IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.Doi Number

Hand Gesture Recognition for Sign Language Using 3DCNN

Muneer Al-Hammadi, Ghulam Muhammad, *Senior Member, IEEE*, Wadood Abdul, *Member, IEEE*, Mansour Alsulaiman, Mohamed A. Bencherif, and Mohamed Amine Mekhtiche

Department of Computer Engineering, College of Computer and Information Sciences, King Saud University, Riyadh 11543, Saudi Arabia
Center of Smart Robotics Research, College of Computer and Information Sciences, King Saud University, Riyadh 11543, Saudi Arabia

Corresponding authors: Mansour Alsulaiman (email: msuliman@ksu.edu.sa) and Ghulam Muhammad (e-mail: ghulam@ksu.edu.sa).
The authors would like to thank the Deanship of Scientific Research (DSR), King Saud University, Riyadh, Saudi Arabia, for supporting this research through the research group program with number RG- 1437-018.

ABSTRACT Recently, automatic hand gesture recognition has gained increasing importance for two principal reasons: the growth of the deaf and hearing-impaired population, and the development of vision-based applications and touchless control on ubiquitous devices. As hand gesture recognition is at the core of sign language analysis a robust hand gesture recognition system should consider both spatial and temporal features. Unfortunately, finding discriminative spatiotemporal descriptors for a hand gesture sequence is not a trivial task. In this study, we proposed an efficient deep convolutional neural networks approach for hand gesture recognition. The proposed approach employed transfer learning to beat the scarcity of a large labeled hand gesture dataset. We evaluated it using three gesture datasets from color videos: 40, 23, and 10 classes were used from these datasets. The approach obtained recognition rates of 98.12%, 100%, and 76.67% on the three datasets, respectively for the signer-dependent mode. For the signer-independent mode, it obtained recognition rates of 84.38%, 34.9%, and 70% on the three datasets, respectively.

INDEX TERMS 3DCNN, computer vision, deep learning, hand gesture recognition, sign language recognition, transfer learning

I. INTRODUCTION

The hand gesture is a nonverbal form of communication. It consists of linguistic content that carries a large amount of information in sign language. It also plays a pivotal role in human-computer interaction (HCI) systems. Therefore, automatic hand gesture recognition is in high demand. Since the end of the last century, this field has attracted the attention of many researchers. The importance of automatic hand gesture recognition has increased for the following reasons [1]: (1) the growth of the deaf and hard-of-hearing populations, and (2) the extended use of vision-based and touchless applications and devices such as video games, smart TV control, and virtual reality applications.

Robust hand gesture recognition is required as a part of sign language interpretation to help hearing-impaired people. There is a significant communication gap between people who can hear and hearing-impaired people. A translation system between gestural language and verbal language will bridge this communication gap. This translation system will facilitate the lives of hearing-impaired people and help them to integrate with society. Unlike sign language translation, hand gesture recognition techniques involve HCI to a great degree. Today, HCI has a wide range of applications from video games to telesurgery. As with all time-varying signals, hand gestures cannot be directly compared in Euclidean space because of their temporal dependency. This dependency indicates important discriminative features. Temporal misalignment, in addition to massive irrelevant regions in every frame, makes it very hard to extract representative hand-engineered features for hand gestures. For conventional classifiers to perform well, the extracted features should implicate vigorous descriptors. These descriptors code enough information for the inter-frames temporal dependency, as well as the hand position, shape and orientation in each frame. The computed features should be able to minimize the effect of different circumstances like background clutter and occlusions. Therefore, we employed deep learning in this paper as a promising solution.

In recent years, many researchers have efficiently exploited convolutional neural networks (CNNs) deep architectures for feature engineering. CNNs have shown

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2990434, IEEE Access

excellent performance in fields such as object and speech recognition, image classification, and edge distribution [2][3][4], and human activity recognition [5][6][7]. The existence of large datasets that comprise millions of annotated samples is the main reason behind such excellent performance. Unfortunately, the requirement for a large labeled dataset is not met in the case of hand gestures. To beat the scarcity of labeled dataset for fitting deep architectures, the transfer learning is investigated in this study. We propose a well-adapted deep architecture for automatic hand gesture recognition. The main contributions of this study are as follows:

(1) A method to normalize the spatial dimensions of gesture videos based on the facial position, facial length, and human body part ratios. The signer does not need to be in the center of the frame or be a fixed distance from the camera.

(2) A 3DCNN model to learn region-based spatiotemporal features for hand gestures. The input of this model is a sequence of RGB frames captured by a basic camera. It does not require other input channels, colored gloves, or a complex setup.

(3) Developing different fusion techniques to globalize the local features learned by the 3DCNN model and comparing their performance.

The rest of this paper is organized as follows. Section II reviews related work in the literature. Section III provides the description of the dataset. The proposed approach is presented in Section IV. The experimental results and discussions are presented in Section V. The research conclusions are presented Section VI.

II. RELATED WORK

As a form of the human-computer interaction, hand gesture recognition has attracted the attention of many researchers since the end of the last century. Based on the acquisition technique of the input data, the research efforts can be categorized into two approaches, the contact-based approach and the vision-based approach. In the contact-based approach, the signer should be familiar with interfacing devices like motion sensors, data gloves, position trackers, and accelerometers, to collect hand gesture data [8][9][10][11][12]. This approach disadvantages are the high cost and discomfort to the signer. The studies in the vision-based approach revoked these drawbacks. various imaging devices such as cameras are used for hand gesture recording (without contacting the signer body or restricting his movement).

In one of the oldest papers published in sign language recognition, the authors proposed artificial neural networks to recognize 42 finger alphabets (static gestures). [13]. Time-delay neural networks were proposed for hand gesture recognition by Yang et al. [14]. They utilized skin color and motion for hand segmentation and tracking. The reported accuracy for 40 gestures was 93.42%. The method presented in [15] utilized a block-based histogram representation of the optical flow (BHOF) for gesture recognition. It was evaluated using three different datasets. The reported accuracies were 93.33% on the RWTH-BOSTON-50 dataset [16], 60.0% on the American sign language dataset (Purdue RVL-SLLL) [17], and 85.9% on the American sign language lexicon [18]. The presented method in [19] compressed the motion information in a video segment into a single image via temporal prediction and accumulated differences. The K nearest neighbors (K-NNs) and Bayesian classifiers were then used to evaluate the frequency transformation of this representative image. The reported recognition rate on a database of 23 isolated Arabic gestures was 100%.

The approach proposed in [20] used hidden Markov models (HMMs) on the discrete cosine transform (DCT) coefficients of consecutive frames. This was evaluated on the same dataset of 23 isolated gestures used in [19] and a recognition accuracy of 94% was obtained. The local binary pattern in the spatiotemporal representation of three orthogonal planes (LBP-TOP) was investigated with a support vector machine (SVM) in [21]. For the same 23-gesture dataset in [19], a recognition rate of 99.5% was reported. Abid et al. presented a dynamic sign language recognition system using a bag of features and a local part model approach [22]. The experimental results yielded a 97% recognition rate for six dynamic gestures. The approach presented in [23] involves the motion data in the optical flow in addition to the RGB frames. This information fusion is performed at the data level. The fused information is then used to adapt the pre-trained inception architecture. The reported accuracy for this approach was 96.28% on the Jester dataset, 56.7% on the ChaLearn dataset, and 84.7% on the nvGesture dataset.

A two-streams 3DCNN architecture was proposed by Molchanov et al. The interleaved volumes of precomputed Sobel gradient and depth maps were taken with two different resolutions as the input for the two streams [24]. The output of each stream represented the class membership probability values. Both streams were fused by performing an element-wise multiplication. An accuracy of 77.5% on the VIVA dataset was reported. Poon et al. proposed a bimanual hand gesture recognition technique [25]. They fit independent SVM classifiers on the shape and color-encoded features of three different views. These views were the front, right, and left. The drawback of the proposed approach is that the input is static images of the small region of the hands; this is not applicable for real dynamic hand gesture recognition.

The proposed approach in [26] used the ResNet architecture to fetch features from each frame and encode the entire video in a single matrix. A CNN was followed to extract the spatiotemporal features' evolution. A recognition accuracy of 95.31% using the Jester dataset was reported.

Even though there is intensive research conducted on gesture recognition, the presented solutions are limited. They have shortcomings or only operate under constraints. The methods reviewed here require colored gloves, complex hardware, or

specific dressing such as in [25][27][28]. Others tested only simple static gestures [13][28], a few number of gestures [22], or tested only in signer-dependent mode [20][19]. They may also require inputs from multiple channels with heavy preprocessing steps [23][24]. The vision-based approach presented in this paper is designed to overcome such barriers and bridge this gap. It does not require complex hardware colored gloves or special clothing and was tested on datasets of various sizes in signer-dependent and signer-independent modes. In the signer-dependent mode, the dataset is split for training and testing randomly, while in the signer-independent mode, the signers in the testing dataset should not be included in the training dataset (more details in sec **V**).

III. DATASETS

The presented approach in this paper was evaluated by using three different hand gesture databases:

A. KING SAUD UNIVERSITY SAUDI SIGN LANGUAGE (KSU-SSL) DATASET

This dataset was created by the Center of Smart Robotics Research and Higher Education Program for the Deaf and Hard of Hearing at King Saud University. The dataset comprises selected gestures from the common Saudi sign language words and expressions. These expressions contain single-handed actions as well as two-handed actions. 40 subjects were involved in recording this dataset. Some of the subjects were deaf people. The nondeaf subjects were guided by sign language experts. Each subject was asked to perform the gestures five times during five different sessions. Different devices such as RGB cameras and Microsoft Kinect were used for recording this dataset. The dataset recording sessions were performed without restrictions in an uncontrolled environment. There were no constraints on the clothing of the participants, lighting conditions, or background color. There was also a high degree of variation in the distances between the camcorder device and the signers.

Because of this restriction-free recording, KSU-SSL is a challenging dataset. In most cases, the signer's hands are blurred and difficult to detect and track. The first row in Fig. 1 shows sample frames from this dataset and its uncontrolled recording environment. To evaluate our approach on the KSU-SSL dataset, we selected forty gesture classes. Table I listed the selected classes.

B. ARABIC SIGN LANGUAGE (ARSL) DATASET

This dataset was created by the College of Engineering at the American University of Sharjah [19]. It contains 23 gestures performed by three participants. Each subject was asked to repeat the gestures 50 times. Therefore, there are 150 samples in the dataset for each gesture. An analog camcorder was used to record this dataset. Table II lists all the gestures in this dataset. The second row in Fig. 1 displays sample frames from this dataset.

