Deep Learning-based Fast Hand Gesture Recognition using Representative Frames

Vijay John and Ali Boyali and Seiichi Mita
Toyota Technological Institute, Nagoya, Japan
E-mail: $\{vijayjohn; aliboyali; smita\}$@toyota-ti.ac.jp.

Masayuki Imanishi and Norio Sanma
Nippon Soken, Denso, Japan
E-mail: $\{masayuki\_i\_imanishi; norio\_sanma\}$@soken1.denso.co.jp

***Abstract*—In this paper, we propose a vision-based hand gesture recognition system for intelligent vehicles. Vision-based gesture recognition systems are employed in automotive user interfaces to increase the driver comfort without compromising their safety. In our algorithm, the long-term recurrent convolution network is used to classify the video sequences of hand gestures. In the standard long-term recurrent convolution network-based action classifier, multiple frames sampled from the video sequence are given as an input to the network, to perform classification. However, the use of multiple frames increases the computational complexity, apart from reducing the classification accuracy of the classifier. We propose to address these issues by extracting a fewer representative frames from the video sequence, and inputting them to the long-term recurrent convolution network. To extract the representative frames, we propose to use novel tiled image patterns and tiled binary pattern within a semantic segmentation-based deep learning framework, the deconvolutional neural network. The novel tiled image patterns contain multiple non-overlapping blocks and represent the entire gesture-based video sequence within a single tiled image. These image patterns represent the input to the deconvolution network and are generated from the video sequence. The novel tiled binary pattern also contain multiple non-overlapping blocks and represent the representative frames of the video sequence. These binary patterns represent the output of the deconvolution network. The training binary patterns are generated from the training video sequences using the dictionary learning and sparse modeling framework. We validate our proposed algorithm on the public Cambridge gesture recognition dataset. A comparative analysis is performed with baseline algorithms and an improved classification accuracy is observed. We also perform a detailed parametric analysis of the proposed algorithm. We report a gesture classification accuracy of 91% and report a near real-time computational complexity of 110 ms per video sequence.**

I. Introduction

In recent years, hand gesture recognition has received significant attention from the research community, with the increased interest in advanced driver assistance systems (ADASs). In ADAS applications, the vision-based hand gesture recognition is used in the design and development of automotive user interfaces. Such user interfaces enable the driver to interact with the vehicle, without affecting their concentration. This method of interaction also increases the comfort of the driver, without compromising the driver's safety. However, the video-based gesture recognition problem is not trivial and is challenging due to: the intra and inter-persons variations in human hand gesture motion; inter-person variations in the shape and size of the human hand; illumination variations; and background noise.

In this paper, we propose to address these issues by utilizing the deep learning framework. Recently, Donahue et al. [1] proposed the long-term recurrent convolution network (LRCN) for human action recognition and reported state-of-the-art classification accuracy. Multiple frames sampled from the video sequence containing the human action are given as input to the LRCN, which identifies the associated the class label. In this paper, we adopt the LRCN model for the video-based hand gesture recognition. We propose to improve the LRCN's computational efficiency and classification accuracy by extracting fewer representative frames from the video sequence, and inputting them to the LRCN framework. The trained LRCN model estimates the class label associated with the hand gesture.

The deep learning framework is employed to extract the representative frames from the video sequence and classify the gesture. We propose to extract the representative frames from the video sequence using the deconvolution neural network (Deconvnet), a semantic segmentation framework. To extract the representative frames, we utilize novel tiled image patterns and novel tiled binary patterns. The tiled image and binary patterns are represented in the form of a single image with multiple non-overlapping blocks of smaller images. The tiled image pattern, which functions as the input to the Deconvnet, encapsulates the video-based hand gesture motion within a single image. On the other hand, the tiled binary pattern functions as the output of the Deconvnet, and contains the information of the representative frames. The tiled image patterns are generated from the video sequence. While, the tiled binary pattern for the Deconvnet training are generated from the training video sequence using the sparse modelling representative frame extraction algorithm (SMRF) proposed by Elhamifar et al. [2]. The SMRF algorithm identifies the representative frames directly from the video sequence. Given the extracted representative frames, the LRCN is then used to classify the gesture video sequence. The LRCN is trained with the extracted representative frames and their corresponding gesture labels. Owing to computational complexity, the SMRF is used in the training phase alone, to generate the tiled binary patterns for the Deconvnet and representative frames for the LRCN.

During testing, we first generate the tiled image pattern for the test sequence. The test tiled image pattern is then given as an input to the trained Deconvnet, which generates the tiled binary pattern as the output. The representative frames are extracted from the predicted tiled binary pattern. These representative frames are given as input to the trained LRCN, which predicts the output class label. Our proposed algorithm is validated on the publicly available Cambridge gesture recognition dataset and compared with baseline algorithms. Additionally, we report a detailed parameter evaluation of the proposed algorithm. We report a classification accuracy of 95% and computational efficiency of 110ms per sequence. Our main contribution to the literature are as follows: a novel representative frame-based deep learning framework is proposed for hand gesture recognition; the semantic segmentation-based deep learning framework is proposed for the identification of the representative frames in the video sequence; the application of LRCN for hand gesture recognition.

The rest of the paper is structured as follows. In Section I-A, we review the literature in gesture recognition. The proposed algorithm is presented in Section I-B, while the results are presented in Section II. We conclude the paper and present the direction of our future research in Section III.

