at a diverse scale, it is able to detect objects of a mixture of scales. In the paper, we use SSD based on CNN. SSD is faster than YOLO, and more accurate than Faster R-CNN. More detailed comparisons of object detection methods are

provided in the papers [36,37]. From these comparative results, it was clear that the SSD method has better performance than other methods.

After detection of images, the feature extraction and classification algorithms are used to extract and classify more important features of images. There are a set of methods used for feature extraction and classification. Recently, principal component analysis, local binary patterns, Gabor filters, Scale Invariant Feature Transform (SIFT), speeded up robust features (SURF) semantic analysis, independent component analysis, histogram of gradient are widely used for feature extraction [39,40]. The extracted features are used in classification. Conventional classification algorithms are based on k-means, linear discriminant analysis, c-means, supervised clustering, fuzzy c-means, etc [40]. Some studies address the limitations of the existing conventional tools. Nowadays machine learning techniques are extensively used for feature extraction and classification purpose. These are neural networks, SVM, radial based networks, neuro-fuzzy networks, different types of deep learning structures. The integration of deep learning structures and SVM [41] are becoming extensively used for solving feature extraction and classification problems. In the paper [42], a 2D convolutional layer based CNN architecture, Fisher ratio (F-ratio) based feature selection and SVM classifier are used for P300 detection. A new deep architecture that uses SVMs with class probability output networks is presented in [43] in order to provide better generalization power for pattern classification problems. The paper [44] presents a combination of deep belief network and kernelized SVM for classification multiclass dataset. [45] proposed a deep ranking structural Support Vector Machine with deep learning for image tagging. In the paper [46] integration of deep learning and SVM is proposed for acquisition of deep features afterwards, standard SVM is used for classification. The paper [47] proposes deep neural mapping support vector machine which is trained by gradient descent. In [48] a combination of CNN and SVM is presented for Grapevine variety identification.

In feature extraction and classification, the speed, sensitivity, occlusion and accuracy of the system are very important. In this paper, we propose a sign language translation model based on hybrid structure that uses SSD to detect hand gestures, and uses Inception v3 plus SVM to obtain features for classification purposes. The used SSD is one of the fastest algorithms with high accuracy proposed for detection of hand gestures, the Inception v3 module is a convolutional neural network (CNN) that extracts feature matrices from the detected hand gestures to higher dimensional spaces for further examination. After this these are incorporated with the SVM classifier that performs sign classification. Integration of these structures allows us to design fast, robust SLT system with high accuracy. The outcome of the proposed hybrid model has shown the efficiency of the proposed model in the execution of the sign language translation problem.

The remaining sections of the paper are organized as follows. Section 2 discusses the proposed algorithm and other deep learning approaches. Section 3 presents the simulation results and discussions. Finally, the conclusion and further thoughts are presented in Section 4.

2. Sign Language Translation System

2.1 Structure of the System

The proposed translation system includes three distinctive structures - SSD, Inception v3 and SVM that are integrated to form the hybrid model that constructively translates sign gestures **(Fig. 1)**. In this paper, for designing such a translation system, the American Sign Language (ASL) fingerspelling dataset is utilized. In this hybrid model, SSD is utilized for

hand detection, Inception v3 is applied for feature extraction and the last module, SVM, is utilized for classification purposes.

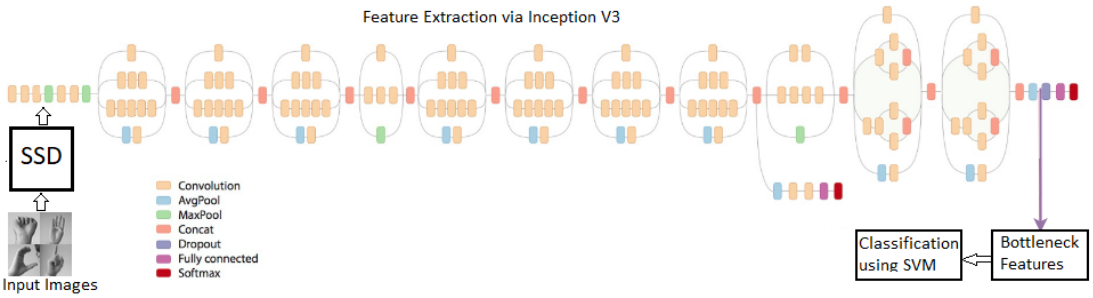


Fig. 1. Structure of the proposed model

Inception v3 which is based on CNNs is a basic module of the translation system which is used to extract features for future classification purpose. CNNs are types of deep learning architecture that have one or more convolution, pooling and feedforward layers. They are a biologically-inspired variation of the multilayer perceptron (MLP). Every neuron in an MLP has its own weight vector, but neurons in CNNs share weight vectors and the sharing of weights reduces the number of trainable weights. By using the weight sharing technique, neurons perform convolutions on the input data with the convolution filters. The obtained output features from the convolutional layers are fed to the ReLU layer. After applying an activation function $f(x)=max(0,x)$ to the obtained feature map, the obtained signals are entered into the pooling layer. After several layers of convolution and pooling, the image size is reduced and more complex features are extracted. Subsequently, the contents are moved into a one-dimensional vector with a sufficiently small feature map and fed into the classification module. This module calculates the output of CNN.

CNNs have convolutional layers that are defined by an input map I, biases b and a bank of filters [49,50]. Assume that $l = 1$ is the first layer and $l = L$ is the last layer and x is the input with H x W dimensions, which has i and j as the iterators. The kernel ω with $k_1 x k_2$ dimension has m by n as the iterators. $\omega^l_{m,n}$ is the weight matrix which connects the l layer's neurons with l-1 layers' neurons. b^l is the bias unit at layer l. At layer l, the convolved input vector $x^l_{i,j}$ plus bias is represented as:

$$x^l_{i,j} = \sum_m \sum_n \omega^l_{m,n} o^{l-1}_{i+m,j+n} + b^l \tag{1}$$

The output of the convolutional and pooling layers is determined as

$$o^l_{i,j} = pool(f(\sum_m \sum_n \omega^l_{m,n} o^{l-1}_{i+m,j+n} + b^l)) \tag{2}$$

After pooling, the flatten operation that concatenates the obtained features has been carried out using $o^L_{flatten} = flatten(o^L_{i,j}))$. The obtained feature vector is transformed into model outputs using a fully-connected layer $y=F(o^L_{flatten})$.

After obtaining the output signals, the training of the unknown parameters (weight coefficients) of CNN is started. Let's denote the unknown parameters of CNN by θ. An appropriate loss function is constructed in order to determine the accurate values of parameters

θ. This is implemented by minimizing the loss function through the usage of input-output training pairs $\{(x^{(i)},y^{(i)}); i \in [1,\ldots,N]\}$. Here, $x^{(i)}$ is the *i-th* input data, and $y^{(i)}$ is its corresponding output target data. If we denote the current output of CNN as $o^{(i)}$, then the loss of CNN can be determined as:

$$L = \frac{1}{N}\sum_{i=1}^{N} l(\theta; y^{(i)}, o^{(i)}) \tag{3}$$

The training of the CNN parameters is carried out through minimization of the loss function. In the course of the training, the values of the parameters are determined. Updating of the parameters of the network is performed using an Adam optimizer (adaptive moment estimation) [51]. Adam optimizer uses first-order gradients of the loss function for parameter learning. The method is a stochastic optimization that uses first and second moments of the gradients to calculate individual adaptive learning rates for different parameters [51].