TABLE I
SELECTED GESTURES FROM KSU-SSL DATASET

#	KSU-SSL Gesture	English Meaning	#	KSU-SSL Gesture	English Meaning
1	السلام عليكم	Peace be upon you	21	لغة الإشارة	Sign language
2	حبوب الدواء	Pills	22	كيف حالك؟	How are you?
3	شكراً	Thanks	23	تفضل	Come in
4	متعب	Tired	24	أصم	Deaf
5	أخت	Sister	25	أم	Mother
6	أين؟	Where?	26	أسرة	Family
7	السبب	Reason	27	دكتور	Doctor
8	بارد	Cold	28	مسجد	Mosque
9	صلاة	Prayer	29	مساء	Evening
10	ملف	File	30	إجازة	Vacation
11	جامعة	University	31	أخ	Brother
12	مرحبا	Hello	32	ملك	King
13	حار	Hot	33	مدير	Manager
14	الملك سعود	King Saud	34	بماذا تشعر	What are you feeling?
15	مستشفى	Hospital	35	اجتماع	Meeting
16	ألم	Pain	36	أب	Father
17	وفاة	Death	37	صباح	Morning
18	عملية جراحية	Surgery	38	أخصائي نفسي	Psychologist
19	لغة الاشارة العربية	Arabic sign language	39	لغة الاشارة الإنجليزية	English sign language
20	آسف	sorry	40	اسم	Name

TABLE II
ARSL DATASET GESTURES

#	ArSL Gesture	English Meaning	#	ArSL Gesture	English Meaning
1	صديق	Friend	13	أنا	I
2	جار	Neighbor	14	أكل	Ate
3	ضيف	Guest	15	نام	Slept
4	هدية	Gift	16	يشرب	Drink
5	عدو	Enemy	17	يستيقظ	Wake up
6	السلام عليكم	Peace be upon you	18	يسمع	Hear
7	أهلا وسهلا	Welcome	19	يسكت	Stop talking
8	شكراً	Thank you	20	يشم	Sniff
9	تفضل	Get in	21	يساعد	Help
10	عيب	Shame	22	أمس	Yesterday
11	بيت	Home	23	ذهب	Went
12	أتى	Came			

C. PURDUE RVL-SLLL AMERICAN SIGN LANGUAGE DATASET

This dataset consists of 43 classes of isolated hand gestures [17]. Fourteen fluent deaf were involved in recording this dataset. The recording sessions took place in a professional recording studio under perfect lighting conditions. We found only one method evaluated on ten gestures from this dataset. To evaluate the proposed approach, we selected the same gestures to make a fair comparison with that method. The selected gestures were "Away," "Up," "Down," "Left,"

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2990434, IEEE Access

"Right," "Inform," "Happen," "Skilled," "Illegal," and "Influence." The third row in Fig. 1 shows sample frames from this dataset.

IV. PROPOSED SYSTEM

In this study, we utilized a 3DCNN architecture for spatiotemporal feature learning using two approaches.

In the first approach, 3DCNN was used to extract the features from the entire video sample, while a SoftMax layer was used for classification. In the second approach, we aimed to enhance the temporal dependency of the video frames. To achieve this, the same 3DCNN architecture was trained to extract the features from different regions in the video sample. We then investigated different techniques for feature fusion.

A. SINGLE 3DCNN STRUCTURE

The system proposed in the first approach is illustrated in Fig. 2. It consists of three main phases: video preprocessing, feature learning, and classification.

1) VIDEO PREPROCESSING

The first step in the preprocessing phase was converting the input video into RGB frames sequence. Because the video sequences had different durations, linear sampling was applied to normalize all the sequences to a fixed length of 16 frames, as the original model was fit on video sequences of 16 frames each. The corresponding indices of 16 frames are calculated as in (1).

$$index_i = \text{round}\left(\frac{\text{len}(input)}{16} \times i\right), i \in \{1,16\} \quad (1)$$

where, len(*input*) is the length of the input sequence.

Other techniques such as the Bag of Visual Words have been used in the literature to normalize the temporal dimension of the input videos. Linear sampling was preferred in this work to preserve the order of the selected frames. The order of the selected frames indicates essential discriminative features in gesture recognition. However, spatial dimension normalization was also required to overcome variations in the heights and distances of the signers from the camera. We achieved this normalization in two steps:

- First, we employed the face detection algorithm proposed by Viola and Jones [29] to detect the face of the signer in the first frame of the sequence.
- Then, based on the position and height of the detected face, we used the human body part ratios [30] to estimate the height and width of the gesture space to be cropped in all frames, as illustrated in Fig. 3.

The final step in the preprocessing phase was to resize the cropped square frames in all input videos to a fixed size of 112 × 112 pixels while maintaining the same aspect ratio.

The RGB channels of each gesture sample were also normalized separately such that each channel had a zero mean and unit variance.

FIGURE 1. Sample frames from hand gesture datasets.

This resizing and normalization reduced the computation cost and training convergence of the model in the next phase. The final inputs to the feature learning phase were 112 × 112 × 16 × 3 volumes.

2) FEATURE LEARNING

A deep 3DCNN is proposed for feature learning, to extract the local spatiotemporal features of gesture sequences. Transfer learning was employed here to beat the scarcity of a large labeled dataset of gestures. We started with a pre-trained version of the 3DCNN structure as in [31]. This had already been trained using millions of samples of human action recognition [6].

After excluding the output layer, the structure consisted of six consecutive blocks. The first two blocks have a single 3DCNN layer each. The first layer comprises 64 kernels and the second comprises 128 kernels. The third block contains two 3DCNN layers, each of 256 kernels. The fourth block contains two 3DCNN layers, each of 512 kernels. The fifth block contains two 3DCNN layers, each of 512 kernels and a zero-padding layer. The sixth block consists of two dense layers with 4096 neurons each. These two layers globalize the feature modeling. The output of each of the first four blocks transits to the successive block through a max-pooling layer.

All 3DCNN kernels are (3 × 3 × 3) in size with a stride of (1 × 1 × 1). All max-pooling kernels are of size (2 × 2 × 2) with stride (2 × 2 × 2), except for the max-pooling kernel that follows the first block, which is of size (1 × 2 × 2) with stride (1 × 2 × 2) to preserve the temporal information in the early stage. A simple nonlinear function rectified linear unit (ReLU) was used for activation, as shown in (2). This function was preferred as it has a simple derivative to speed up large-network training [32].

$$\boldsymbol{ReLU(x)} = \begin{cases} \boldsymbol{x}, & \boldsymbol{x \geq 0} \\ \boldsymbol{0}, & \boldsymbol{x < 0} \end{cases} \quad (2)$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2990434, IEEE Access

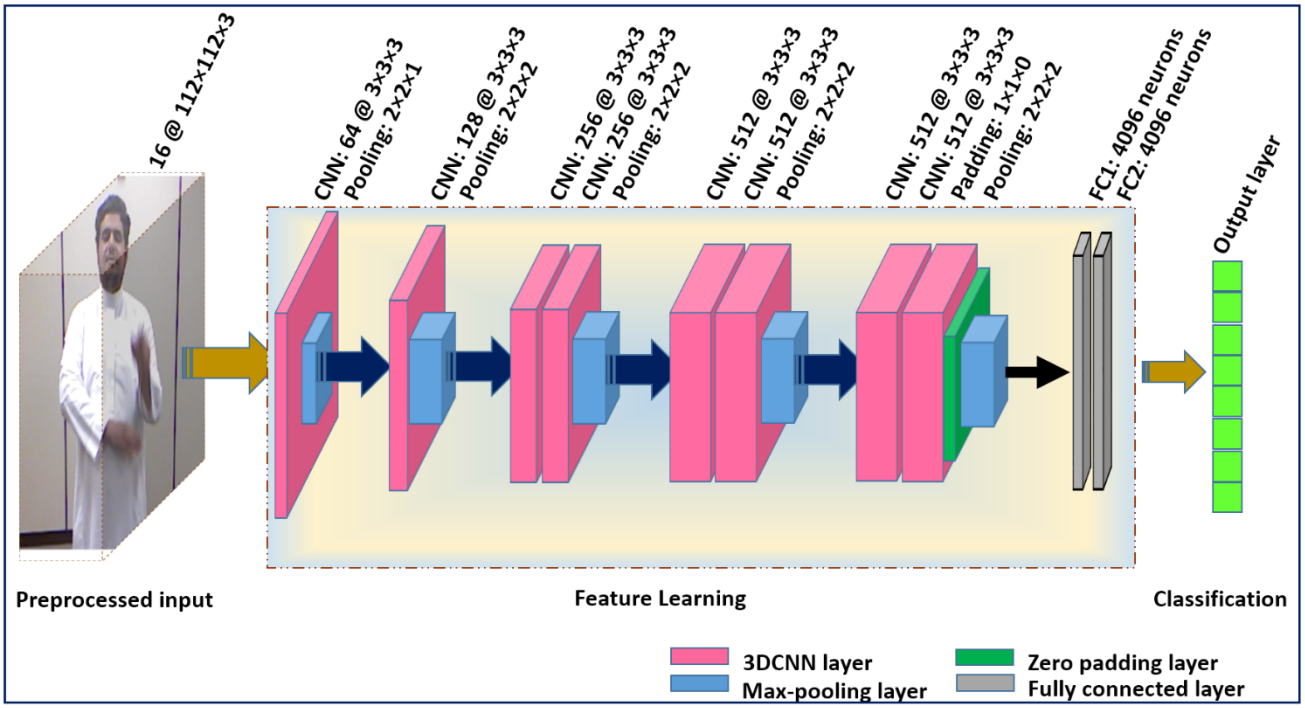

FIGURE 2. **Single 3DCNN-based structure.**

Each 3D kernel in the first layer is convolved to a volume of the 16 input stacking frames to produce a spatiotemporal feature map. The 3D kernels in the successive layers are similarly convolved to a volume of stacking feature maps produced by the predecessor layers.

In general, the value at any position (x, y, z) on the Kth feature map in the Lth layer is calculated as

$$\boldsymbol{V}_{LK}^{xyz} = ReLU\left(b_{LK} + \sum_m \sum_{p=0}^{P_L-1} \sum_{q=0}^{Q_L-1} \sum_{r=0}^{R_L-1} \boldsymbol{W}_{LKm}^{pqr} \boldsymbol{V}_{(L-1)m}^{(x+p)(y+q)(z+r)}\right) \quad (3)$$

where P_L, Q_L, and R_L are the dimensions of the 3D kernel and $\boldsymbol{W}_{LKm}^{pqr}$ is the (p, q, r) th value of the kernel connected to the mth feature map in the preceding layer. The two fully connected layers globalize the feature modeling where the last layer outputs a feature vector length of 4096 to represent each sample.

3) CLASSIFICATION

The features extracted in the previous phase are fed into a SoftMax layer for classification, as illustrated in Fig. 2. The SoftMax activation function as defined in (4) outputs the probability of each class. The predicted output is the class with the maximum probability.

$$SoftMax(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{k} e^{x_j}} \quad (4)$$

where x_i is the corresponding class, and k is the number of classes.

The proposed model is fine-tuned by backpropagation with negative log likelihood to recognize the hand gestures. Only the fully connected and convolutional layers in the last two blocks were optimized on the datasets of gestures, while the other layers of the architecture were frozen. To optimize the model parameters, we utilized stochastic gradient descent (SGD) with an adaptive learning rate. We used values of 10^{-4} for initial learning rate, 10^{-6} for decay, and 0.9 for momentum.

To avoid overfitting and enhance the model generalization on test data, we applied 50% dropout after each fully connected layer [33].