A. Related Work

Hand gesture recognition is an important research area and has received significant attention in literature as described in the following survey [3]. In the standard vision-based hand gesture recognition framework, spatio-temporal features are first extracted from the video frames [4]. The inter-frame dynamics of these extracted features are then modeled using action classifiers such as the support vector machines (SVM) [5], neural network [7] or compressed sensing [8]. Dardas et al. [5] extract SIFT-based bag-of-word features from the input image. These discriminative features are then used to train a multiclass SVM-based gesture classifier. A similar work is reported by Ahmed et al. [7] , where the image moments-based feature are used to train an artificial neural network-based classifier. The standard vision-based hand gesture recognition algorithms are limited by varying illumination and inter and intra-person appearance and motion variations. To address these issues, and enhance the classification accuracy, researchers have utilized multi-modal sensors to perform the gesture recognition [9]–[11]. For example, in the work by Neverova et al. [10] and Ohn-Bar et al. [11], color and depth sensors (RGB-D) are utilized perform the gesture recognition. In [11], the authors extract HOG features from the RGB-D sensors, which are then used to train a SVM classifer. Apart from RGB-D sensors, researchers have also utilized depth, colour and radar sensors to enhance the classification accuracy [9].

In recent years, deep learning-based classification frameworks have reported state-of-the-art results in various image-based classification tasks [12]. Consequently, researchers have sought to adapt the image-based deep learning framework to perform activity recognition [13]–[16]. Two categories of literature exist for the deep learning-based activity recognition algorithms. In the first category, the deep learning framework is trained on the input data without explicitly modeling the inter-frame temporal information. Typically, multiple CNNs are adopted and trained with multiple scale data acquired from multiple sensors [13]–[15]. For example, Karpathy et al. [14] propose a multiresolution CNN architecture that are trained with original and cropped video frames to perform action recognition. A similar multiple network-based CNN is reported by Molchanov et al. [13], where the color and depth channels are combined and used to train two resolution-based CNN subnetworks. In the work by Neverova et al. [10], a multi-scale and multimodal deep learning framework is utilized. Multi-scale gesture information from the motion capture system along with the video are provided as inputs to multiple CNNs. The output of the different CNNs are fused by a multilayered perceptron in the final layer, and the activity is classified. A multiscale CNN is also adopted by Pigou et al. [15] to classify the American sign language. In their work, multiple frames of RGB and depth data derived from videos containing hand gestures and full body gestures are given as an input to two CNN networks to perform the classification. Finally, in the work by Simonyan et al. [16], two CNN architectures are trained on the video and the optical flow, derived from the video, respectively.

In the second category of deep learning-based activity recognition, researchers model the inter-frame temporal information using the recurrent neural network (RNN) [17], [18]. Murukami et al. [17] utilize the RNN to classify Japanese sign languages, while Ordonez et al. [18] utilize the long-term short memory-based RNN (LSTM) to classify gestures from accelerometer data. Recently, Donahue et al. [1] integrate the CNN within the LSTM to formulate the long recurrent convolutional neural network (LRCN). In the LRCN, multiple frames are sampled from a longer video sequence and given as inputs to multiple CNN, with each CNN obtaining an input from an individual frame. The outputs of the multiple CNN are then given as an input to the LSTM to predict the class label. The LSTM-based action classifiers report state-of-the-art classification accuracy on different activity recognition problems.

In our proposed work, the classification accuracy of the LRCN algorithm is enhanced by extracting representative frames from the video sequence. Additionally, a novel semantic segmentation-based deep learning framework, the deconvolution neural network, is utilized for the representative frame extraction.

B. Algorithm

Given a video sequence containing hand gesture motion, $\mathbf{V}_\alpha = \{I_t\}_{t=1}^{T}$ with T frames, the hand gesture recognition algorithm assigns a gesture label g from a set of pre-defined gesture labels to the video. In this paper, the LRCN deep learning framework [1] is adopted to estimate the hand gesture label. In the original LRCN algorithm, the authors sample $N < T$ frames from the original video sequence, and obtain the sampled video sequence, $\mathbf{V}_\beta = \{I_n\}_{n=1}^{N}$. The sampled sequence is given as the input to the LRCN. In our proposed algorithm, we improve the accuracy and classification accuracy of the LRCN framework, by extracting $K < N$ *representative* frames from the gesture-based video sequence. These *representative* frames are then given as an input to the LRCN to estimate the hand gesture label. The extracted representative frames generate the representative video sequence, $\mathbf{V}_\lambda = \{I_k\}_{k=1}^{K}$.

To extract the representative frames from the video sequence, we train the semantic segmentation-based deconvolution neural network (Deconvnet) [19] using tiled image patterns and tiled binary patterns. The tiled image and binary patterns are formulated as a single image with multiple non-overlapping blocks of smaller images. The tiled image patterns represent the input to the Deconvnet and are generated from the sampled sequence $\mathbf{V}_\beta$. On the other hand, tiled binary patterns represent the output to the Deconvnet and are generated using the representative frames extracted from sampled sequence $\mathbf{V}_\beta$. The representative frames are extracted from the video sequence using the sparse modeling algorithm (SRMF) proposed by Elhamifar et al. [2]. Given the training of the Deconvnet, the SMRF-based representative frames along with their corresponding gesture labels are used to train the LRCN . The trained LRCN predicts the gesture labels for a given video sequence. We next provide a brief overview of the Deconvnet, the LRCN and the sparse modeling algorithm, before presenting a detailed overview of our algorithm.

1) Deconvolutional Network: The deconvolution neural network is proposed by Noh et al. [19] as a semantic segmentation-based deep learning framework. The Deconvnet is an extension of the convolutional neural network (CNN) [12] used for image classification. The Deconvnet demonstrates state-of-the-art results on the semantic segmentation problem, where regions-of-interest within the image are identified using pixel-level labels. Consequently in our proposed algorithm, we adopt the pixel-labelling approach to identify the representative frames, considered as the regions-of-interest, within our novel tiled image pattern.