Large quantities of training data are necessary for the efficient training of deep CNNs. Data augmentation is applied to solve this problem. This procedure also mitigates the relative data scarcity to equate the number of parameters of the CNN. Here, the explored data augmentation transforms the existing data into new data without changing their nature. For data augmentation, some geometric transformations such as rotating, shifting, shearing, zooming and mirroring are used. In the paper, we used Inception v3 as a CNN model for feature extraction. The descriptions of Inception v3 is given in section 2.4.

2.2 Dataset Analysis

To evaluate the proposed hybrid model, we utilized the 'Kaggle' American Sign Language fingerspelling dataset. This database is comprised of 24 classes of signs or letters. All the letters of the English alphabet are in the database except *Z* and *J*. This is because *Z* and *J* do not have static postures. Training and test sets consist of 27,455 and 7,172 items respectively. The data is provided in its raw form as a pixel to pixel intensity [0-255] class-wise distributed XLS files. To achieve high performance, each of the images is converted into a 28x28 grayscale image. As demonstrated in the dataset description file, the images have been modified with at least 50+ variations. For instance, three degrees rotation, +/- 15% contrast/brightness/, 5% random pixelation, and so on. When exploring this domain, researchers face numerous difficulties as a result of these modifications, which in turn alter the resolutions of the images. Fragments of the explored data are demonstrated in **Fig. 2**.

2.3 Single Shot Multibox Detector

In this paper, the Single Shot Multibox Detector (SSD) [36] is used to detect the hand, after which the detected hand acts as the input to the classification system. SSD is the machine learning model used by C. Szegedy et al. 2016 [52] for object detection problems where it recorded high precision and performance. In this paper, the structure of SSD is based on a VGG-16 structure, as demonstrated in **Fig. 3**. In the classification system, a set of convolutional layers is added for feature extraction and fully connected layers are removed.

Fig. 2. Fragment of ASL fingerspelling dataset

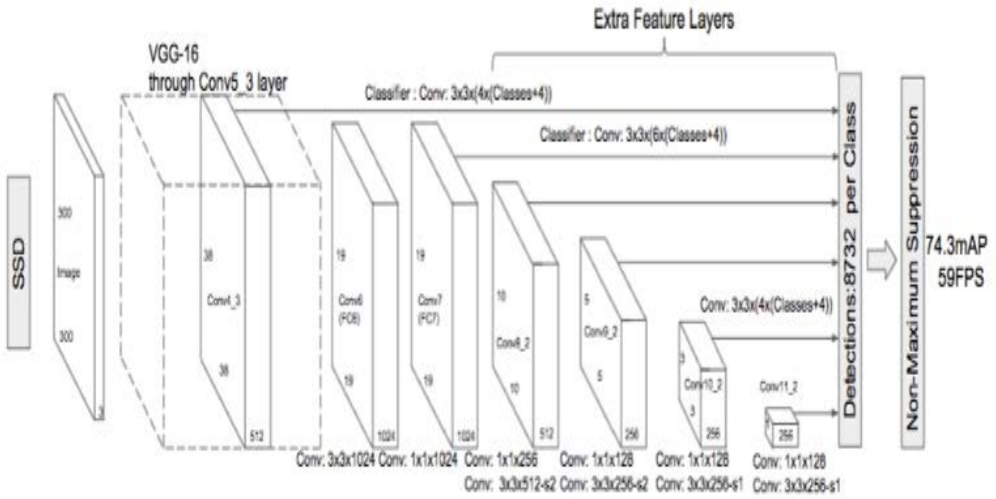


Fig. 3. SSD network structure

By doing so, the CNN extracts useful features on multiple scales and gradually reduces the size of the input to each of the subsequent layers. In Multi Box, the pre-calculated, constant dimension-limiting prerogatives closely matching the distribution of the original ground truth boxes are searched. In fact, these priorities are chosen such that the Intersection ratio over Union rate is greater than 0.5.

As demonstrated in **Fig. 4** below, an IOU of 0.5 is still not adequate, but the limiting box provides the regression algorithm with a strong starting point. For this reason, the Multi Box starts with its priorities as estimators and tries to stretch closer to the bounding boxes of the ground truth.

In conclusion, the multi-box of the SSD retains the top *K* predictions, which minimize both location and confidence losses.

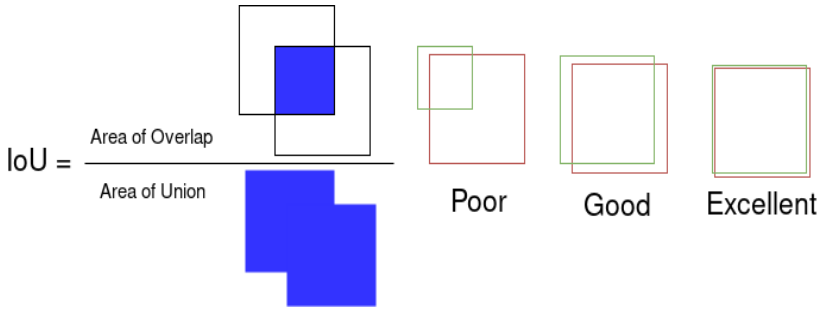


Fig. 4. Box overlapping generated by SSD structure

2.4 Inception v3

In this paper, we explore the Inception v3 CNN model for feature extraction. As seen in the convolutional layers of the traditional CNN structure, this model handles both data pre-processing and feature extraction. The inception deep convolutional structure was introduced as GoogLeNet [52] and referred to as Inception v1. After this, the Inception v1 structure was reconstructed considering several factors. At first, batch normalization [53] was introduced after which the architecture was renamed as Inception v2. Furthermore, in the third iteration, additional factorization ideas [54] was introduced and the new architecture is called Inception v3. Thus, while Inception v2 is made of batch normalization, Inception v3 is made of factorization ideas. The Inception v3 structure comprises symmetric and asymmetric building blocks, including convolutions, average pooling, max pooling, concats, dropouts, and fully connected layers. However, in this paper, we replace the fully connected layers with an SVM classifier. The Inception v3 architecture incorporates factorization of initial convolutions into smaller convolutions and the addition of batch normalization to the fully connected layer correspondingly. The Inception model architecture is predominantly a 299x299x3 input, which represents a field of 299 pixels and 3 channels representing the standard RGB image, converged with a set of convolutional layers, a series of max-pooling operations, and sequential inception modules stacks (set of different convolution filters and max-pool filter) performing concatenation of filters. Ultimately, the output is a softmax layer. Usually, a 2048-dimensional vector serves as the input to the top layer of the Inception v3 model where the softmax layer is trained. For instance, a softmax layer with X labels learns $X + 2048*X$ parameters with respect to the learned weights and biases.

In Inception v3, the aim of factorizing convolutions is to minimize the number of connections/parameters without decreasing the network efficiency [47]. For example, **Fig. 5** depicts two 3×3 convolutions replacing one 5×5 convolution. As demonstrated in the figure, by using 1 layer of 5×5 filter, the number of parameters = 5×5=25 and by using 2 layers of 3×3 filters, the number of parameters = 3×3+3×3=18. This implies that further factorization of the filter of the same layer reduces the number of parameters leading to an increase in learning speed and network efficiency. With the factorization technique, the number of parameters is reduced for the whole network. It is less likely to be overfitting, and consequently, the network can go deeper.