B. FUSION OF PARALLEL 3DCNN STRUCTURE

In the second approach, the proposed system enhanced the temporal contribution of the extracted features. To achieve this, in the preprocessing phase, linear sampling is applied to select 32 frames instead of 16 frames. Thereafter, three instances of the deep 3DCNN structure described in the previous approach are utilized to learn the spatiotemporal features in the beginning, middle, and end of the video sequence. The selected frames are divided into three short clips of 16 frames each with a 50% overlap. Each deep 3DCNN instance is trained to extract the features from one of the three clips. Various techniques are then utilized to fuse the features extracted from different parts of the video.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2990434, IEEE Access

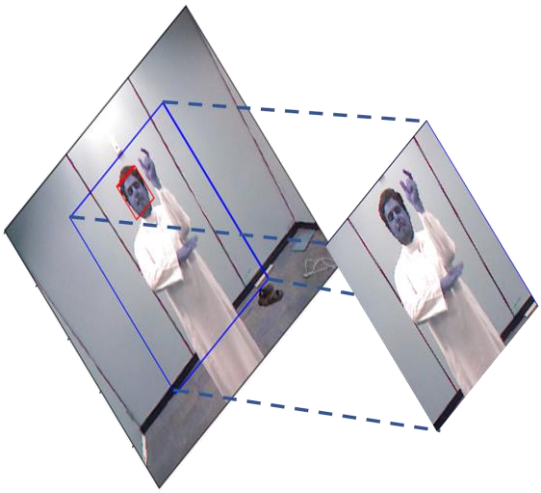

FIGURE 3. Cropping the signing space.

Finally, the fused features are forwarded to the SoftMax layer for classification. Fig. 4 illustrates a general diagram of the second approach.

Feature fusion

Three techniques for feature fusion, multilayer perceptron (MLP) neural network, long short-term memory (LSTM) network, and stacked autoencoder were investigated.

1) MLP FUSION

MLP processes the input features through a series of computational nodes called neurons.

These neurons are grouped into consecutive layers and interconnected with one another via weighted connections. These neurons transform the features by performing nonlinear operations. The features are then projected into a space where the input becomes linearly separable [34]. MLP architectures with different numbers of layers were investigated in this research.

2) LSTM FUSION

An LSTM is a recurrent neural network (RNN) adopted to learn long-term contextual dependencies from learned local feature sequences [4].

Fig. 5 illustrates the basic building block of LSTM networks. The behavior of this LSTM unit is controlled by three gates: the input gate, the forget gate, and the output gate. The input is fed into these gates to control which operations are to be performed by the unit. The memory state and output of an LSTM unit are updated at each time step [34]. The LSTM transition equations at time step t can be formulated as

$$\boldsymbol{i}_t = \sigma(\boldsymbol{W}_i.[\boldsymbol{c}_{t-1}, \boldsymbol{h}_{t-1}, \boldsymbol{x}_t] + \boldsymbol{b}_i) \quad (5)$$

$$\boldsymbol{f}_t = \sigma(\boldsymbol{W}_f.[\boldsymbol{c}_{t-1}, \boldsymbol{h}_{t-1}, \boldsymbol{x}_t] + \boldsymbol{b}_f) \quad (6)$$

$$\boldsymbol{c}_t = \boldsymbol{f}_t.\boldsymbol{c}_{t-1} + \boldsymbol{i}_t.\tilde{\boldsymbol{c}}_t \quad (7)$$

$$\boldsymbol{o}_t = \sigma(\boldsymbol{W}_o.[\boldsymbol{c}_t, \boldsymbol{h}_{t-1}, \boldsymbol{x}_t] + \boldsymbol{b}_o) \quad (8)$$

$$\boldsymbol{h}_t = \boldsymbol{o}_t.tanh(\boldsymbol{c}_t) \quad (9)$$

where $\boldsymbol{x}_t$, $\boldsymbol{h}_t$, and $\boldsymbol{c}_t$ are the input vector, output vector, and memory state, respectively, at time t. Terms $\boldsymbol{i}$, $\boldsymbol{f}$, $\boldsymbol{o}$, and $\tilde{\boldsymbol{c}}$ represent the input gate, forget gate, output gate, and cell activation, respectively, all of which are the same size as the input vector. Term σ represents nonlinear sigmoid functions.

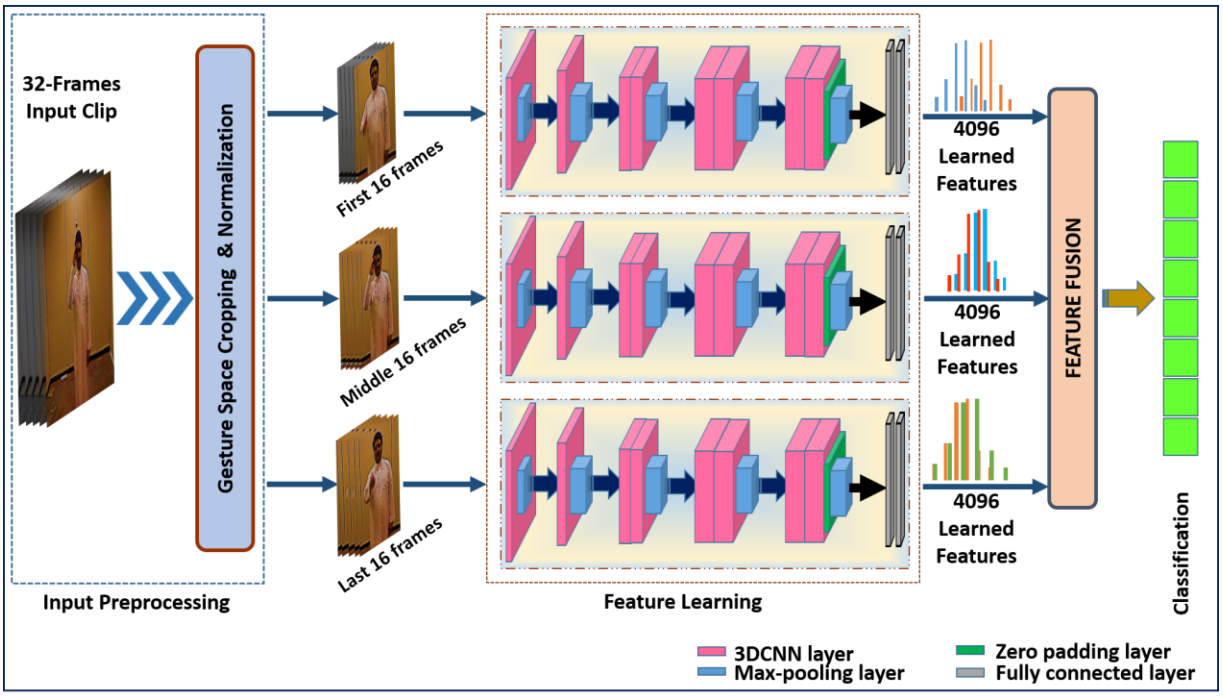

FIGURE 4. Fusion of parallel 3DCNN structure.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2990434, IEEE Access

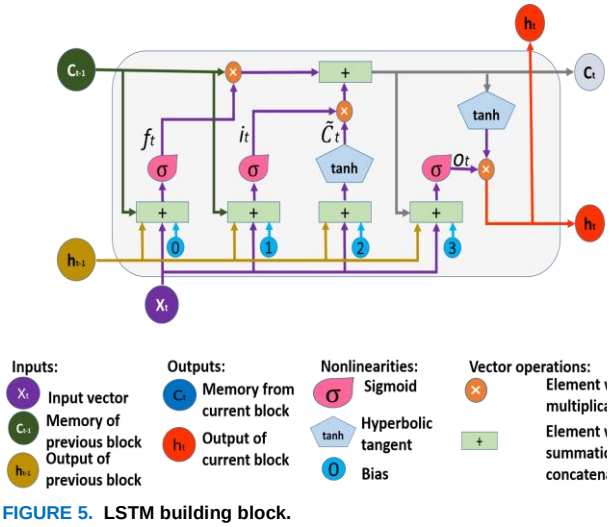

FIGURE 5. **LSTM building block.**

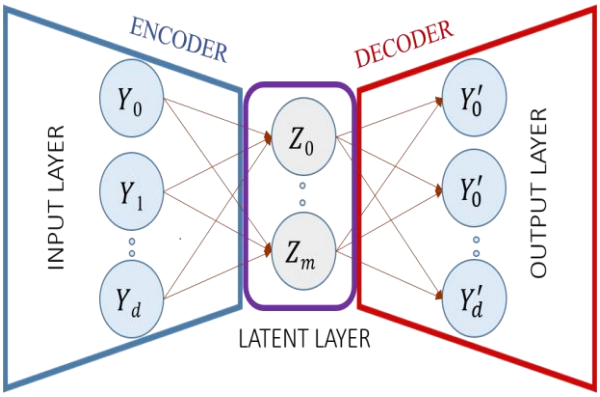

FIGURE 6. **Single-hidden-layer autoencoder.**

We utilized recurrent LSTM architectures of different numbers of layers to enhance the automatic feature representation and model the temporal dependencies.

3) AUTOENCODER FUSION

The simplest form of an autoencoder is a single hidden fully connected layer with input and output layers, as shown in Fig. 6. The number of nodes in the output layer must be the same as in the input layer. The autoencoder creates a new representation for the input data through a pair of maps $\boldsymbol{y} \xrightarrow{f} \boldsymbol{z} \xrightarrow{g} \boldsymbol{y'}$. The first one is the encoder map $\boldsymbol{z} = f(\boldsymbol{y})$, and the second one is the decoder map $\boldsymbol{y'} = g(\boldsymbol{z})$. The input data dimension is reduced by the encoder. The encoder transforms the input data of dimension d to a smaller dimension m [35].

The decoder reconstructs the input data from the reduced dimension m back to the original dimension d. During training, the autoencoder is usually forced to prioritize which aspects of the input should be kept [36]. The autoencoder maps an input point y to a code z via a sigmoid activation function as

$$\boldsymbol{z} = f(\boldsymbol{y}) = \sigma(\boldsymbol{W}\boldsymbol{y}^T + \boldsymbol{b}) \tag{10}$$

where $\mathbf{W}$ is a weight matrix, and b is a bias vector. The z code is also termed the latent representation of point y. The sigmoid function transforms the input values to the activation values, which are mostly either close to zero or 1.

The decoder then maps this activation to the reconstructed $\boldsymbol{y'}$ to the same dimensional space of $\boldsymbol{y}$ such that

$$\boldsymbol{y'} = \boldsymbol{g}(\boldsymbol{z}) = \boldsymbol{\sigma}(\boldsymbol{W'}\boldsymbol{z} + \bar{\boldsymbol{b}})^{\boldsymbol{T}} \tag{11}$$

where the weight matrix of the decoder is often the transpose of the weight matrix of the encoder, $\boldsymbol{W'} = \boldsymbol{W^T}$. Training the autoencoder means finding the optimal values for $\boldsymbol{W}, \boldsymbol{b}$, and $\bar{\boldsymbol{b}}$ that minimize the cost function [35]. A deeper stacked autoencoder, as used in our experiments, can be built by adding more paired layers to the encoder and decoder sides.

V. EXPERIMENTAL RESULTS AND DISCUSSION

The two proposed approaches were implemented in Python. Our experiments were conducted on a machine with a Nvidia GTX 980 ti GPU.

In this section, we study the two proposed approaches performance on the three hand gesture datasets detailed in section III.

A. SINGLE 3DCNN STRUCTURE

Intensive experiments were conducted to evaluate the performance of the single 3DCNN-based structure in different modes. A fixed batch size of 16 samples was used in each experiment in this section.

1) EXPERIMENTS ON KSU-SSL DATASET

Signer-dependent mode: The KSU-SSL dataset consists of 40 classes and each class has 200 gesture samples. Therefore, the total number of samples is 8000. These 8000 samples were shuffled and 80% of them were randomly picked for training and the rest 20% were used for testing. The samples performed by a specific signer were thereby, divided into training and testing samples with random ratios.