The Deconvnet contains the fully convolutional layers, the unpooling layers and the deconvolutional layers, in addition to the CNN's convolutional and pooling layers. The convolutional layers contain learnable filters which perform convolution operation with the previous layers, generating the feature maps. These feature maps are then sub-sampled by the pooling layers, by calculating the maximum over a feature map block. In the Deconvnet, the pooling layers store the location of the maximum activations within each block using switch variables. These switch variables are used by unpooling layers to perform the reverse pooling operation, which generates the sparse feature maps. The sparse unpooling feature maps are densified by the deconvolutional layers. The deconvolutional layers perform convolution-like operation with learnable filters. Finally, the fully convolutional layers contain reformulated fully connected layer weights, that perform the convolution over the entire input in the previous layer. In Deconvnet, the lower convolutional and deconvolution layers learn the low-level representation, while the higher layers learn the high-level feature representation.

2) Sparse Modeling: Elhamifar et al. [2] propose an algorithm to find the representative frames in a given dataset or video sequence. The representative frames are described as the subset of frames that efficiently represent and describe the entire dataset. The authors formulate the problem of finding the representative frames, within a sparse coding and dictionary learning framework. The problem is formulated under the assumption that each frame in the video sequence can be represented as a linear combination of the representative frames. The standard dictionary learning and sparse coding problem is given as,

$$\sum_{i=1}^{T} \|y_i - Dc_i\|_2^2 = \|Y - DC\|_F^2 \tag{1}$$

, where $Y = \{y_i\}_{i=1}^{T} \in \mathbb{R}^{m\times t}$ is data matrix with column-wise data points $y_i \in \mathbb{R}^m$, $D = \{d_i\}_{i=1}^{L} \in \mathbb{R}^{m\times l}$ is the dictionary and $C = \{c = i\}_{i=1}^{T} \in \mathbb{R}^{l\times t}$ represents the sparse coefficients. By solving the objective function in Eqn 1, the best representation of the data points in terms of the D and C is obtained, subject to constraints. In the work by Elhamifar et al. [2], the standard dictionary learning framework is modified to learn the representative frames. This is given by,

$$\sum_{i=1}^{T} \|y_i - Yc_i\|_2^2 = \|Y - YC\|_F^2 \tag{2}$$

$$\min \|Y - YC\|_F^2 \quad s.t. \|C\|_{1,q} \leq \tau, \quad \mathbf{1}^\top C = \mathbf{1}^\top, \tag{3}$$

, where $\|C\|_{1,q}$ is the sum of the q norms of the rows of C, and τ is an empirical parameter. $\mathbf{1}^\top C = \mathbf{1}^\top$ is an affine constraint. In the modified formulation, the dictionary is set to be the matrix of data points Y or the video sequence. Subsequently, only the sparse coefficients C, instead of both the D and C, is minimized, subject to constraints.

The solution of the objective function in Eqn 3, generates the representative frames as the non-zero rows of C or the columns in Y. Consequently, in our algorithm, the K representative frames for a video sequence Y are selected from the non-zero rows of C or the columns in Y. In this algorithm, owing to the computational complexity of the sparse learning framework, the sparse modeling algorithm is restricted to extracting the representative frames for the training phase. While the trained Deconvnet is used to extract the representative frames for the testing phase. In our experiment, the sparse modeling framework takes 400ms to extract the representative frames from a 16 frame sampled video sequence.

3) Long Convolutional Recurrent Neural Network: The long convolutional recurrent neural network (LRCN) algorithm proposed by Donahue et al. [1] is based on the LSTM and the CNN. The LSTM is an extension of the recurrent neural network (RNN), which models the temporal dynamics. The RNN models the temporal dynamics sequentially using the following equations recurrently,

$$h_t = g(W_{xh}x_t + W_{hh}h_{t-1} + b_h) \tag{4}$$

$$z_t = g(W_{hz}h_t + b_z) \tag{5}$$

where g is an elementwise activation function, x_t is the input , $h_t \in \mathbb{R}^N$ is the hidden state with N hidden units and z_t is the output. W_{xh}, W_{hh} and W_{hz} are the RNN weights and b_h and b_z are the RNN biases.

Despite obtaining good results on various problems, the RNN is limited by the vanishing and exploding gradient problem [1]. This is addressed by the LSTM, which incorporates memory units in the form of gates within the RNN framework. The LSTM gates are used to selectively forget, update and transmit the hidden states between layers. Utilizing the memory gates, the LSTM update equations at a given time step t are given by,

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i) \tag{6}$$
$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + b_f) \tag{7}$$
$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o) \tag{8}$$
$$\widetilde{C}_t = \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \tag{9}$$
$$C_t = f_t \odot C_{t-1} + i_t \odot \widetilde{C}_t \tag{10}$$
$$h_t = o_t \odot \tanh(C_t) \tag{11}$$

, where $i_t \in \mathbb{R}^N$ is the input gate, $f_t \in \mathbb{R}^N$ is the forget gate, $\widetilde{C}_t \in \mathbb{R}^N$ is the input modulation gate, $C_t \in \mathbb{R}^N$ is the memory cell unit and $o_t \in \mathbb{R}^N$ is the output gate. $\odot$ denotes element-wise product operation. σ represents the sigmoid activation function, while tanh represents the hyperbolic tangent activation function. W_{xi}, W_{hi}, W_{xf}, W_{hf}, W_{xo} and W_{ho} are the LSTM weights. b_i, b_f, b_o and b_c are the LSTM biases.

In the LSTM, the input gate i_t and forget gate f_t employs the sigmoid function on the current input x_t and previous hidden state h_{t-1} to train the LSTM to selectively forget its previous memory. On the other hand, the input modulation gate $\widetilde{C}_t$ employs the tanh function on the current input x_t and previous hidden state h_{t-1}. The memory cell unit C_t are a summation of two terms, a) the previous memory cell unit C_{t_1} modulated by the forget gate f_t, b) the input modulation gate $\widetilde{C}_t$ modulated by the input gate i_t. The output gate o_t is used to train the LSTM to decide how much of the memory cell is transferred to the next hidden state. Finally, the output h_t at time t is obtained by utilizing the tanh function on memory cell unit C_t, and modulating the resulting output by the output gate o_t. During the training of the LSTM, the multiple weights in the LSTM are reused at every time step. This forces the LSTM model to learn the generic temporal dynamics across the entire input sequence, instead of learning step-by-step temporal dynamics.