In the case of asymmetric convolutions factorization, let's consider one 3×1 convolution and one 1×3 convolution replacing one 3×3 convolution, as depicted in **Fig. 6**. By using a 3×3 filter, the number of parameters will be equal to 3×3=9, and by using 3×1 and 1×3 filters, the number of parameters becomes 3×1+1×3=6. It can be questioned why two 2×2 filters are not used to replace one 3×3 filter. If two 2×2 filters are used, the number of parameters becomes

2×2×2=8. Comparing the rate of reduction of the parameters between two 2×2 filters and one 3×1 and 1×3 filters, it is evident that the latter produces a lower number of parameters as compared to the number of parameters produced by the former.

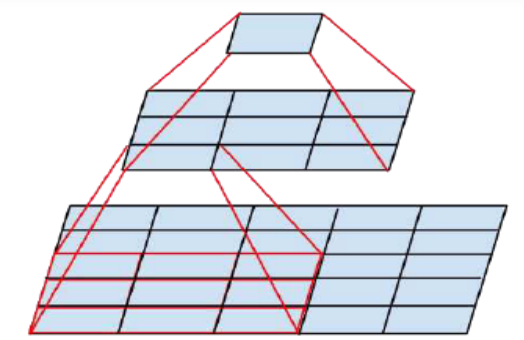


Fig. 5. Two 3×3 convolutions replacing one 5×5 convolution

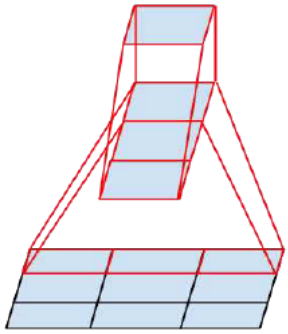


Fig. 6. One 3×3 convolution replaced by one 3×1 convolution with one 1×3 convolution

2.5 Support Vector Machine

The support vector machine (SVM) uses a learning technique that finds a hyperplane. This hyperplane produces the best separation using the largest distance to the nearest training data point of any of the classes [55]. In many experiments, the support vector machine has proven to be successful, especially when dealing with a high-dimensional feature space, as demonstrated in [56]. Using a hyperplane, SVM in its simplest structure can classify data into two classes (binary classification). The support vector machine maximises the margin between the closest samples to the hyperplane (the support vectors) using an optimization strategy. SVM can also support nonlinear classification problems. In several ways, SVM can support multi-class classification including the one-versus-one [57], one-versus-all [58] and directed acyclic graph (DAG) [59]. A support vector machine library (LIBSVM) was utilized in [60] to support the multi-class nature of the domain. Practically, the support vector machine performs classification between two classes by drawing a border between the two classes in a plane. The drawn border separates the two classes from each other, as illustrated in **Fig. 7**. To this end, two parallel and two near border lines are drawn across the two classes and these boundaries are drawn closer to each other to produce a corresponding boundary demarcation.

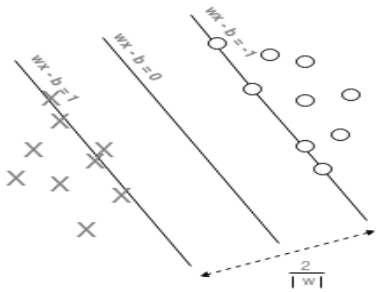


Fig. 7. SVM boundaries

As depicted in the figure above, two classes appeared on a two-dimensional plane. It is conceivable to perceive these dimensions and planes as properties. The SVM operation can be viewed from the feature extraction perspective. Here, feature extraction is carried out on each input entering the model in a simple sense, bringing about an alternate point demonstrating each input on this two-dimensional plane. Classification of these points is the classification of inputs with respect to the extracted properties. In this plane, if we let training vectors $x_i \in R^p$, $i=1,\ldots, n$, in the two classes, and a vector $y \in \{1,-1\}^n$, then the following primal task is solved by the support vector classifier as follows:

$$\begin{gathered}\min_{f,k,c} \frac{1}{2} f^T f + C \sum_{i=1}^{n} c_i \\ \text{Subject to } y_i(f^T \phi(x_i) + k) \geq 1 - c_i \\ c_i \geq 0, \ i=1,\ldots,n\end{gathered} \tag{4}$$

The dual of the support vector classifier becomes:

$$\begin{gathered}\min_{\alpha} \frac{1}{2} \alpha^T Q \alpha - e^T \alpha \\ \text{Subject to } y^T \alpha = 0 \\ 0 \leq \alpha_i \leq C, \ i=1,\ldots,n\end{gathered} \tag{5}$$

Here, a vector of all ones is represented as e, the upper bound is denoted as $C > 0$, Q is an nxn positive semi-definite matrix, $Q_{ij} \equiv y_i y_j K(x_i, x_j)$ *where* $K(x_i, x_j) = \phi(x_i)^T \phi(x_j)$ represents the kernel. During training, training vectors are explicitly spread into higher dimensional space via function ϕ. After this, the decision function is computed as:

$$sgn\left(\sum_{i=1}^{n} y_i \alpha_i K(x_i, x_j) + \rho\right) \tag{6}$$

The support vector classifier parameters are usually accessed using $y_i \alpha_i$, kernel and the independent term ρ.

3. Simulation Results and Discussion

3.1 Simulation of Proposed Model

The proposed model, given in **Fig. 1,** is designed using the American Sign Language (ASL) fingerspelling dataset, where each of the hand images in the dataset represents a sign in ASL. The constructed hybrid model will translate a real-time finger sign into one of the 24 signs in the ASL. The system is implemented using SSD plus Inception v3 plus SVM classifier by referencing the CNN learning technique. The input of the proposed framework is images from the camera mounted in front of the user. Here, the user's hand images are fed into the proposed hybrid model via the SSD module. The cropped hand image is fed into the input layer of the Inception v3 module, where hand gesture features are extracted. The user's extracted hand gesture features are used as an input to the SVM classifier. The incorporated SVM classifier uses the user's extracted hand gesture features to determine the state/position of the user's hand. The determined state or position corresponds to a sign language, which in this case is American Sign Language.

In this paper, the inception module of the proposed hybrid model is defined with four

parameters, namely width, height, depth and number of classes (signs). The width and height are the input size. The depth is the number of channels of input images. The input size is 299x299x3, where the width is 299, the height is 299 and the depth is 3 corresponding to the standard RGB. As previously mentioned, we subjected this high dimensional input space to factorization operations, which drastically reduced the high input space to a lower dimension. After this, the factorized low space is utilized as the input to the SVM classifier. The modules of the proposed hybrid model are defined in a sequential way since layers are added sequentially as demonstrated in **Fig. 1**.

Table 1 depicts a classification report and **Fig. 8** depicts a confusion matrix of the proposed hybrid model used for translation of hand gestures into American Sign Language. These results were obtained from an experiment conducted using a cross-validation method. As shown in both the table and the figure, miss-classification occurred once on the letters *m, s* and *u* out of the total of 24 signs/letters (classes) in the explored dataset. From these results, it is clear that the miss-classification rate is minimised and this shows the efficiency of the proposed hybrid model.