The training datasets of 6400 samples were used to tune the model over 100 iterations. We randomly picked five percent of the training samples for validation after each iteration. Then, we used the testing dataset of 1600 samples to evaluate the trained model. A recognition rate of 96.69% was achieved by the model. Fig. 7 illustrates the training vs. validation loss achieved by the model during the model tuning iterations. We notice that the training and validation loss curves are close to each other during the training iterations, which is an indicator of good training behavior without overfitting.

Signer-independent mode: For this scenario, the model was trained on 6400 video samples and evaluated on 1600 video samples. The training samples were performed by 32 signers and the evaluation samples by the other eight signers. Except for this data separation, we repeated the configuration of the signer-dependent experiment. The behavior of the model loss

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2990434, IEEE Access

is illustrated in Fig. 8. The model achieved a recognition rate of 72.32%.

2) EXPERIMENTS ON ArSL DATASET

There are 3444 valid samples in the ArSL dataset, which were used in our experiments.

Signer-dependent mode: In this experiment, we randomly picked two-third of the samples in the dataset for tuning the system, and the remaining one third of the dataset samples for evaluation. Fig. 9 illustrates the system loss behavior on training and validation samples over 100 iterations. Five percent of the training 2298 samples were picked randomly for validation at each iteration. A recognition rate of 100% was achieved by the system.

Signer-independent mode: In this case, the training samples were performed by two signers and the testing samples were performed by the third signer. The system loss behavior during training is illustrated in Fig. 10.

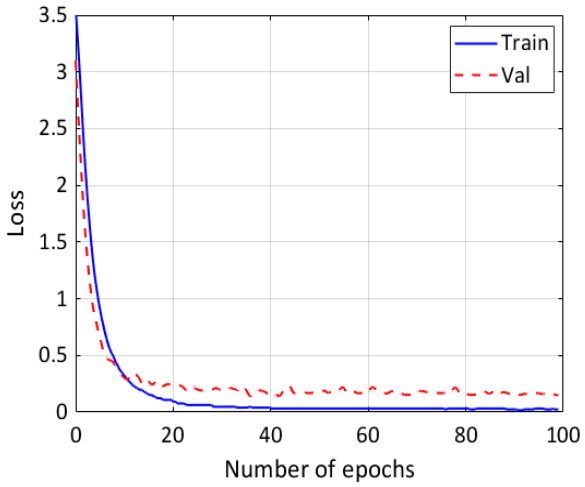

FIGURE 7. Single 3DCNN-based structure training loss on KSU-SSL in signer dependent mode.

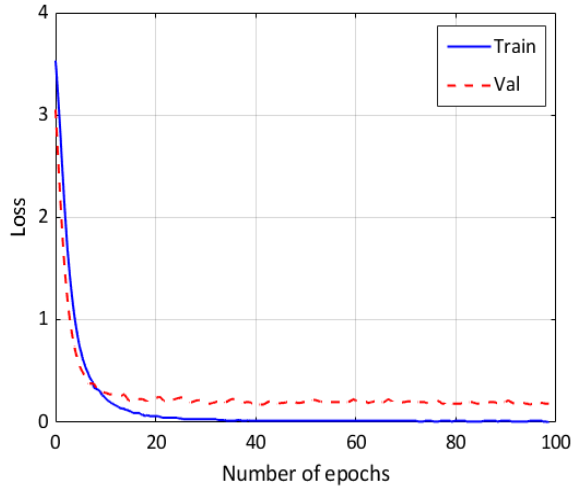

FIGURE 8. Single 3DCNN-based structure training loss on KSU-SSL in signer independent mode.

The system obtained a recognition rate of only 34.9% on the testing dataset which was expected in the signer-independent mode. Even though the system performance on the validation dataset as shown in Fig. 10 was excellent, it is clear that the system was overfitted on the training dataset. The samples were not diverse enough for system training. This is why it did not generalize well. Only the samples of two signers were used for tuning the system, while the samples of the third one were used for evaluation.

Conversely, on the KSU-SSL database better accuracy was achieved by the system for signer-independent scenario. In the KSU-SSL dataset, the training samples were performed by 32 signers and the testing samples were performed by eight signers.

This leads to better generalization. The effects of the dataset size in terms of the number of subjects, number of gestures, and number of repetitions appear in Figs. 7-10.

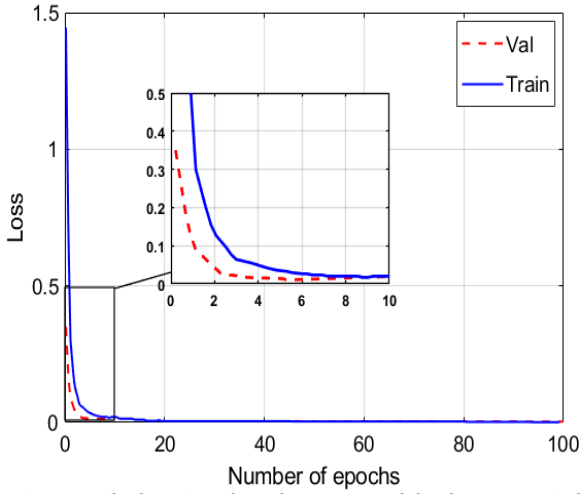

FIGURE 9. Single 3DCNN-based structure training loss on Ar-SL in signer dependent mode.

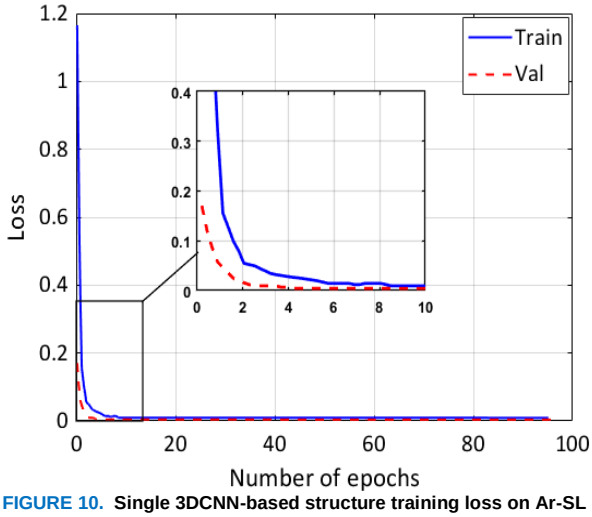

FIGURE 10. Single 3DCNN-based structure training loss on Ar-SL in signer independent mode.

The training and validation loss reached the minimum value within the first ten epochs in the case of the ArSL dataset and

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2990434, IEEE Access

the first 40 epochs in the case of the KSU-SSL dataset. The curves become stationary on that value in the case of the ArSL dataset, which indicates that the training and validation samples are highly overlapped.

Even though the validation samples are selected randomly, the possibility of overlapping between the training and validation samples is still high because the number of dataset samples is limited. For this reason, the training and validation loss curves are nearly coincident. The situation is completely different in the case of the KSU-SSL dataset. The random selection of validation samples leads to low overlapping between the training and validation samples. The loss curves, in this case, are not coincident, and their transition to the minimum value is smoother than that of the ArSL dataset.

3) EXPERIMENTS ON PURDUE RVL-SLLL ASL DATASET

A portion of only ten gestures from this dataset was involved in these experiments. These gestures were collected by 14 subjects. By testing the proposed approach on these gestures, we aimed to compare our approach with advanced methods evaluated on this dataset. As the method that we compared with used these ten gestures, we had to use the same gestures to make a fair comparison. As there are only 280 gesture samples, which are involved in this experiment. it is hard to generalize the trained model on such a small data. Therefore, we investigated the effect of data augmentation on the efficiency of our approach.

Signer-dependent mode: Before data augmentation, the model was trained on 80% of the samples and tested on the remaining 20%. The model achieved a recognition rate of 62.5%. The experiment was repeated after applying data augmentation to the 220 training samples. To achieve this, we performed the following two operations on the training dataset: (1) a rotation with four small angles (3˚,6˚,9˚, and 12˚), which produced four extra samples from every training sample. (2) Gaussian blurring with three different kernel sizes (3,5, and 7), which produced three additional samples from every training sample. As a result of this augmentation operations, we got a total of 1670 training samples.

The remaining 60 samples from the original dataset were used to test the trained system. The recognition accuracy increased from 62.5% to 76.67% by data augmentation. This improvement can be noticed in the system loss behavior shown in Fig. 11.

Signer-independent mode: Before data augmentation, we used 220 samples performed by 11 subjects for training the system. The remaining samples in the dataset, which were performed by the other three subjects, were used to evaluate the trained system. A recognition rate of 58.33% was achieved by the system. Then, on the training samples, we repeated the same data augmentation process described in the previous experiment and the recognition accuracy increased from 58.33% to 70%. This enhancement is illustrated in Fig. 12.

Fig. 13 summarizes the single 3DCNN structure performance on the three datasets in different modes. The performance in the signer-dependent mode is excellent on the KSU-SSL and ArSL datasets but still needs enhancement on RVL-SLLL-ASL and in the signer-independent mode. A preliminary results was also discussed in [37].

B. FUSION OF PARALLEL 3DCNN STRUCTURE

We realized that the ArSL and Purdue RVL-SLLL ASL datasets are not comprehensive in terms of the number of samples, subjects, and video lengths. For instance, we cannot investigate the fusion of parallel 3DCNN structure on short video samples. This is why we chose only the KSU-SSL dataset for further experiments related to the fusion of parallel 3DCNNs. In the preprocessing phase, 32 frames were linearly sampled from each video instance. The signing space was then cropped as detailed in the first approach given in section A.

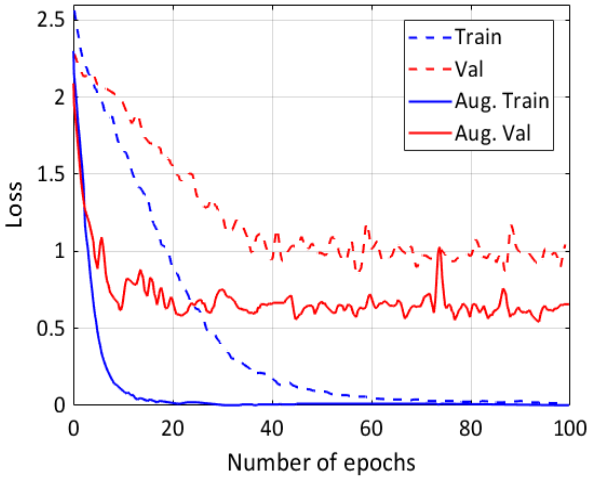

FIGURE 11. Single 3DCNN-based structure training loss on RVL-SLLL- ASL dataset in signer-dependent mode.