The LSTM has reported state-of-the-art recognition accuracy on different recognition problems including activity recognition [1]. In the work by Donahue et al. [1], the authors employ the LSTM to recognize actions from video sequences. To recognize the actions, the authors first sample N frames from the original video sequence, $\mathbf{V}_\alpha$, and generate the sampled sequence $\mathbf{V}_\beta$. This sampled sequence is given as the input to the LRCN. Within the LRCN each n-th frame in the sampled sequence, I_n, is given as input to an individual CNN, which extracts the n-th feature vector ϕ_n. The CNN model is based on the Caffenet [20] model. By adopting the formulation, N-CNN models extracts N feature vectors from the sampled sequences, which are represented by $\Phi = \{\phi_n\}_{n=1}^N$. A set of these CNN-features $\Psi = \{\Phi_s\}_{s=1}^S$ along with the corresponding set of action labels $\mathbf{g} = \{g_s\}_{s=1}^S$ are used to train the LSTM, as an action classifier. The trained LSTM model incorporates a softmax regression function to predict the action label at each time-step n. These individual LSTM predictions are then averaged to obtain the final action label for the given video sequence.

4) Training Phase: In the training phase of our algorithm, the Deconvnet is fine-tuned to extract the representative frames from the input video sequence. While the LRCN algorithm is fine-tuned to estimate the gesture labels.

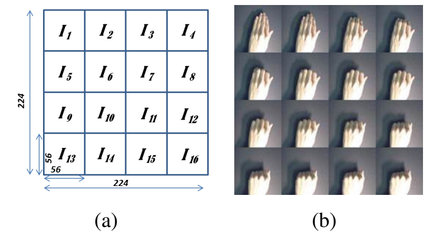
I_1 I_2 I_3 I_4
I_5 I_6 I_7 I_8
I_9 I_{10} I_{11} I_{12}
I_{13} I_{14} I_{15} I_{16}
224
56
56
224
(a)
(b)

Fig. 2. An illustration of the tiled image patterns and tiled binary patterns. (a) The dimensions and ordering of the N sampled frames (I_n) in the tiled image pattern P. The block indices corresponding to the 16 sampled frames are shown. (b) Tiled image pattern in Cambridge dataset.

a) Fine-tuning the Deconvnet.: Prior to fine-tuning the Deconvnet, we first generate the tiled image patterns and the tiled binary patterns. We next describe the characteristics of the tiled image and binary patterns. The tiled image patterns P and tiled binary patterns B are generated from the gesture-based video sequence, as single images with multiple non-overlapping blocks of smaller images (Fig 2-3). To construct the tiled image pattern, we first uniformly sample $N < T$ frames from the original video sequence $\mathbf{V}_\alpha$ to generate the sampled video sequence, $\mathbf{V}_\beta$. In this algorithm, we set N as 16 frames similar to the methodology adopted by Donahue et al. [1]. These 16 frames in $\mathbf{V}_\beta$ are then resized to images with the dimension $56 \times 56 \times 3$ and spatially ordered to generate the tiled image pattern P with dimension $224 \times 224 \times 3$. P encapsulates the gesture motion present in a video sequence within a single image. An illustration of the generated tiled image pattern and the spatial ordering are shown in Fig 2 and 3.

Following the generation of P, the tiled binary pattern B is constructed using the representative frames extracted from the sampled sequence. Similar to P, the tiled binary pattern is also a single image with multiple non-overlapping blocking. The image dimensions of B is similar to P with image size 224×224. The binary pattern B includes 16 non-overlapping blocks of size 56×56 as shown in Fig I-B4-a. The block indices in B directly correspond to the frame indices in $\mathbf{V}_\beta$, as the number of blocks are equal to the number of sampled frames.

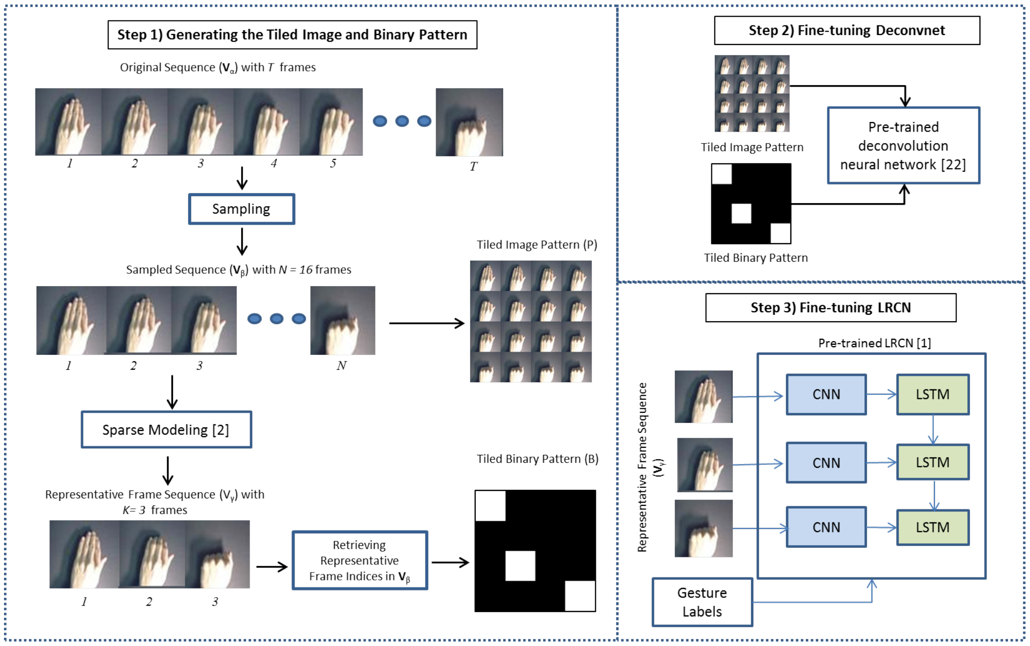
Step 1) Generating the Tiled Image and Binary Pattern
Original Sequence ($\mathbf{V}_\alpha$) with T frames
1
2
3
4
5
T
Sampling
Sampled Sequence ($\mathbf{V}_\beta$) with $N = 16$ frames
1
2
3
N
Tiled Image Pattern (P)
Sparse Modeling [2]
Tiled Binary Pattern (B)
Representative Frame Sequence ($\mathbf{V}_\gamma$) with $K = 3$ frames
1
2
3
Retrieving Representative Frame Indices in $\mathbf{V}_\beta$
Step 2) Fine-tuning Deconvnet
Tiled Image Pattern
Pre-trained deconvolution neural network [22]
Tiled Binary Pattern
Step 3) Fine-tuning LRCN
Pre-trained LRCN [1]
Representative Frame Sequence ($\mathbf{V}_\gamma$)
CNN
LSTM
CNN
LSTM
CNN
LSTM
Gesture Labels