Table 1. Classification Report

Letters	Precision	Recall	F1-score	Support
a	1.00	1.00	1.00	137
b	1.00	1.00	1.00	152
k	1.00	1.00	1.00	145
l	1.00	1.00	1.00	137
m	0.99	0.99	0.99	167
n	1.00	0.99	1.00	146
o	1.00	1.00	1.00	146
p	1.00	1.00	1.00	165
q	1.00	1.00	1.00	153
r	1.00	0.99	1.00	147
s	0.99	1.00	1.00	147
t	1.00	1.00	1.00	146
c	1.00	1.00	1.00	141
u	0.99	1.00	1.00	151
v	1.00	1.00	1.00	158
w	1.00	1.00	1.00	155
x	1.00	1.00	1.00	140
y	1.00	1.00	1.00	155
d	1.00	1.00	1.00	174
e	1.00	1.00	1.00	143
f	1.00	1.00	1.00	137
g	1.00	1.00	1.00	152
h	1.00	1.00	1.00	177
i	1.00	1.00	1.00	155

In the paper, a LinearSVC estimator was utilized as the SVM class. LinearSVC has a simplified learning structure, is significantly less tuneable and is basically just a linear interoplation. It is integrated with the Inception v3 for classification purposes. The SVM classifier is comprised of parameters including loss='squared_hinge' representing the square of the hinge loss, penalty='l2' specifying the norm for penalization, C=5 representing the parameter C of the error term, and finally, multi_class='ovr' determines the multi-class strategy if *y* contains more than two classes. Here, "ovr" trains n_classes of one-vs-rest classifiers.

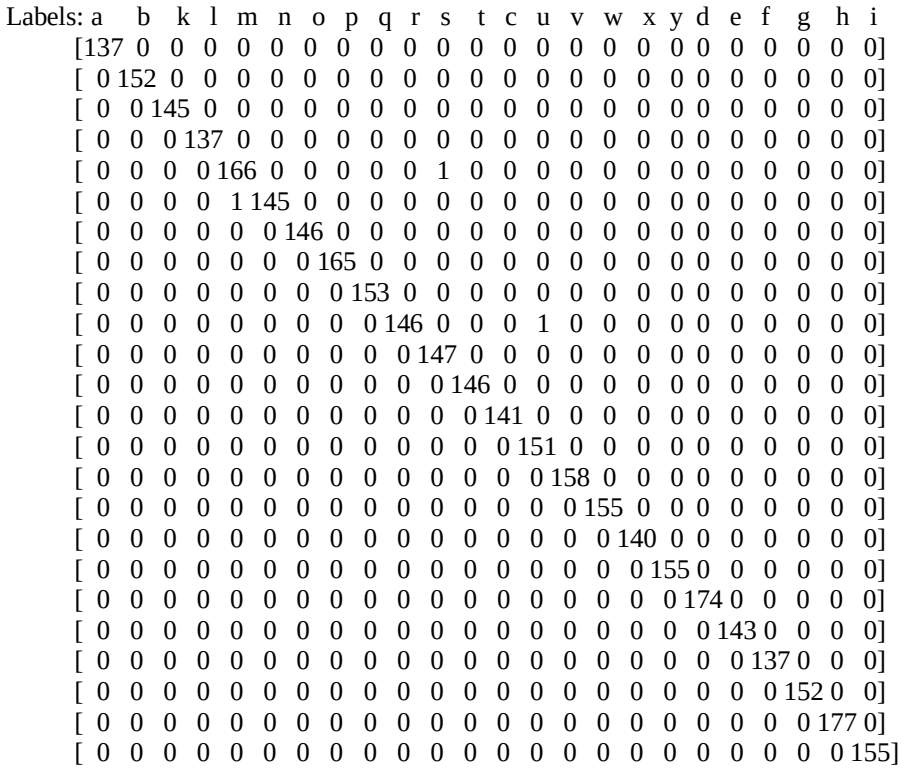


Fig. 8. Confusion matrix

In the first simulation, the design process is carried out using cross-validation on the proposed hybrid model. Ten-fold cross-validation is used in this experiment. Here the explored sample is divided into ten equal portions. Nine out of the ten are used for training, and the remaining portion is used for testing. The training process is repeated ten times by shifting train and test signals. The LinearSVC learning algorithm is also applied for implementation of training using 150 training epochs. The demonstrated accuracy value is the average of ten simulation results. The average value of the accuracy rate for the test achieved is 99.9% and the error is 0.0126.

In the second simulation, Monte Carlo-style estimation is performed on the explored ASL dataset where the hybrid model was run 500 times, each time splitting the sample randomly into 60% training and 40% test and calculating the error of the prediction on the test set. Monte Carlo estimators are an expansive class of algorithms which depend on the iteration of random sampling to produce numerical outcomes. Here, the main idea is utilizing randomness for solving deterministic tasks. They are regularly utilized in mathematical and physical problems and are most helpful when it is impossible or extremely difficult to utilize other methodologies. Monte Carlo estimators are mainly utilized in three class of problems: generating solutions using a probability distribution, numerical integration and optimization [61]. On a basic level, Monte Carlo estimators can be utilized for solving any task with a probabilistic interpretation. Integrals described by the expected value of some random variables can be approximated by taking the empirical mean of independent samples of the variable. As previously mentioned, we performed a Monte Carlo experiment on the ASL fingerspelling dataset where the proposed model was run 500 times, each time splitting the sample randomly into 60% training and 40% test where the error of the prediction on the test set obtained was 0.023 and the accuracy was 98.91%. **Table 2** contains the results of simulations of the proposed hybrid

network using both cross-validation and Monte Carlo approaches.

Table 2. Simulation Results of the proposed model

	Accuracy (%)	RMSE
Monte Carlo Method	98.91	0.023
Cross-Validation Method	99.90	0.0126

3.2 Other Tested Models

In order to present the best performing model for the ASL fingerspelling dataset, we applied other machine learning approaches such as CNN [15,62], a histogram of oriented gradient (HOG) plus neural networks (NN) and finally, HOG plus SVM. After a set of simulations, a comparative analysis is made in order to determine the best performing model.

3.2.1 CNN Simulation

In the first stage, the CNN structure with a fully connected network is applied for performing ASL translation. **Table 3** presents the structure of the CNN utilized for American Sign Language translation. The structure of the CNN includes two convolutional layers, max pooling and the fully connected layers.

Table 3. CNN Structure

Layer (type)	Output Shape	Parameters
conv2d_1 (Conv2D)	(None, 26, 26, 16)	160
Max_pooling2d_1	(None, 13, 13, 16)	0
conv2d_2 (Conv2D)	(None, 11, 11, 32)	4640
max_pooling2d_2	(None, 5, 5, 32)	0
conv2d_3 (Conv2D)	(None, 3, 3, 64)	18496
max_pooling2d_3	(None, 1, 1, 64)	0
flatten_1 (Flatten)	(None, 64)	0
dense_1 (Dense)	(None, 768)	49920
dense_2 (Dense)	(None, 128)	98432
dense_3 (Dense)	(None, 24)	3096

For training of the CNN, the data is divided into two parts: 80% and 20%. 80% of the domain is used for training while the remaining 20% of the data is used for testing. From the 80% of the data assigned for training, 60% is used for training while 40% is used for validation. We utilized equations (1-3) to determine the output signals of the CNN.

During the simulation, the size of available training data and system specifications used to construct the CNN model were taken into consideration. Thus, during simulation, Z-score normalization was applied to scale each input signal and this technique enhanced the generalization of the model. Note that an RMSprop learning algorithm is employed for training. In addition, we trained the CNN model using 150 epochs. CNN includes two convolutional layers. The input size is 4096 and the kernel size is 3. Accordingly, the fully connected network is applied for American Sign Language classification purpose. As earlier stated, CNN was trained using 150 epochs. At every iteration of each epoch, 60% of the training set was utilized for training and 40% for validation. **Fig. 9**. depicts the simulation results realized for the loss function and accuracy, while **Table 4** depicts the simulation results of the CNN. During training, the value of the loss function obtained is 1.5676e-08. The value of the loss function for validation data obtained is 0.0054, and that for test data is 0.0054. For

test data, the value of accuracy is 92.21% and the error is as low as 0.0234.