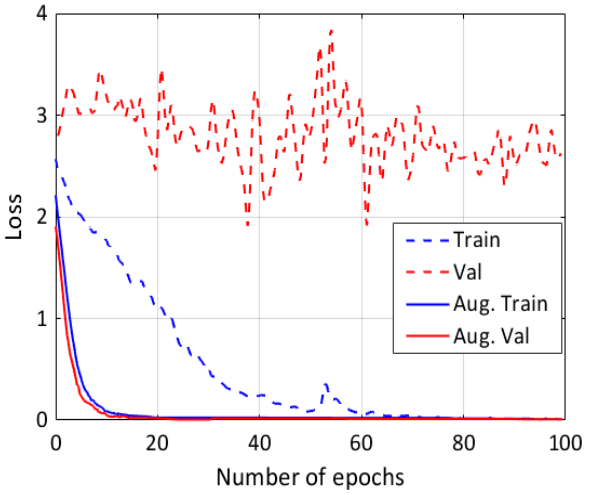

FIGURE 12. Single 3DCNN-based structure training loss on RVL-SLLL- ASL dataset in signer-dependent mode.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2990434, IEEE Access

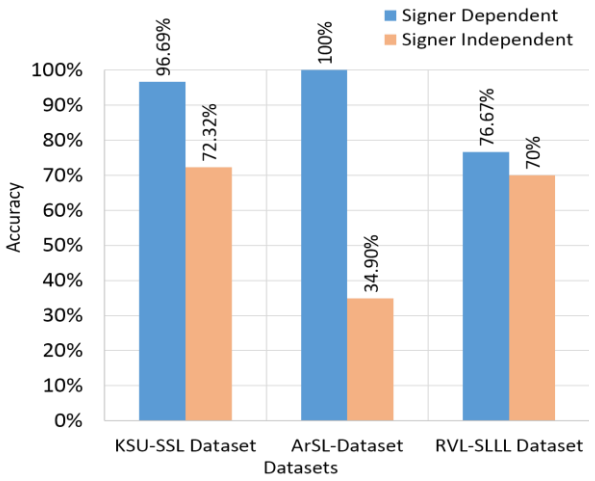

FIGURE 13. Single 3DCNN-based structure performance on three datasets.

- ***Feature learning***

Signer-dependent mode:

After the preprocessing step, each gesture sample had a fixed size of 112 × 112 × 32 × 3. These numbers determine the height and width of the frames, the number of frames per video, and the number of channels per frame. The dataset samples were shuffled before randomly splitting them into training and testing samples of 80% and 20%, respectively. A sliding window of 16 frames in width was then used to divide each video sample into three clips each 16 frames in length with eight frames overlapping.

Each of the three clips in the training samples was used to refine one of the 3DCNN instances.

The setup used for the single 3DCNN structure was also applied for training each 3DCNN instance. The trained 3DCNN instances were then used to extract the features from the corresponding clips in the training dataset. This feature learning step represents each sample in the training dataset by three feature vectors of 4096 dimensions each. The same feature representation was performed on the testing dataset.

Signer-independent mode:

As detailed for the single 3DCNN approach given in section A, the training and testing samples use separate groups of signers. Except for this data separation step, we repeated the procedure as in the signer-dependent mode.

- ***Feature Fusion***

Before being used by the classifier the three feature vectors of each sample were fused using three different techniques. We performed end-to-end training for the fusion network with the classification layer. The SGD optimizer with an adaptive learning rate (initial learning rate = 10^{-4}, decay = 10^{-6}, and momentum = 0.9) was used to optimize the negative log-likelihood cost over 100 iterations. This a small value of momentum will avoid falling in the local minima of the cost.

1) MLP FUSION

In this section, we utilized MLP networks for feature fusion. We also studied the effect of the MLP depth (number of layers), number of neurons in each layer, and training batch size on the system's performance. We conducted the first experiments on a single layer of 8192 neurons to select the initial learning rate from common values in the literature. The results in Table III show that 10^{-4} achieved the best performance. This value was then fixed in the rest of the experiments. Intensive experiments were then conducted with batch sizes of 16, 32, and 64 samples and three different MLP architectures. The first architecture consisted of a single layer of 8192 neurons. We added another layer of 4096 neurons to the second architecture. The third architecture also had another layer of 4096 neurons added. The best recognition accuracies of 98.12% and 84.38%, respectively, were achieved in the signer-dependent and signer-independent modes. The best accuracy in both cases was achieved by the third architecture with a batch size of 16 samples. Figs. 14 and 15 illustrate the confusion matrices for the best architectures.

2) LSTM FUSION

In this section, we used different LSTM architectures for feature fusion. We also studied the effects of batch size on the system performance. Intensive experiments were conducted with batch sizes of 16, 32, and 64 samples. Three different LSTM architectures were investigated.

The first architecture consisted of a single layer of 4096 memory cells (units). The second architecture consisted of two stacked layers with 4096 and 1024 memory cells, respectively. The third architecture consisted of three stacked layers of 4096, 1024, and 1024 memory cells, respectively. Backpropagation through time was used for end-to-end training of the LSTM layers with the classification layer. The best recognition accuracies of 97.94% and 82.19% were achieved in the signer-dependent and signer-independent modes, respectively. This best accuracy was achieved by the second architecture with a batch size of 16 samples. Fig. 16 and Fig. 17 illustrate the confusion matrices for the best architectures.

3) AUTOENCODER FUSION

In this section, we used a stacked autoencoder architecture for feature fusion. The autoencoder consisted of an extra pair of hidden layers in addition to the latent, input, and output layers, as illustrated in Fig. 18. The size of the input/output layer was 12288. The sizes of the hidden and the latent layers were 4096 and 2048, respectively. We noted that the best accuracy was achieved with the smallest batch size in the previous two fusion architectures. We, therefore, fitted the autoencoder architecture with a batch size of 16 samples.

TABLE III
ACCURACY (%) ACHIEVED BY A SINGLE LAYER WITH DIFFERENT INITIAL LEARNING RATE

Initial learning rate	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}
Signer dependent	4.94	97.06	**97.75**	97.41	97.0
Signer independent	5.0	83.81	**83.94**	83.38	82.94

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2990434, IEEE Access

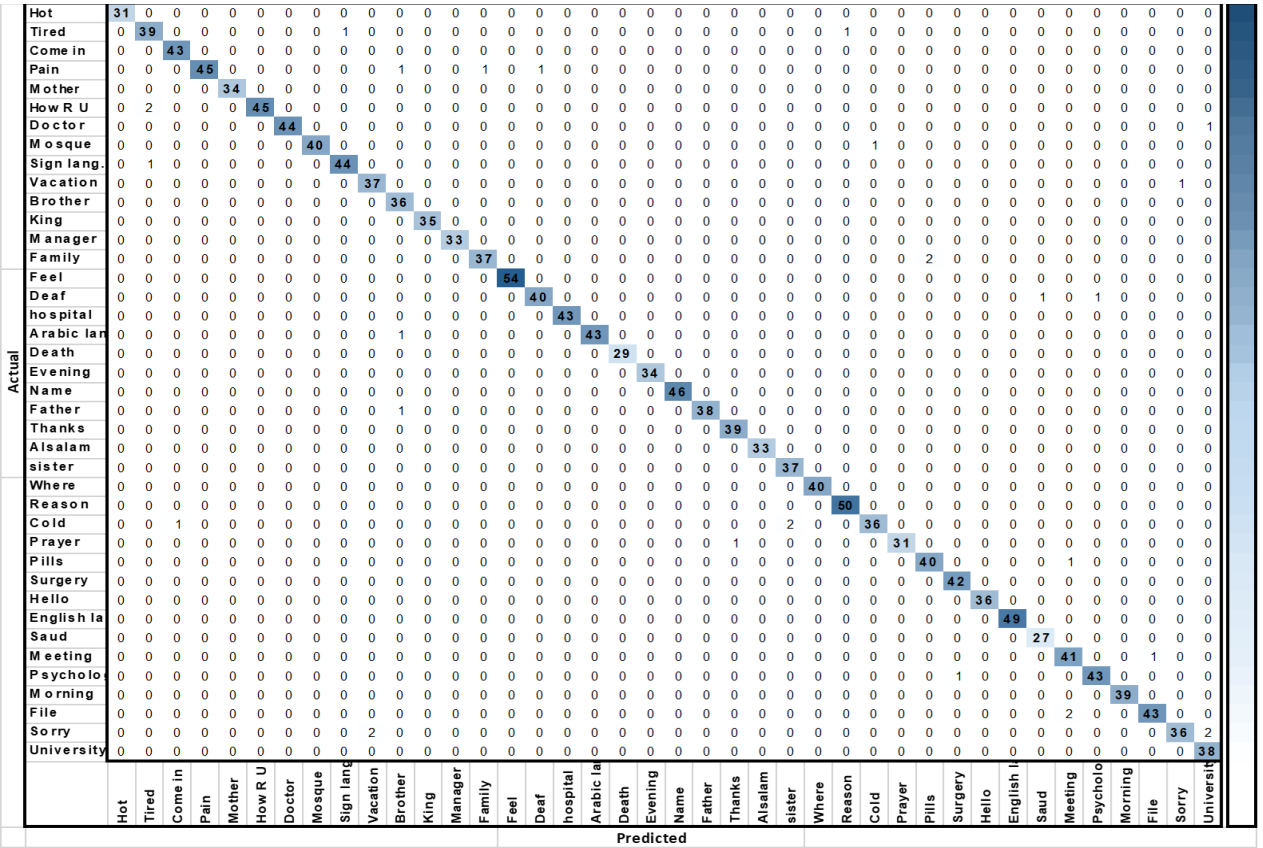

FIGURE 14. Confusion matrix of MLP fusion for parallel structure in signer-dependent mode on KSU-SSL dataset.

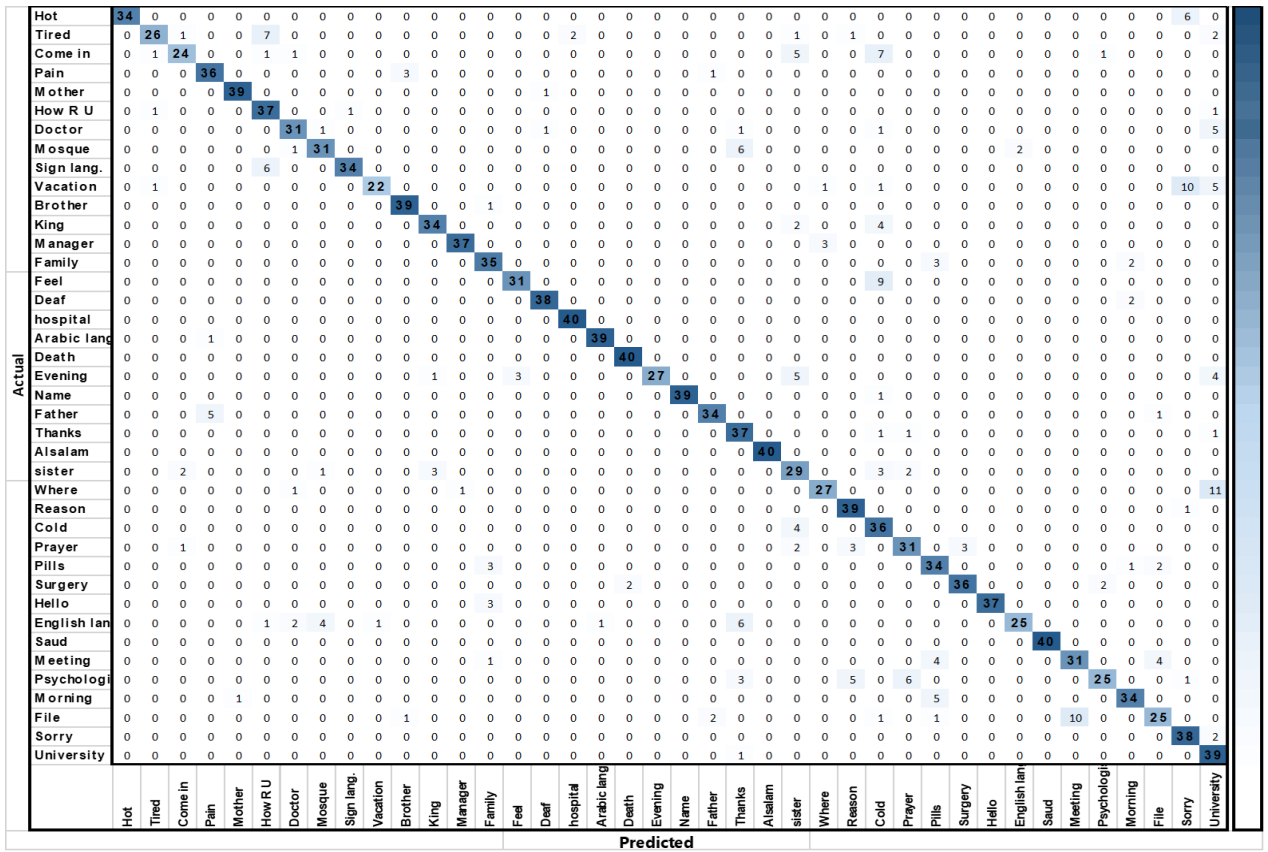

FIGURE 15. Confusion matrix of MLP fusion for parallel structure in signer-independent mode on KSU-SSL dataset.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2990434, IEEE Access