Fig. 1. An overview of the training phase of the algorithm.

To construct B, the SMRF algorithm [2] is utilized to extract the representative frames from $\mathbf{V}_\beta$ using Eqn 3. The representative frames are derived using the indices of the non-zero rows of C (Eqn 3). We select K representative frames from the non-zero row indices, such that each row in B contains at most one representative frame. Note that each row in B corresponds to 4 frames out of the total 16 frames. Given the selected representative frames, their frame indices are used to construct B. First, we retrieve all the block indices in B corresponding to K representative frame indices. Subsequently, all the pixels within the retrieved block indices are set to one, generating a tiled binary pattern B as shown in Fig 3. In our algorithm, we set the value of K to be *three*, and retrieve *three* representative frames from the gesture sequence. The *three* representative frames are in line with the *three* gesture key frames used to describe an action as formulated by McNeil [21], the pre-stroke frame, the mid-stroke frame and the post-stroke frame. Example representative frames extracted from the video sequence are shown in Fig 3.

Given the constructed P and B, a training set is generated, $\mathbf{P} = \{P_s\}_{s=1}^{S}$ and $\mathbf{B} = \{B_s\}_{s=1}^{S}$, and used to fine-tune the Deconvnet within a semantic segmentation framework. In this paper, we fine-tune the original Deconvnet trained on the 20-class PASCAL VOC 2012 benchmark [19]. The Deconvnet architecture is adapted for our problem by retaining all the layers apart from the final output layer. Unlike the original architecture with 21 output channels, we only have two channels. The architecture for the Deconvnet along with the number and size of filters is presented in Table I. The input to the Deconvnet is the tiled image pattern P with $224 \times 224 \times 3$. While fine-tuning the Deconvnet, the weights and biases of the following layers are initialized with the pre-trained weights

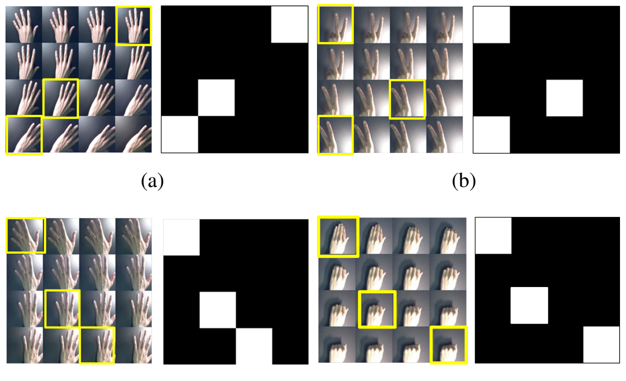

Fig. 3. Examples of the representative frames extracted from the video sequence using the SMRF algorithm [2] are shown. The video sequences are represented using the tiled image patterns, while the extracted representative frames are shown using the highlighted and non-zero blocks in the tiled image and binary patterns.

and biases: C1-C5, FC6,FC7, DC-FC and DC1-DC5. Owing to the initialization, the momentum, weight learning multiplier and bias learning multipliers of these layers are set to 0.9,1 and 2, respectively. In case of the final layer with 2 output channels, we do not perform any pre-initialization. Thus, the learning multipliers are set to 10 and 20, respectively. To account for pre-initialization, the overall learning rate is set α is set to 0.001.

b) Fine-tuning the LRCN.: In this paper, we fine-tune the original LRCN model pre-trained on the UCF-101 dataset [1] to detect the hand gestures. The architecture of our proposed LRCN model is similar to the original LRCN model, apart

TABLE I

THE FINE-TUNED DECONVNET ARCHITECTURE, P1-P5 AND UP1-UP5 ARE THE MAX POOLING AND UNPOOLING LAYERS, RESPECTIVELY, WITH SIZE 2 AND STRIDE 2. C1-C5 ARE THE CONVOLUTION LAYERS, DC1-DC5 ARE THE DECONVOLUTION LAYERS, AND FC REPRESENTS THE FULLY CONVOLUTION LAYERS.