Fig. 9. CNN simulation results obtained for loss function and accuracy

Table 4. CNN Simulation Results

	Loss Function	AUC (%)	RMSE	Accuracy(%)
Training	105676e-08	100	2.2019e-05	100
Validation	0.0054	97.72	0.0234	92.22
Testing	0.0054	97.71	0.0234	92.21

3.2.2 Simulation using HOG plus NN

In the next simulation, the ASL fingerspelling dataset is used for the design of the translation system using the histogram of oriented gradient (HOG) plus neural networks (NN) structure. Here, each of the hand images in the dataset represents a sign in American Sign Language. As mentioned, this model is capable of translating real-time hand gestures into one of the 24 signs in the American Sign Language. The design of this model is in line with that of the proposed hybrid model. Here, we use the HOG module for data pre-processing and feature extraction, while the NN module classifies the extracted features into American Sign Language. **Table 5** depicts the structure of the HOG plus NN utilized for the classification.

Table 5. HOG plus NN Structure

Layer (type)	Output Shape	Parameters
dense_1 (Dense)	(None, 768)	787200
dense_2 (Dense)	(None, 128)	98432
dense_3 (Dense)	(None, 24)	3096
activation_1 (Activation)	(None, 24)	0

During training, the data are divided into two parts: 80% and 20%. 80% of the dataset is utilized for training while the remaining 20% of the dataset is utilized for testing. From the 80% of the dataset assigned for training, 60% is used for training while 40% for validation.

During simulation, Z-score normalization was used for signal scaling and Gaussian activation function was employed for training. Also, we trained the model using 150 epochs. **Fig.10.** illustrates the simulation outcomes obtained for the loss function and accuracy, while **Table 6** depicts the simulation results of the model. During training, the value of the loss function obtained is 0.0568. The value of the loss function for validation data obtained is 0.1541, and that for test data is 0.12037. For test data, the value of accuracy is 96.30%.

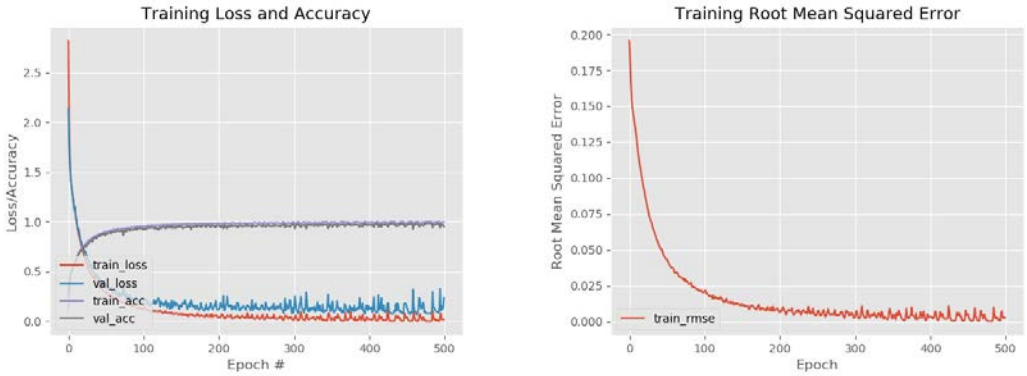


Fig. 10. Simulation results obtained for the loss function, accuracy and RMSE

Table 6. CNN Simulation Results

	Loss Function	Accuracy(%)	AUC (%)	RMSR (%)
Training	0.0568	97.92	99.99	0.0098
Validation	0.1541	95.04	99.77	0.0164
Testing	0.12037	96.30	99.63	0.0141

3.2.3 Simulation using HOG plus SVM

In this simulation, by combining the HOG module plus SVM module, a hybrid model is formed for the design of the translation system. But here, the HOG module is utilized for data pre-processing and feature extraction, while the SVM module is used to classify the extracted features into ASL. **Table 7** and **Fig. 11** depict a classification report and confusion matrix, respectively. As seen in the results, miss-classification is low leading to high accuracy. These results were obtained using a cross-validation approach as described in the implementation of the proposed hybrid model. The LinearSVC learning algorithm is also utilized in the design of the model. Training of the model is performed using 150 epochs. The demonstrated accuracy value is the average of ten simulation results. The average value of the accuracy rate for the test achieved is 99.28% and the error is 0.5981.

Table 7. Classification report for HOG plus SVM

Letters	Precision	Recall	F1-score	Support
a	1.00	1.00	1.00	168
b	0.99	1.00	1.00	166
k	0.99	1.00	1.00	163
l	1.00	1.00	1.00	155
m	0.99	0.97	0.98	172
n	0.97	0.96	0.97	147

o	1.00	1.00	1.00	134
p	1.00	1.00	1.00	133
q	1.00	1.00	1.00	157
r	1.00	0.99	1.00	130
s	0.99	1.00	1.00	161
t	0.99	1.00	1.00	147
c	1.00	1.00	1.00	148
u	0.99	0.98	0.98	155
v	0.99	0.95	0.97	147
w	0.96	0.99	0.98	154
x	1.00	0.99	1.00	135
y	1.00	1.00	1.00	136
d	0.97	1.00	0.99	146
e	0.99	1.00	1.00	182
f	1.00	1.00	1.00	135
g	1.00	1.00	1.00	139
h	1.00	1.00	1.00	177
i	0.98	1.00	0.99	139

```
Labels: a  b  k  l  m  n  o  p  q  r  s  t  c  u  v  w  x  y  d  e  f  g  h  i
[168  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
[  0 166  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
[  0  0 163  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
[  0  0  0 155  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
[  0  0  0  0 166  3  0  0  0  0  0  1  0  0  0  0  0  0  1  0  0  0  0  1]
[  0  0  0  0  0 142  0  0  0  0  1  0  0  1  0  0  0  0  0  1  0  0  0  2]
[  0  0  0  0  0  0 134  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
[  0  0  0  0  0  0  0 133  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
[  0  0  0  0  0  0  0  0 157  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
[  0  0  0  0  0  0  0  0  0 129  0  0  0  0  0  0  0  0  1  0  0  0  0  0]
[  0  0  0  0  0  0  0  0  0  0 161  0  0  0  0  0  0  0  0  0  0  0  0  0]
[  0  0  0  0  0  0  0  0  0  0  0 147  0  0  0  0  0  0  0  0  0  0  0  0]
[  0  0  0  0  0  0  0  0  0  0  0  0 148  0  0  0  0  0  0  0  0  0  0  0]
[  0  1  1  0  0  0  0  0  0  0  0  0  0 152  0  0  0  0  1  0  0  0  0  0]
[  0  0  0  0  1  1  0  0  0  0  0  0  0  0 141  4  0  0  0  0  0  0  0  0]
[  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1 153  0  0  0  0  0  0  0  0]
[  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 134  0  1  0  0  0  0  0]
[  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 136  0  0  0  0  0  0]
[  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 146  0  0  0  0  0]
[  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 182  0  0  0  0]
[  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 135  0  0  0]
[  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 139  0  0]
[  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 177  0]
[  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 139]
```

Fig. 11. Confusion matrix for HOG plus SVM

3.3 Comparative Results of Different Models

We have listed the results of performances of most competitive models used for sign language translation in **Table 8**. We have considered the papers that presented accuracy results. The simulations were performed using different databases selected by the authors. As shown some studies use American sign language dataset some of them Indian sign language dataset, some-Indonesian sign language dataset. The analyses of some of these studies are given in the

introduction section.