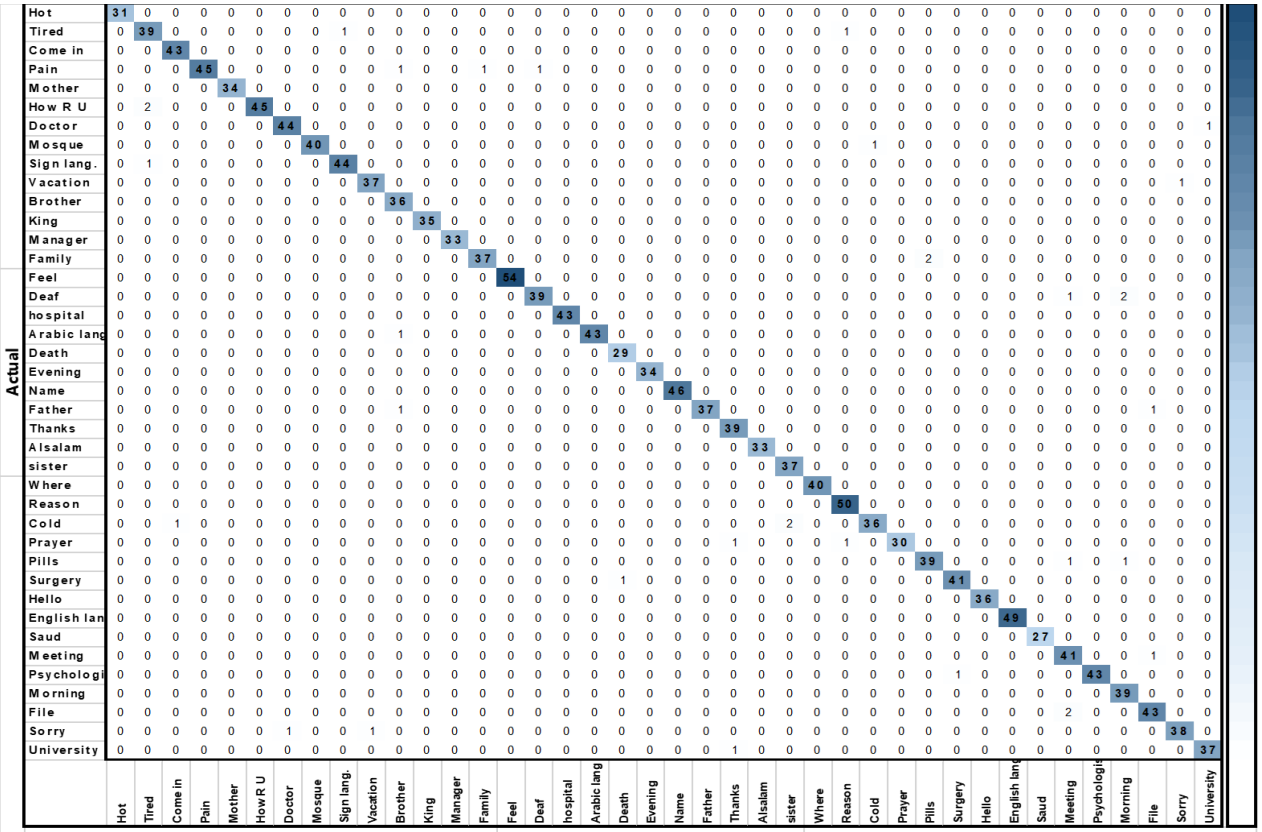

FIGURE 16. Confusion matrix of LSTM fusion for parallel structure in signer-dependent mode on KSU-SSL dataset.

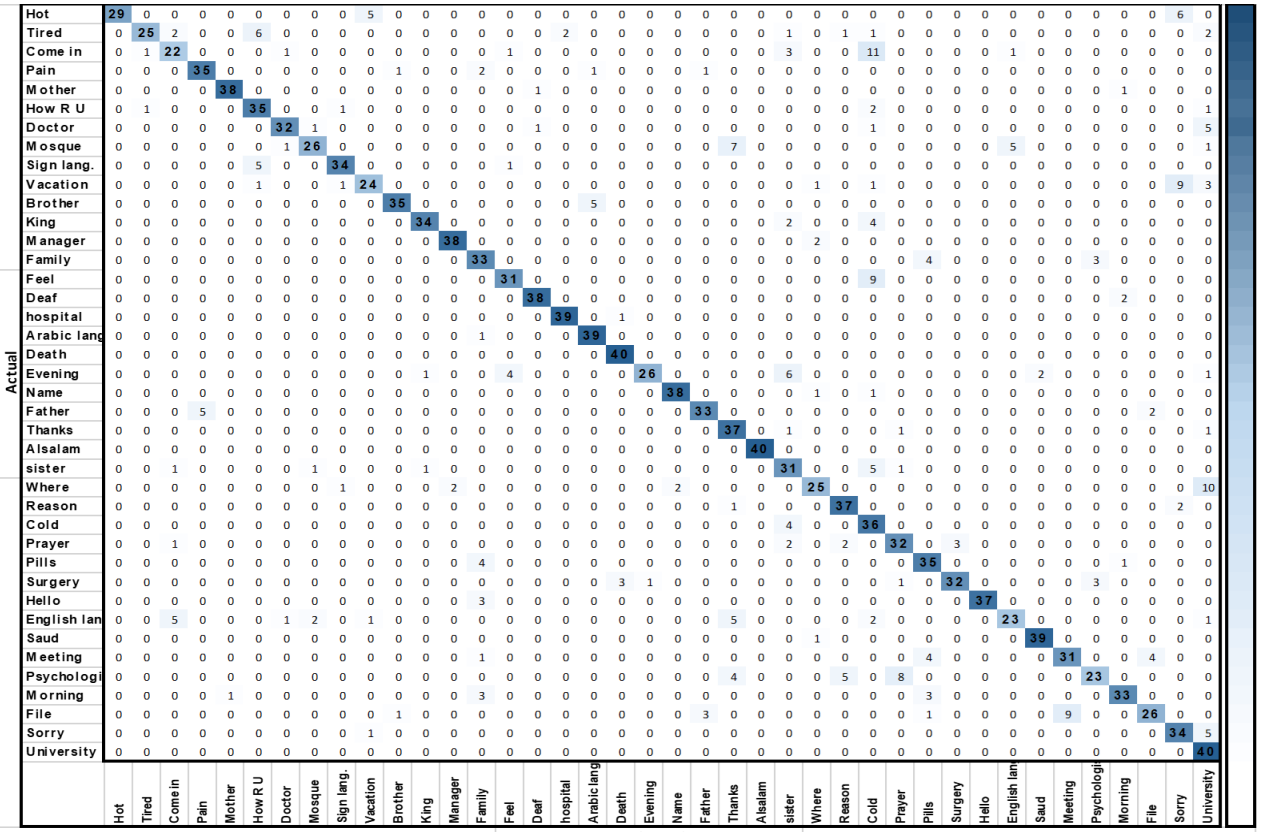

FIGURE 17. Confusion matrix of LSTM fusion for parallel structure in signer-independent mode on KSU-SSL dataset.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2990434, IEEE Access

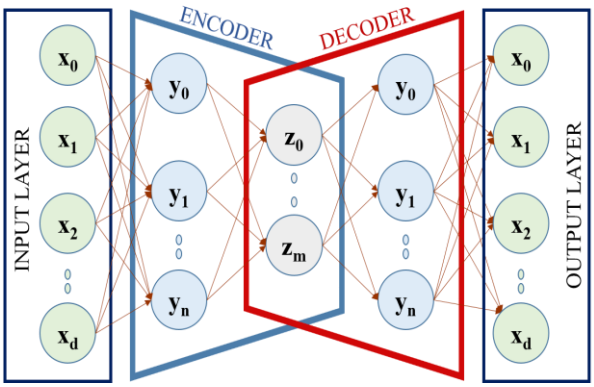

FIGURE 18. Stacked autoencoder of five layers.

The final features were extracted from the latent layer of the trained autoencoder. We achieved an accuracy of 90.88% in signer-dependent mode and 62.44% in signer-independent mode. Fig. 19 and Fig. 20 illustrate the confusion matrices for the autoencoder architecture evaluation.

Finally, the configuration of the best scenario was also tested with other values of the initial learning rate to ensure that the selection of the 10^{-4} value was still valid for that architecture. The result showed that the value of 10^{-4} still achieves the best performance as shown in Table IV.

TABLE IV
ACCURACY (%) ACHIEVED BY THE BEST CONFIGURATION WITH DIFFERENT INITIAL LEARNING RATE

Initial learning rate	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}
Signer dependent	2.44	97.69	**98.12**	98.06	97.88
Signer independent	2.5	81.69	**84.38**	83.25	83.19

C. RESULTS DISCUSSION AND COMPARISON

Table V summarizes the accuracy achieved by the MLP and LSTM architectures. We note that the MLP architecture achieved better recognition accuracy in all scenarios. Even though there is no clear trend change in accuracy despite the number of layers, the smallest batch size still achieves the best accuracy. This might be attributed to the fact that a smaller batch size means that the model parameters were updated more frequently. However, this kind of update based on a small number of noisy samples adds a regularizing effect and results in a lower generalization error.