1) Input	2) C1	3) C2	4) C3	5) C4	6) C5	7) FC
TIP Input $224 \times 224 \times 3$	$C1\_1(3,64)$ $C1\_2(3,64)$ $P1$	$C2\_1(3,128)$ $C2\_2(3,128)$ $P2$	$C3\_1(3,256)$ $C3\_2(3,256)$ $C3\_3(3,256)$ $P3$	$C4\_1(3,512)$ $C4\_2(3,512)$ $C4\_3(3,512)$ $P4$	$C5\_1(3,512)$ $C5\_2(3,512)$ $C5_3(3,512)$ $P5$	$FC1(7,4096)$ $FC2(1,4096)$
8) DC-FC	9) DC1	10) DC2	11) DC3	12) DC4	13) DC5	14) Output
$(7,512)$	$DC1_1(3,64)$ $DC1\_2(3,64)$ $UP1$	$DC2\_1(3,128)$ $DC2\_2(3,128)$ $UP2$	$DC3\_1(3,256)$ $DC3\_2(3,256)$ $DC3\_3(3,256)$ $UP3$	$DC4\_1(3,512)$ $DC4\_2(3,512)$ $DC4\_3(3,512)$ $UP4$	$DC5\_1(3,512)$ $DC5\_2(3,512)$ $DC5_3(3,512)$ $UP5$	2 channels with Softmax layer during training

from the number of LSTM hidden units, final output layer and CNN-based inputs. In our algorithm, we have 9 channels in the final output layer instead of 101 original channels. Additionally, we have 3 CNN-based inputs corresponding to the representative frames, instead of 16 CNN-based inputs in the original algorithm. Finally, we have 64 LSTM hidden units instead of the 256 LSTM hidden units in the original architecture.

The LRCN model is trained with set of representative frames-based video sequence $\Lambda = \{\mathbf{V}_\lambda(s)\}_{s=1}^{S}$ and corresponding set of gesture labels $\mathbf{g} = \{g_s\}_{s=1}^{S}$. While fine-tuning the LRCN, the weights of the CNN-based feature extraction are initialized with the pre-trained weights. Consequently, the momentum, the weight learning multiplier and bias learning multiplier are set to 0.9,1 and 2, respectively. On the other hand, the learning multiplier for the final layer with 9 output channels is set to 10 and 20, respectively, due to the absence of pre-initialization. The overall learning rate is set to 0.001, and the training iteration is set to 7500 iterations. An overview of the training phase is presented in Fig 1.

5) Testing Phase: Given a test video sequence $\widetilde{V_\alpha}$ with T frames, we first uniformly sample 16 to obtain the sampled video sequence $\widetilde{V_\beta}$. The sampled sequence is then used to generate the test tiled image pattern $\widetilde{P}$. The tiled image pattern $\widetilde{P}$ is then given as an input to the trained Deconvnet, which generates the tiled binary pattern $\widetilde{B}$ as the output.

From the estimated $\widetilde{B}$, we then select K block indices to obtain the K representative frames. A block index is selected if the number of non-zero pixels present within the block are above an empirical threshold. Using the selected block indices, we then retrieve the corresponding K frame indices in $\widetilde{V_\beta}$. These retrieved frame indices are used to generate the representative frame-based video sequence $\widetilde{V_\lambda}$. The generated sequence $\widetilde{V_\lambda}$ is then given as an input to the LRCN algorithm, which then estimates the gesture label. An overview of the testing phase is presented in Fig 4.

II. EXPERIMENTAL SECTION

We validate our proposed algorithm on the Cambridge gesture recognition dataset [22] using a 5-fold cross validation scheme. The Cambridge dataset contains 9 gesture classes,

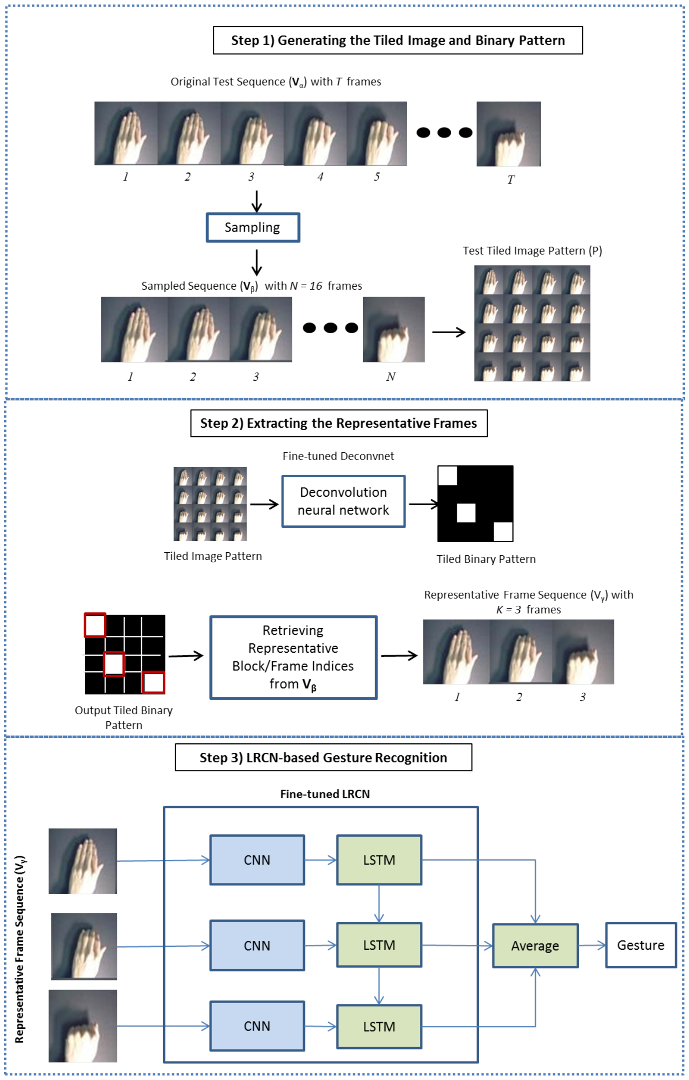
Step 1) Generating the Tiled Image and Binary Pattern
Original Test Sequence (V_α) with T frames
1
2
3
4
5
T
Sampling
Test Tiled Image Pattern (P)
Sampled Sequence (V_β) with $N = 16$ frames
1
2
3
N
Step 2) Extracting the Representative Frames
Fine-tuned Deconvnet
Deconvolution neural network
Tiled Image Pattern
Tiled Binary Pattern
Representative Frame Sequence (V_γ) with $K = 3$ frames
Retrieving Representative Block/Frame Indices from V_β
Output Tiled Binary Pattern
1
2
3
Step 3) LRCN-based Gesture Recognition
Fine-tuned LRCN
Representative Frame Sequence (V_γ)
CNN
LSTM
CNN
LSTM
CNN
LSTM
Average
Gesture

Fig. 4. An overview of the testing phase of the algorithm

with three variations in shape and motion. Each gesture class contains 100 video sequences with 5 illuminations, 10 trials and 2 subjects. Consequently, the dataset has significant inter-person and intra-person spatial and temporal variations. Example frames from the dataset are shown in Fig 5. We perform a comparative analysis with the LRCN-based baseline algorithm, and report better computational efficiency and classification accuracy. We implement our algorithm using on a Linux machine with Nvidia Geforce GTX 980 graphics card using Python with the Caffe [20] and scikit-library.