Table 8. The simulation results of the selected studies obtained for the language translation

Author (year)	Method	Database	Performances based on the accuracy (%)
Jalal et al. (2018) [1]	capsule-based deep neural network	'Kaggle' ASL Letter	99.74
Rastgoo et al. (2018) [62]	Restricted Boltzmann Machine	ASL Finger-spelling A	98.13
Rastgoo et al. (2018) [62]	Restricted Boltzmann Machine	Massey University Dataset	99.31
Lahamy and Lichti (2012) [63]	Real-Time and Rotation-Invariant	ASL alphabet	93.88
Vaitkevičius et al. (2019) [64]	Hidden Markov classification	ASL alphabet	86.10
Atwood et al. (2012) [65]	Neural network and principal component analysis	ASL alphabet	96.10
Bheda & Radpour (2017) [15]	deep CNN	NZ ASL	82.50
Dong et al. (2015) [66]	Microsoft Kinect	ASL alphabet	90.00
Kacper and Urszula (2018) [67]	Snapshot learning	ASL Finger-spelling	93.30

We have also depicted the performances of different models based on deep learning structures. Extensive comparison of object detection methods is provided in the papers [36,37]. From these papers, it is clear that SSD method has better performance than other methods. In this paper, using feature extraction and classification techniques we have designed different models for comparative purpose. These models are based on HOG-SVM, HOG-NN, CNN-fully connected network (FCM), inception V3-SVM. **Table 9** includes the performances of the models used for ASL translation. In the sign language translation model, we performed transfer learning where we reused pre-trained SSD and Inception V3 models, and concatenate them with SVM classifier to perform classification on our original data set. As demonstrated, the proposed hybrid model (SSD + Inception v3 + SVM) has better performance as compared to other models in **Table 9**. The incorporation of SSD for hand detection makes feature extraction easy and faster for the Inception v3 in the second module.

Table 9. Comparative results of different models

Author (year)	Method	Performances based on the accuracy (%)
Current research	SSD+HOG + SVM	99.28
Current research	SSD+HOG + NN	96.30
Current research	SSD+CNN+FCN	92.21
Propose hybrid model	SSD + Inception v3 + SVM	99.90

The proposed hybrid model has been tested in real-time and is capable of translating real-time hand gestures into one of the 24 signs in the American Sign Language. The fragments of experiments about real-time implementation are provided in the web pages[1]. The real-time experiments for sign language translation are repeated ten times. In each case, the

[1]https://www.youtube.com/watch?v=FRUvbRRfZMw and https://www.youtube.com/watch?v=TzwfcW3Ufts.

recognition system is run to recognize all the 24 signs presented on-line. The accuracy rate of the system was obtained as 96%. As shown in real-time, we got accuracy rate less than the accuracy rate obtained with ASL data set as depicted in **Table 9**. This low accuracy is due to inaccurate representation of some signs by the user's hand. This can be corrected by changing hand orientation and clearer representation of signs, after this correction, we got the same result (99.90%) as shown in **Table 9**. The obtained outcomes depict the good learning convergence and high performance of the proposed hybrid model. The presented model combines object detection, feature extraction and classification modules, which simplifies the structure of the ASL translation model. These comparative results certainly demonstrate the efficiency of the proposed hybrid system over other models targeted at solving the same problem.

4. Conclusion

In this paper, using the integration of SSD, Inception v3 and SVM, a vision-based American Sign Language Translator is implemented. Training of the hybrid system is implemented using the cross-validation technique and validated using a Monte Carlo estimator. The results obtained from these two experiments show the effectiveness of the cross-validation approach over the Monte Carlo method. The design of the proposed hybrid system is implemented using the ASL fingerspelling dataset. As a result of the simulation of the proposed hybrid model using a cross-validation approach, the average value of the accuracy rate obtained is 99.9%, and RMSE is 0.0126. One of the basic advantages of the proposed hybrid system is its simplified structure that seamlessly combines object detection, feature extraction and classification stages in the body of the deep learning structure without ambiguities. The designed system can automatically detect hands from camera images and classifies the detected object into one of the 24 signs in the ASL. The designed system can improve information communication between people, particularly between those who are deaf, hard of hearing and people who have some speaking difficulties.

References

[1] M.A. Jalal, R.Chen, R.Moore, et al., "American sign language posture understanding with deep neural networks," in *Proc. of 21st International Conference on Information Fusion (FUSION)*, pp. 573-579, 10-13 Jul 2018. Article (CrossRef Link)

[2] S.P. Becky, "Sign Language Recognition and Translation: A Multidisciplined Approach From the Field of Artificial Intelligence," *Journal of Deaf Studies and Deaf Education Advance Access*, Vol. 11, no.1, pp.94-101, 2006. Article (CrossRef Link)

[3] R.Bowden, A. Zisserman, T. Kadir, and M. Brady, "Vision-based interpretation of natural sign languages," in *Proc. of the 3rd International Conference on Computer Vision Systems*, pp.391-401, April 2003.

[4] American Sign Language, National Institute on Deafness and Other Communication Disorders. http://www.nidcd.nih.gov/health/hearing/asl.asp

[5] Rahib H. Abiyev, "Facial Feature Extraction Techniques for Face Recognition," *Journal of Computer Science*, Vol.10, no.12, pp.2360-2365, 2014. Article (CrossRef Link)

[6] C. Jennings, "Robust finger tracking with multiple cameras," in *Proc. of Int. Workshop on Recognition, Analysis, and tracking of Faces and Gestures in Real-Time Systems*, 1999. Article (CrossRef Link)

[7] S. Malassiotis, N. Aifanti, and M. G. Strintzis, "A Gesture Recognition System Using 3D Data," in *Proc. of IEEE 1st International Symposium on 3D Data Processing Visualization and Transmission*, June 2002. Article (CrossRef Link)

[8] B. S.Parton, "Sign Language Recognition and Translation: A Multidisciplined Approach From the Field of Artificial Intelligence," *Journal of Deaf Studies and Deaf Education*, Vol.11, no.1, pp.94-101, 2006. Article (CrossRef Link)

[9] G. Fang, W. Gao, X. Chen, C. Wang, and J. Ma, "Signer independent continuous sign language recognition based on SRN/HMM," in *Proc. of International Gesture Workshop, Gesture and Sign Language in Human-Computer Interaction*, pp. 76-85, 2001. Article (CrossRef Link)

[10] S. A. K. Mehdi Y. N., "Sign language recognition using sensor gloves," in *Proc. of the 9th International Conference on Neural Information Processing, 2002. ICONIP '02.*, vol. 5, pp. 2204–2206, 2002. Article (CrossRef Link)

[11] S. Sidney Fels and G. E. Hinton, "Glove-Talk: A Neural Network Interface Between a Data-Glove and a Speech Synthesizer," *IEEE Transactions on Neural Networks*, vol. 4, no. 1, pp. 2–8, 1993. Article (CrossRef Link)

[12] A. K. Singh, B. P. John, S. R. Venkata Subramanian, A. Sathish Kumar, and B. B. Nair, "A low-cost wearable Indian sign language interpretation system," in *Proc. of International Conference on Robotics and Automation for Humanitarian Applications, RAHA 2016 - Conference Proceedings*, 2017. Article (CrossRef Link)

[13] T. E. Starner and A. Pentland, "Visual Recognition of American Sign Language Using Hidden Markov Models," *Media*, pp. 189–194, 1995.