In general, the parallel 3DCNN-based architecture features are superior to those of the single 3DCNN-based architecture. This superiority can be noticed in the accuracy achieved in the signer-independent mode. The single 3DCNN architecture achieved an accuracy of 72.32%. The accuracy increased to 82.19% when using the parallel 3DCNN with the LSTM architecture and 84.38% with the MLP architecture.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2990434, IEEE Access

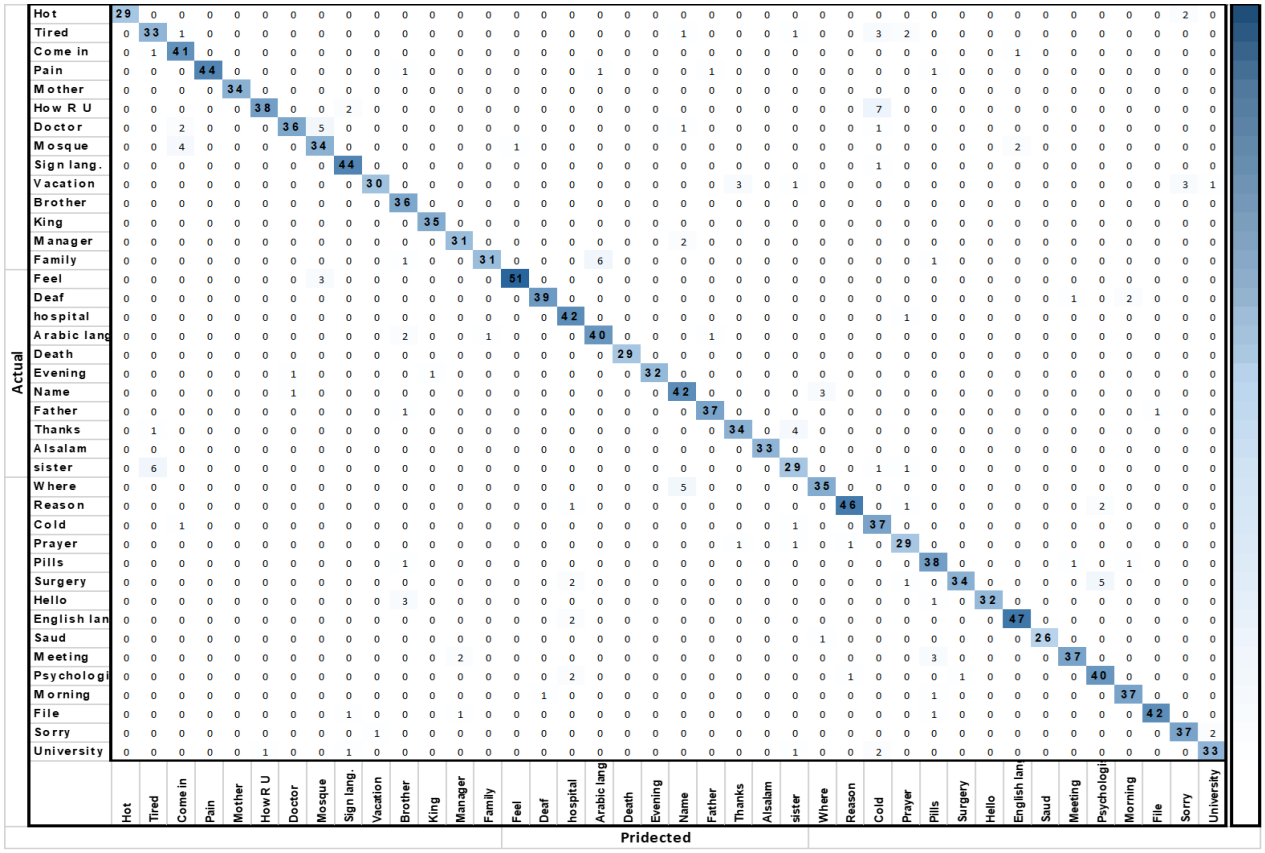

FIGURE 19. Confusion matrix of autoencoder fusion for parallel structure in signer-dependent mode on KSU-SSL dataset.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2990434, IEEE Access

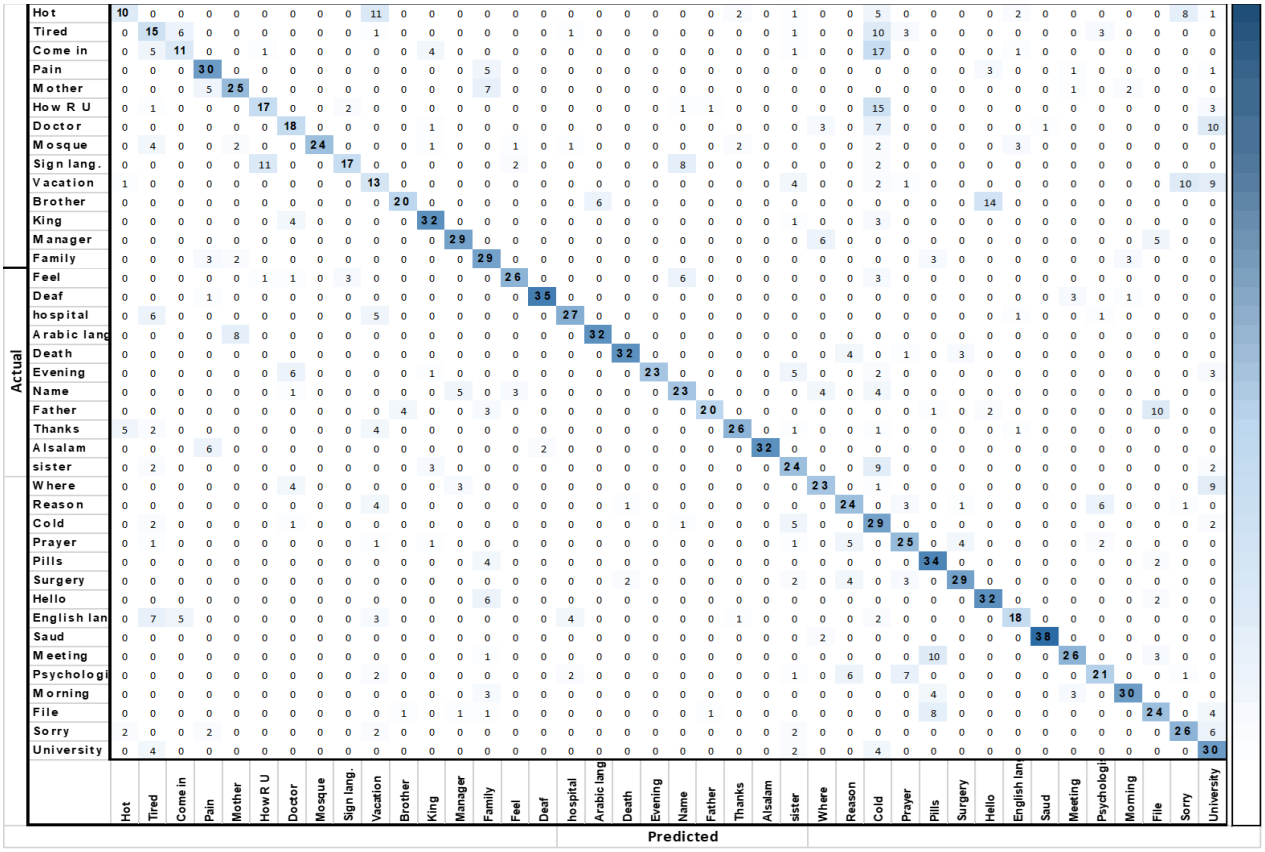

FIGURE 20. Confusion matrix of autoencoder fusion for parallel structure in signer-independent mode on KSU-SSL dataset.

This enhancement in recognition accuracy is objective proof of the superiority mentioned above.

Furthermore, we utilized the principal component analysis (PCA) and t-distributed stochastic neighbor embedding (t-SNE) techniques to reduce and visualize the learned features of the two approaches.

PCA and t-SNE are two popular techniques for data reduction. PCA uses the correlation between some dimensions and provides minimum number of variables while maintaining the maximum information about the distribution of original data. To achieve that, it performs mathematical calculations for the eigenvalues and eigenvectors of the data-matrix. These eigenvectors of the covariance matrix have the property that they point along the major directions of variation in the data. These are the directions of maximum variation in the dataset [38]. t-SNE on the other hand, is a probabilistic technique rather than mathematical technique. It is convenient for high-dimensional data reduction and visualization. It tries to represent the data of high dimensionality in a lower dimensional space, while minimizing the divergence between the distributions of the data in the two spaces. The main advantage of the t-SNE over the PCA is the preservation of the neighborhood structure of the original data. Unfortunately, t-SNE performs heavy computations to achieve this representation, therefore t-SNE use is limited.

TABLE V
ACCURACY ACHIEVED BY MLP AND LSTM FEATURE FUSION IN DIFFERENT SCENARIOS

	Batch size	1 layer	2 layers	3 layers
		MLP Fusion Accuracy (%)		
Signer dependent	16	97.75	98.06	**98.12**
	32	97.75	97.94	98.06
	64	97.75	97.94	98.06
Signer independent	16	83.94	83.88	**84.38**
	32	84.00	84.31	84.25
	64	83.94	84.06	84.06
		LSTM Fusion Accuracy (%)		
Signer dependent	16	97.69	**97.94**	96.12
	32	97.75	97.81	95.94
	64	97.69	97.75	95.38
Signer independent	16	81.25	**82.19**	79.31
	32	81.25	82.12	77.88
	64	81.06	81.94	77.94
Autoencoder				
Signer dependent	16	90.88		
Signer independent		62.44		

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2990434, IEEE Access

For instance, applying another dimensionality reduction technique before using t-SNE is needed in case of very high dimensional data [39]. In our case the feature vector in the first approach had a dimension of 4096, while in the second approach it had a dimension of 12288.

For that reason, we started by performing PCA reduction to intermediate space of 50 dimensions by using 50 principal components. Then, t-SNE was used to project and visualize the reduced features in a 2D space, as depicted in Fig. 21. The left side in Fig. 21 shows the features learned by the single 3DCNN structure and the right side shows the features learned by the parallel 3DCNN structure. This figure provides subjective proof of the superiority of the parallel 3DCNN fused features.

It is clear from the figure that the features learned by the parallel 3DCNN architecture are more discriminative than those learned by the single 3DCNN architecture. Unfortunately, the autoencoder achieved the lowest accuracy even on these more discriminative features. The autoencoder regularization is dedicated to maintaining strong traces to allow reconstruction of the reduced data dimensions to the original dimensions. These traces might not be efficient discriminators for classification.

In the confusion matrices, we noticed that the performance of our system in the signer-independent mode is weak compared to the performance in the signer-dependent mode. There is significant variation when the gesture is performed by a large number of signers. This high variation leads to a low recognition rate when the system is tested on samples performed by some of the signers while none of the samples performed by these signers were involved in the model training.

TABLE VI
RECOGNITION ACCURACY (%) COMPARED WITH OTHER METHODS

Methods	ArSL		RVL-SLLL		KSU-SSL	
	Signer Dep.	Signer Indep.	Signer Dep.	Signer Indep.	Signer Dep.	Signer Indep.
[20] (DCT, HMM)	94					
[19] (DCT, K-NN)	100					
[21] (LBP-TOP, SVM)	99.5					
[15] (BHOF, distance)	--	--	--	60		
[23] (RGB + Flow, inception)	--	--	--	--	98.25	84.22
[26] DenseImage Net.					78.12	59.35
[26] DenseImage Net with gesture localization					89.52	70.3
Single 3DCNN	**100**	34.9	76.67	**70**	**96.69**	**72.32**
Parallel 3DCNN	--	--	--	--	**98.12**	**84.38**

We also noticed that there is a high level of confusion between some classes, especially in the signer-independent mode. We analyzed the confusion matrices of the three architectures to investigate the most confused classes. More focus was given to cases in which two gestures exhibited high confusion in all architectures in both signer-dependent mode and signer-independent mode. We found that the confused gestures had almost the same global signers' body configurations and very closed relative hand positions and orientations in the sampled frames.