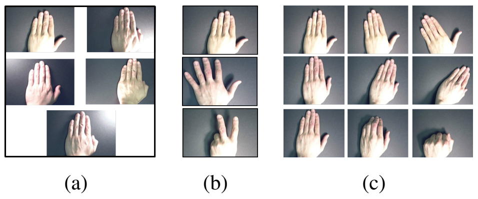

Fig. 5. An overview of the Cambridge public dataset [22] with (a) 5 varying illumination, (b) 3 varying shapes and (c) 3 varying motion.

A. Comparative Analysis

We perform a comparison of our proposed algorithm with the original 16-frame LRCN algorithm proposed by Donahue et al. [1]. To facilitate a fair comparison, the original LRCN algorithm is fine-tuned to detect the 9 hand gesture classes. Consequently, only the final layer of the original LRCN algorithm is changed from 101 channels to 9 channels. During the fine-tuning, the weights and biases of all the LRCN layers apart from the final layer are initialized with the pre-trained LRCN weights and biases. The momentum and learning multipliers are kept at $0.9, 1, 2$ for these layers. The overall learning rate is kept at 0.001, and we perform the fine-tuning for 7500 iterations.

Apart from the 16-frame LRCN algorithm, we also evaluate our algorithm with two deep learning-based single frame gesture classifiers. In the first classifier, the novel 16-framed tiled image pattern P is used to fine-tune the Alexnet-based CNN architecture [12] as a single frame-based gesture classifier (16-frame TIP-CNN). In the second classifier, we generate a tiled image pattern with 4 representative frames obtained from the video sequence using the SMRF algorithm. The 4 frame tiled image pattern is then used to fine-tune the Alexnet-based CNN architecture [12] (4-frame TIP-CNN). An overview of the single frame classifiers are shown in Fig. 6. In these baseline gesture classifiers, the final layer of the Alexnet is changed to reflect the 9 gesture classifiers. Additionally, similar to the other fine-tuning methodology, the momentum and learning parameters are kept at $0.9, 1, 2$. While the learning rate is kept at 0.001, and the fine-tuning is performed for 7500 iterations.

The classification accuracies of the different algorithms are reported in Table II. As shown in Table II, the proposed algorithm reports better classification accuracy than the baseline algorithms. We also report the failure of the single-frame CNN-based gesture classifiers to perform the gesture classification. The improved accuracy of the proposed algorithm, when compared with the 16-frame LRCN algorithm, can be attributed to the fewer representative frames used to train the LSTM's weights. More specifically, the LSTM framework reuses the weights to learn the generic temporal dynamics for the entire video sequence. Given this learning mechanism, the generic temporal dynamics across 3 representative frames are easier to model than the generic temporal dynamics across 16 sampled frames in the baseline LRCN. Comparing the computational times, our proposed algorithm takes 110ms per sequence, while the baseline 16-frame LRCN takes an average of 180ms per sequence. In our proposed algorithm, the *LRCN* takes 40ms to estimate the gesture label, while the Deconvnet takes 70ms to extract the representative frames.

TABLE II
THE MEAN AND STD.DEV OF THE GESTURE CLASSIFICATION ACCURACIES FOR THE CAMBRIDGE GESTURE DATASET.

Algorithm	Classification Accuracy %
Proposed Algo.	90.9 ± 1.1
16-frame LRCN.	86.5 ± 2.7
4-frame TIP-CNN.	14.6 ± 2.1
16-frame TIP-CNN.	13.2 ± 1.2

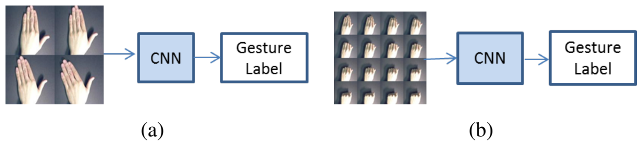

Fig. 6. An illustration of the CNN-based single frame gesture classifiers with (a) 4-frame tiled image pattern input and (b) 16-frame tiled image pattern input.

While analyzing our experimental results, we observe that the missed detections are attributed to similar motions, such as left-right motion. However, even when the wrong motion is detected, our proposed algorithm, detects the correct shape of the hand.

B. Parameter Analysis

We evaluate the following parameters of our proposed algorithm using a 5-fold cross validation: number of hidden LSTM units, fine-tuning mechanism and representative frames.

1) Number of LSTM Hidden Units: In this experiment, we vary the number of LSTM hidden units and report the classification accuracy in Table III. We perform the experiments with 64, 128 and 256 LSTM units within the proposed algorithm. It can be observed that the classification accuracy decreases with the number of hidden units. The training parameters are the same across the different LSTM units.

2) Fine-Tuning Mechanism: In this experiment, we evaluate the fine-tuning mechanism by directly training the 3-frame LRCN framework, without any pre-initialization. As a result, the learning rate is kept at 0.01, while the number of iterations are increased to 10000. The results as shown in Table IV, demonstrate the advantages of fine-tuning.