[14] Pradeep Kumar, Himaanshu Gauba, Partha Pratim Roy, Debi Prosad Dogra, "A multimodal framework for sensor based sign language recognition," *Neurocomputing*, Vol.259, pp. 21-38, 11 October 2017. Article (CrossRef Link)

[15] V. Bheda and D. Radpour, "Using Deep Convolutional Networks for Gesture Recognition in American Sign Language," *CoRR, vol. abs/1710.06836*, 2017. Article (CrossRef Link)

[16] S. Ameen and S. Vadera, "A convolutional neural network to classify American Sign Language fingerspelling from depth and colour images," *Expert Systems*, vol. 34, no. 3, 2017. Article (CrossRef Link)

[17] J. Singha and K. Das, "Indian Sign Language Recognition Using Eigen Value Weighted Euclidean Distance-Based Classification Technique," *Inter. Journal of Advanced Computer Science and Applications*, vol. 4, no. 2, pp. 188–195, 2013. Article (CrossRef Link)

[18] Nada B. Ibrahim, Mazen M. Selim, Hala H. Zayed, "An Automatic Arabic Sign Language Recognition System (ArSLRS)," *Journal of King Saud University – Computer and Information Sciences*, Vol. 30, no. 3, pp. 470–477, 2018. Article (CrossRef Link)

[19] Wenwen Yang, Jinxu Tao, Zhongfu Ye, "Continuous sign language recognition using level building based on fast hidden Markov model," *Pattern Recognition Letters*, Vol.78, pp.28–35, 2016. Article (CrossRef Link)

[20] Pradeep Kumar, Partha Pratim Roya, Debi Prosad Dogra, "Independent Bayesian classifier combination based sign language recognition using facial expression," *Information Sciences*, vol.428, pp.30–48, 2018. Article (CrossRef Link)

[21] S.Tan Dat Nguyen, Surendra Ranganath, "Facial expressions in American sign language: Tracking and recognition," *Pattern Recognition*, Vol.45, no.5, pp.1877–1891, 2012. Article (CrossRef Link)

[22] Djamila Dahmani, Slimane Larabi, "User-independent system for sign language finger spelling recognition," *J. Vis. Commun. Image R.*, Vol.25, no.5, pp.1240–1250, 2014. Article (CrossRef Link)

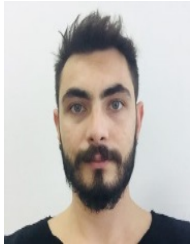
[23] C. Amrutha C U,Nithya Davis, Samrutha K S, Shilpa N S, Job Chunkath, "Improving language acquisition in sensory deficit individuals with Mobile Application," in *Proc. of International Conference on Emerging Trends in Engineering, Science and Technology (ICETEST, 2015)*, 2015. Article (CrossRef Link)

[24] Md. Zia Uddin and Jaehyoun Kim, "A Robust Approach for Human Activity Recognition Using 3-D Body Joint Motion Features with Deep Belief Network," *KSII Transactıons on Internet and Informatıon Systems*, vol. 11, no. 2, pp.1118-1133, Feb. 2017. Article (CrossRef Link)
[25] Jiaquan Shen, Ningzhong Liu, Han Sun, Xiaoli Tao, Qiangyi Li, "Vehicle Detection in Aerial Images Based on Hyper Feature Map in Deep Convolutional Network," *Multimedia Tools and Applications*, vol.76, no.20, pp.21651-21663, 2017. Article (CrossRef Link)
[26] Nguyen Manh Dung , Dongkeun Kim and Soonghwan Ro, "A Video Smoke Detection Algorithm Based on Cascade Classification and Deep Learning," *KSII Transactıons on Internet and Informatıon Systems*, vol. 12, no. 12, pp.6018-6033, Dec. 2018. Article (CrossRef Link)
[27] Sheraz Naseer and Yasir Saleem, "Enhanced Network Intrusion Detection using Deep Convolutional Neural Networks," *KSII Transactıons on Internet and Informatıon Systems*, Vol. 12, No. 10, pp.5159-5178, Oct. 2018. Article (CrossRef Link)
[28] Li Zhang, Jingdun Jia, Yue Li, Wanlin Gao and Minjuan Wang, "Deep Learning based Rapid Diagnosis System for Identifying Tomato Nutrition Disorders," *KSII Transactıons on Internet and Informatıon Systems*, Vol. 13, no. 4, pp.2012-2027, 2019. Article (CrossRef Link)
[29] Viola P.A., Jones M.J., "Rapid object detection using a boosted cascade of simple features," in *Proc. of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Vol.1, pp.511–518, 2001. Article (CrossRef Link)
[30] Dalal N, Triggs B., "Histograms of Oriented Gradients for Human Detection," in *Proc. of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Vol. 1, pp.886-893, 2005. Article (CrossRef Link)
[31] Gu C., Lim J. J., Arbelaez P., and Malik J., "Recognition using ´ regions," in *Proc. of CVPR*, 2009.
[32] Girshick R., Donahue J., Darrell T., and Malik J., "Region-based Convolutional Networks for Accurate Object Detection and Segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.38, Issue 1, pp. 142-158, 2016. Article (CrossRef Link)
[33] Ren, S., He, K., Girshick, R., Sun, J., "Faster R-CNN: Towards real-time object detection with region proposal networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 39, no. 6, pp.1137-1149, 2017. Article (CrossRef Link)
[34] Redmon, J., Divvala, S., Girshick, R., Farhadi, A., "You only look once: Unified, real-time object detection," in *Proc. of CVPR*, 2016. Article (CrossRef Link)
[35] Redmon J., "Yolov3: An incremental improvement," *arXiv:1804.02767 [cs.CV]*, 2018. Article (CrossRef Link)
[36] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, Y., Berg, A., "SSD: Single shot multibox detector," in *Proc. of European Conference on Computer Vision*, pp.21–37, 2016. Article (CrossRef Link)
[37] Zhao Z-Q, Zheng P., Xu S-T, Wu X, "Object Detection with Deep Learning: A Review," *IEEE Transactions on Neural Networks and Learning Systems*, vol.30, no.11, 2019. Article (CrossRef Link)
[38] Uijlings J., van de Sande K., Gevers T., and Smeulders A., "Selective search for object recognition," *Inter. Journal of Computer Vision*, Vol.104, no.2, pp.154-171, 2013. Article (CrossRef Link)
[39] Di Ruberto C. , Putzu L., "A Feature Learning Framework for Histology Images Classification," *Emerging Trends in Applications and Infrastructures for Computational Biology, Bioinformatics, and Systems Biology*, pp.37-47, 2016. Article (CrossRef Link)
[40] Wang Z., Healy G., Smeaton A.F., Ward T.E., "A review of feature extraction and classification algorithms for image RSVP-based BCI," *Signal Processing and Machine Learning for Brain-Machine Interfaces*, 2018. Article (CrossRef Link)
[41] *Cortes C.*,Vapnik V.N., "Support-vector networks," *Machine Learning*, Vol.20, no.3, pp.273–297, 1995.
[42] Kundu S., Ari S., "P300 based character recognition using convolutional neural network and support vector machine," *Biomedical Signal Processing and Control*, Vol.55, 101645, 2020.
[43] Kim S, Yu Z., Kil R.M, Lee M., "Deep learning of support vector machines with class probability output networks," *Neural Networks*, Vol. 64, pp.19–28, 2015. Article (CrossRef Link)

[44] Zareapoor M., Shamsolmoali P., Kumar Jain D., Wang H., Yang J., "Kernelized support vector machine with deep learning: An efficient approach for extreme multiclass dataset," *Pattern Recognition Letters*, Vol.115, pp.4–13, 2018. Article (CrossRef Link)
[45] Chen G., Xu R., Yang Z., "Deep ranking structural support vector machine for image tagging," *Pattern Recognition Letters*, Vol. 105, pp. 30–38, 2018. Article (CrossRef Link)
[46] Qi Z., Wang B, Tian Y., Zhang P., "When. Ensemble Learning Meets Deep Learning: a New Deep Support Vector Machine for Classification," *Knowledge-Based Systems*, Vol.107, pp. 54–60, 2016. Article (CrossRef Link)
[47] Li Y., Zhang T., "Deep neural mapping support vector machines," *Neural Networks*, Vol. 93, pp.185–194, 2017. Article (CrossRef Link)
[48] Fernandes A.M., Utkin A.B., Eiras-Dias J., Cunh J., Silvestre J., Melo-Pinto P., "Grapevine variety identification using "Big Data" collected with miniaturized spectrometer combined with support vector machines and convolutional neural networks," *Computers and Electronics in Agriculture*, Vol.163, 104855, 2019. Article (CrossRef Link)
[49] Rahib Abiyev, Mohammad Khaleel Sallam Ma'aitah, "Deep Convolutional Neural Networks for Chest Diseases Detection," *Journal of Healthcare Engineering*, Vol. 2018, 2018. Article (CrossRef Link)
[50] Rahib Abiyev, Murat Arslan, "Head mouse control system for people with disabilities," *Expert Systems*, 2019. Article (CrossRef Link)
[51] Diederik, P.K., and Jimmy, L.B., "ADAM: a method for stochastıc optımızatıon," *ICLR 2015*, 2015. Article (CrossRef Link)
[52] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, pp.1–9, 2014. Article (CrossRef Link)
[53] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. of The 32nd International Conference on Machine Learning*, pp. 448–456, 2015.
[54] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z., "Rethinking the inception architecture for computer vision," in *Proc. of the IEEE conference on computer vision and pattern recognition*, pp. 2818-2826, 2016. Article (CrossRef Link)
[55] Abiyev, R., Arslan, M., Gunsel, I., Cagman, A., "Robot Pathfinding Using Vision Based Obstacle Detection," in *Proc. of 3rd IEEE International Conference on Cybernetics (CYBCONF), Book Series: IEEE International Conference on Cybernetics-CYBCONF*, pp.29-34, Jun 21-23, 2017. Article (CrossRef Link)
[56] Vapnik V.N., *The nature of statistical learning theory*, Springer, Berlin, 1995.
[57] Hsu CW, Lin CJ., "A comparison of methods for multiclass support vector machines," *IEEE Trans Neural Networks*, vol.13, no.2, pp.415–425, 2002. Article (CrossRef Link)
[58] Rifkin R, Klautau A., "In defence of one-versus-all classification," *J Mach Learn Research*, Vol. 5, pp.101–141, 2004.
[59] Platt J, Cristanini N, Shawe-Taylor J., "Large margin DAGs for multiclass classification," *Adv. Neural Inform Process Syst*, 12, 543–557, 2000.
[60] Chang C-C, Lin C-J, "LIBSVM: a library for support vector machines," *ACM Trans Intell Syst Technol*, Vol.2, pp.27.1–27.27, 2011.
[61] Kroese, D.P., Brereton, T., Taimre, T., Botev, Z.I., "Why the Monte Carlo method is so important today," *WIREs Comput Stat*, Vol.6, no.6, pp. 386–392, 2014. Article (CrossRef Link)
[62] Rastgoo, R., Kiani, K., & Escalera, S., "Multi-Modal Deep Hand Sign Language Recognition in Still Images Using Restricted Boltzmann Machine," *Entropy*, Vol.20, no.11, 809, 2018. Article (CrossRef Link)
[63] Lahamy, H., & Lichti, D., "Towards Real-Time and Rotation-Invariant American Sign Language Alphabet Recognition Using a Range Camera," *Sensors*, Vol.12, no.11, pp.14416-14441, 2012. Article (CrossRef Link)

[64] Vaitkevičius, A., Taroza, M., Blažauskas, T., Damaševičius, R., Maskeliūnas, R., & Woźniak, M., "Recognition of American Sign Language Gestures in a Virtual Reality Using Leap Motion," *Applied Sciences*, Vol.9, no.3, 445, 2019. Article (CrossRef Link)
[65] J. Atwood, M. Eicholtz, and J. Farrell, "American Sign Language Recognition System," *Artificial Intelligence and Machine Learning for Engineering Design. Dept. of Mechanical Engineering, Carnegie Mellon University*, 2012.
[66] Cao Dong, Ming C. Leu and Zhaozheng Yin, "American Sign Language Alphabet Recognition Using Microsoft Kinect," in *Proc. of IEEE International Conference on Computer Vision and Pattern Recognition Workshops*, 2015. Article (CrossRef Link)
[67] Kacper, K., and Urszula M., "American Sign Language Fingerspelling Recognition Using Wide Residual Networks," *Artificial Intelligence and Soft Computing*, pp. 97-107, 2018. Article (CrossRef Link)
[68] Murat Arslan, Rahib Abiyev, Idoko John Bush, "Head Movement Mouse Control Using Convolutional Neural Network for People with Disabilities," in *Proc. of 13th Int. Conf. on Application of Fuzzy Systems and Soft Computing, ICAFS 2018, Advances in Intelligent Systems and Computing*, vol.896, pp.239-248, 2018. Article (CrossRef Link)

Rahib H. Abiyev received the B.Sc. and M.Sc. degrees (First Class Hons.) in electrical and electronic engineering from Azerbaijan State Oil Academy, Baku, in 1989, and the Ph.D. degree in electrical and electronic engineering from Computer-Aided Control System Department, Azerbaijan State Oil Academy, in 1997. He was a Senior Researcher with the research laboratory "Industrial intelligent control systems" of Computer-aided Control System Department. In 1999, he joined the Department of Computer Engineering, Near East University, Nicosia, North Cyprus, where he is currently a Full Professor and the Chair of Computer Engineering Department. In 2001, he founded Applied Artificial Intelligence Research Centre and in 2008, he created "Robotics" research group. He is currently the Director of the Research Centre. He has published over 200 papers in related subjects. His current research interests include softcomputing, control systems, robotics, and signal processing.



Murat Arslan was born on in Antalya on 7th of July, 1988. After finishing secondary school he started to study Computer Technologies and Programming, Suleyman Demirel University. In 2009, he joined Near East University to continue his BSc study. He received BSc and MSc degrees in Computer Engineering in 2013 and 2017 correspondingly. In 2017, he started doctorate study. During his study he started work as a research assistant in the Applied Artificial Intelligence Research Centre. He worked on NEUIslander soccer robots team. Currently, he is doing research on deep learning algorithms and computer vision. His research interests are machine learning, computer vision and robotics

John Bush Idoko was born in Ekeh, Nigeria in 1989. He attended Emmanuel Secondary School Ugbokolo after which he proceeded to the Benue State University Makurdi, Nigeria for his BSc degree in Computer Science and completed in 2010. After receiving BSc degree, he started MSc program in Computer Engineering at Near East University, North Cyprus. After receiving MSc degree, he started PhD degree in the same department in 2017. During his study he started work as a research assistant in the Applied Artificial Intelligence Research Centre. Currently, he is doing research on deep learning algorithms and computer vision. His research interests are machine learning, computer vision and signal processing