TABLE III
THE MEAN AND STD.DEV OF THE GESTURE CLASSIFICATION ACCURACIES FOR THE CAMBRIDGE GESTURE DATASET.

Algorithm	Classification Accuracy %
Proposed Algo-64 LSTM units	90.9 ± 1.1
Proposed Algo-128 LSTM units	87.3 ± 3.1
Proposed Algo-256 LSTM units	86.8 ± 2.6

TABLE IV
THE MEAN AND STD.DEV OF THE GESTURE CLASSIFICATION ACCURACIES FOR THE CAMBRIDGE GESTURE DATASET.

Algorithm	Classification Accuracy %
Proposed Algo-Fine tuning	90.9 ± 1.1
Proposed Algo-Directly trained	65.1 ± 1.2

3) Representative Frames: Finally, we evaluate the methodology of selecting 3 representative frames in our algorithm. In this experiment, 3 random frames selected and ordered from each video sequence are used to fine-tune the LRCN. Additionally, the Deconvnet is not trained with the tiled image and binary patterns. Consequently, during the testing phase, 3 random frames selected from the test video sequence are given as an input to the fine-tuned LRCN, and the gesture label is obtained. The results as shown in Table V demonstrate the enhanced classification accuracy of our proposed framework.

TABLE V
THE MEAN AND STD.DEV OF THE GESTURE CLASSIFICATION ACCURACIES FOR THE CAMBRIDGE GESTURE DATASET.

Algorithm	Classification Accuracy %
Proposed Algo	90.9 ± 1.1
3 random frames-LRCN	87.6 ± 1.4

III. CONCLUSION

In this paper, a fast deep learning hand gesture recognition algorithm is proposed for intelligent vehicle applications. The long-recurrent convolutional neural network is utilized to perform the hand gesture recognition. The classification accuracy and computational efficiency of the long-recurrent convolutional neural network is improved by extracting 3 representative frames from the video sequence. To extract the representative frames, the semantic segmentation-based deconvolutional neural network is utilized. Novel tiled image and binary patterns are proposed to train the deconvolutional network. The tiled binary patterns used for training are generated using the representative frames extracted from the video sequences by the sparse modeling algorithm. Our proposed algorithm is evaluated on the Cambridge public dataset, and a comparative analysis with the baseline algorithms are performed. Our proposed algorithm reports better classification accuracy and computational efficiency than the baseline algorithms. In our future work, we will evaluate with a larger dataset, and improve the computational efficiency to achieve real-time performance.

REFERENCES

[1] J. Donahue, L. A. Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell, "Long-term recurrent convolutional networks for visual recognition and description," in *CVPR*, 2015.

[2] E. Elhamifar, G. Sapiro, and R. Vidal, "See all by looking at a few: Sparse modeling for finding representative objects," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[3] S. Mitra and T. Acharya, "Gesture recognition: A survey," *IEEE Transactions on Systems, Man and Cybernetics - Part C*, vol. 37, no. 3, pp. 311–324, 2007.

[4] P. Trindade, J. Lobo, and J. P. Barreto, "Hand gesture recognition using color and depth images enhanced with hand angular pose data," in *IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*, 2012.

[5] N. H. Dardas and N. D. Georganas, "Real-time hand gesture detection and recognition using bag-of-features and support vector machine techniques." *IEEE Trans. Instrumentation and Measurement*, vol. 60, no. 11, pp. 3592–3607, 2011.

[6] T. Starner, A. Pentland, and J. Weaver, "Real-time american sign language recognition using desk and wearable computer based video," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 12, pp. 1371–1375, Dec. 1998.

[7] T. Ahmed, "A neural network based real time hand gesture recognition system," *International Journal of Computer Applications*, vol. 59, no. 4, pp. 17–22, December 2012.

[8] A. Boyali and N. Hashimoto, "Block-sparse representation classification based gesture recognition approach for a robotic wheelchair," in *2014 IEEE Intelligent Vehicles Symposium Proceedings*, 2014, pp. 1133–1138.

[9] P. Molchanov, S. Gupta, K. Kim, and K. Pulli, "Multi-sensor system for driver's hand-gesture recognition," in *IEEE International Conference and Workshops on Automatic Face and Gesture Recognition*, 2015.

[10] N. Neverova, C. Wolf, G. W. Taylor, and F. Nebout, *Multi-scale Deep Learning for Gesture Detection and Localization*, 2015.

[11] E. Ohn-Bar and M. M. Trivedi, "Hand gesture recognition in real time for automotive interfaces: A multimodal vision-based approach and evaluations," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 6, pp. 2368–2377, Dec 2014.

[12] A. Krizhevsky, I. Sutskever, and G. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proceedings of Neural Information Processing Systems*, 2012.

[13] P. Molchanov, S. Gupta, K. Kim, and J. Kautz, "Hand gesture recognition with 3d convolutional neural networks," in *2015 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2015.

[14] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[15] L. Pigou, S. Dieleman, P.-J. Kindermans, and B. Schrauwen, *Sign Language Recognition Using Convolutional Neural Networks*, 2015.

[16] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," 2014.

[17] K. Murakami and H. Taguchi, "Gesture recognition using recurrent neural networks," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 1991.

[18] F. J. Ordez and D. Roggen, "Deep convolutional and lstm recurrent neural networks for multimodal wearable activity recognition," *Sensors*, vol. 16, no. 1, p. 115, 2016.

[19] H. Noh, S. Hong, and B. Han, "Learning deconvolution network for semantic segmentation." in *ICCV*, 2015.

[20] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guaddarrame, and T. Darrel, "Caffe: Convolutional architecture for fast feature embedding," in *arXiv preprint arXiv:1408.5093*, 2014.

[21] D. McNeil, *Language and Gestures*, 2000.

[22] T.-K. Kim and R. Cipolla, "Canonical correlation analysis of video volume tensors for action categorization and detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 8, pp. 1415–1428, 2009.