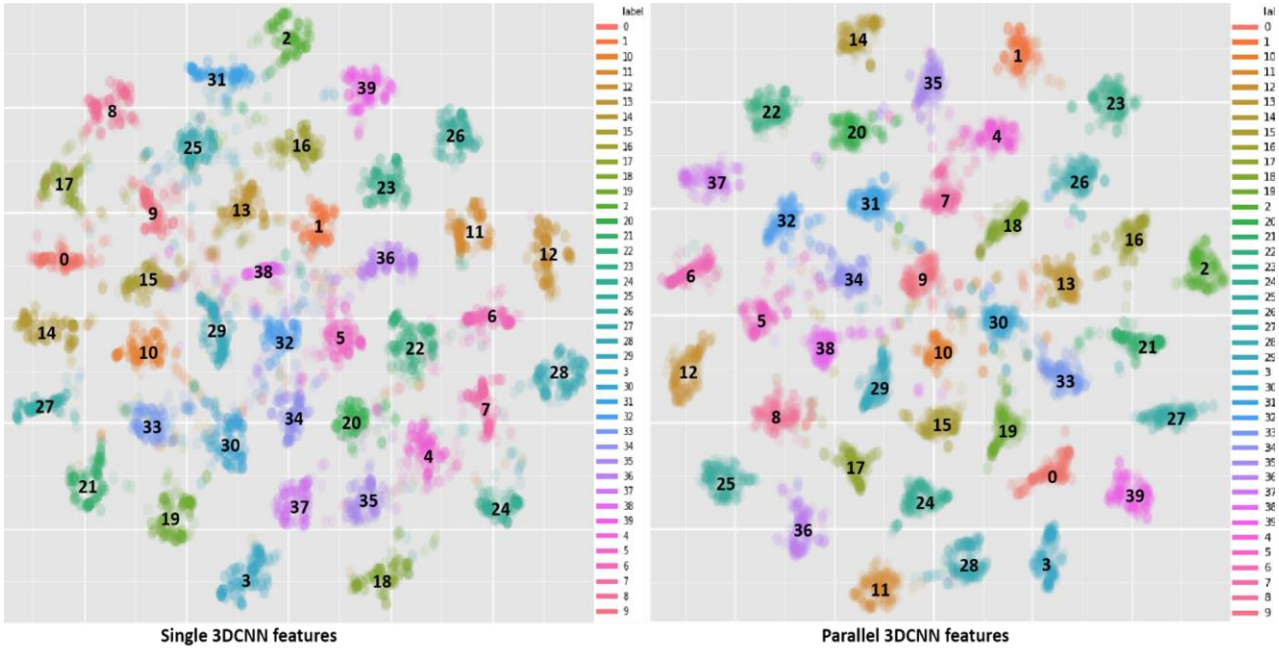

FIGURE 21. Feature visualization for single 3DCNN and parallel 3DCNN architectures.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2990434, IEEE Access

FIGURE 22. Sample frames from four pairs of confusing gestures. This four sets of sequences show the confusion between the following pairs:

(a): gestures "Cold" and "Come in"
(b): gestures "File" and "Meeting,"
(c): gestures "sorry" and "Vacation"
(d): gestures "Cold" and "How are you?"

We can only differentiate between these gestures from the finger configurations. Unfortunately, the small fingers' area received insufficient focus and lighting during the KSU-SSL dataset recording. Furthermore, the hand regions were blurred as the frame rate of the recording cameras was not high enough to avoid motion blur. All of these factors may lead to such misclassification. Samples of the confusing gestures are illustrated in Fig. 22. In this figure, we notice four pairs of confusing gestures: "Cold" with "Come in," "File" with "Meeting," "Sorry" with "Vacation," and "Cold" with "How are you?" The high correlation between the sequences of frames in each pair is very clear, which is another challenge with the KSU-SSL dataset.

In Table VI, we compare the performance of the proposed approaches on the three datasets with some state-of-the-art methods in the literature. Most of the state-of-the-art methods in the literature of hand gesture recognition are based on multimodal input. Various channels such as depth maps and skeletal joints are utilized with the RGB frames in these methods. To make a fair comparison, we only considered the RGB based methods, which are similar to our proposed approach. Unfortunately, The RGB based methods are very limited in the literature.

The selected methods used different techniques to encode the spatiotemporal features of the gestures. Four of these approaches utilized handcrafted features such as combining the accumulated distance between successive frames and DCT or LBP-TOP with conventional classifiers such as SVM, HMM, K-NN, and Bayesian classifiers.

The most recent two approaches [23][26] utilized deep CNN models with the RGB frames. The approach in [23] generates the optical flow information from the RGB frames and feeds them into the inception model together. This approach performed better than our single 3DCNN approach on the KSU-SSL dataset. However, our parallel 3DCNN approach with MLP fusion achieved comparable performance, even though the optical flow generation step makes the computation cost of the approach in [23] higher. In this regard, it was

impossible to execute that approach on the GTX 980 Ti with 6 GB of RAM to make a fair execution-time comparison. We had to execute it on GTX 1080 Ti with 11 GB of RAM. On the other hand, the DenseImage approach [26], encoded each video in a 2D matrix by applying the ResNet deep model followed by another CNN model to learn the encoded video features. On the GTX 980 Ti machine, the DenseImage net architecture took approximately 195 hours for 1000 training iterations with the setup mentioned by the authors [26]. On the other hand, our proposed parallel 3DCNN architecture took approximately 25 hours on the same machine. This lower training time was expected as we froze most of the layers in the pre-trained model. The low accuracy of the DenseImage net architecture might be attributed to the irrelevant features extracted from each frame. The application of ResNet on the entire frame gives the relevant features in the gesture space and the irrelevant ones outside that space the same weight, which increases the misclassification rate. Our approach avoided this issue by employing the face detection and body parts ratios information to involve only the relevant space and exclude most of the irrelevant regions in each frame.

We also evaluated the performance of the DenseImage Net [26] on the dataset after applying our proposed preprocessing method to localize the gesture space to see the effect of this step on the performance. The results showed that there is an excellent enhancement in the accuracy, however, it is still outperformed by the two proposed approaches. We think that, some of the temporal information might be lost by encoding the video in a 2D matrix. The 2DCNN applied on that matrix after that did not consider the temporal information of the gesture.

VII. CONCLUSION

This study investigates the use of 3DCNN for hand gesture recognition. In the preprocessing phase, linear sampling was used to normalize the temporal dimension of hand gesture samples. For spatial dimension normalization, we used the length of the detected face and human body part ratios. Then, we used 3DCNN for feature learning in two approaches. In the first approach, a single 3DCNN instance was trained to extract the hand gesture features from the entire video. In the second approach, three instances of the 3DCNN structure were trained to extract the hand gesture features from the beginning, middle, and end of the video sample. These region-based features were then fused before being fed to the classifier. MLP, LSTM, and an autoencoder were used for feature fusion. In both approaches, we used a SoftMax activated layer for classification. The proposed approaches were evaluated using different datasets. The three datasets exhibited excellent performance in both signer-dependent and signer-independent modes. The proposed approaches were compared with six other state-of-the-art methods from the literature. They outperformed four of these methods and showed comparable performance to the other two.

For future work, we will enhance the performance of the proposed approach by performing a holistic search to optimize all the hyperparameters. We will test the proposed approach online while receiving a live video feed. In this aspect, we may utilize the edge-cloud computing to distribute the processing over edge devices and the core cloud [40][41].

REFERENCES

[1] S. Kausar and M. Y. Javed, "A Survey on Sign Language Recognition," in *IEEE Frontiers of Information Technology*, Islamabad, Dec. 2011.

[2] G. Muhammad, M. F. Alhamid, and X. Long, "Computing and Processing on the Edge: Smart Pathology Detection for Connected Healthcare," IEEE Network, vol. 33, issue 6, pp. 44-49, November-December 2019.

[3] G. Muhammad, M. S. Hossain, and A. Yassine, "Tree-Based Deep Networks for Edge Devices," IEEE Transactions on Industrial Informatics, vol. 16, no. 3, pp. 2022-2028, March 2020.

[4] J. Zhao, X. Mao, and L. Chen, "Speech emotion recognition using deep 1D & 2D CNN LSTM networks," Biomed. Signal Process. Control, vol. 47, pp. 312–323, Jan. 2019.

[5] E. Ijjina and K. Chalavadi, "Human action recognition using genetic algorithms and convolutional neural networks," Pattern Recognit., vol. 59, pp. 199–212, Nov. 2016.

[6] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-Scale Video Classification with Convolutional Neural Networks," in *IEEE Conference on Computer Vision and Pattern Recognition*, Ohio, Jun. 2014.

[7] R. Hou, C. Chen, and M. Shah, "An end-to-end 3D convolutional neural network for action detection and segmentation in videos," *arXiv Prepr. arXiv:1712.01111*, Nov. 2017.

[8] Z. Xu, C. Xiang, L. Yun, V. Lantz, W. Kongqiao, and Y. Jihai, "A Framework for Hand Gesture Recognition Based on Accelerometer and EMG Sensors," *IEEE Trans. Syst. Man, Cybern. - Part A Syst. Humans*, vol. 41, no. 6, pp. 1064–1076, Nov. 2011.

[9] V. Kosmidou and L. Hadjileontiadis, "Sign Language Recognition Using Intrinsic-Mode Sample Entropy on sEMG and Accelerometer Data," *IEEE Trans. Biomed. Eng.*, vol. 56, no. 12, pp. 2879–2890, Dec. 2009.

[10] T. Bui and L. T. Nguyen, "Recognizing Postures in Vietnamese Sign Language with MEMS Accelerometers," *IEEE Sens. J.*, vol. 7, no. 5, pp. 707–712, May 2007.

[11] G. Fang, W. Gao, and D. Zhao, "Large-Vocabulary Continuous Sign Language Recognition Based on Transition-Movement Models," *IEEE Trans. Syst. Man, Cybern. - Part A Syst. Humans*, vol. 37, no. 1, pp. 1–9, Jan. 2007.

[12] U. Cote-Allard *et al.*, "Deep Learning for Electromyographic Hand Gesture Signal Classification Using Transfer Learning," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 27, no. 4, pp. 760–771, Apr. 2019.

[13] K. Murakami and H. Taguchi, "Gesture recognition using recurrent neural networks," in *Proceedings of the SIGCHI conference on Human factors in computing systems Reaching through technology - CHI '91*, 1991, pp. 237–242, doi: 10.1145/108844.108900.

[14] M.Yang, N. Ahuja, and M. Tabb, "Extraction of 2D motion trajectories and its application to hand gesture recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 8, pp. 1061–1074, Aug. 2002.

[15] K. Lim, A. Tan, and S. Tan, "Block-based histogram of optical flow for isolated sign language recognition," *J. Vis. Commun. Image Represent.*, vol. 40, pp. 538–545, Oct. 2016.

[16] M. Zahedi, D. Keysers, T. Deselaers, and H. Ney, "Combination of Tangent Distance and an Image Distortion Model for Appearance-Based Sign Language Recognition," in *Pattern Recognition*, Springer, Berlin, Heidelberg, 2005, pp. 401–408.

[17] R. Wilbur and A. Kak, "Purdue RVL-SLLL American Sign Language Database," *School of Electrical and Computer Engineering Technical Report, TR-06-12, Purdue University, W. Lafayette, IN 47906*. 2006.

[18] V. Athitsos *et al.*, "The American Sign Language Lexicon Video Dataset," in *2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, Alaska, Jun. 2008.

[19] T. Shanableh, K. Assaleh, and M. Al-Rousan, "Spatio-Temporal Feature-Extraction Techniques for Isolated Gesture Recognition in Arabic Sign Language," *IEEE Trans. Syst. Man Cybern. Part B*, vol. 37, no. 3, pp. 641–650, Jun. 2007.

[20] K. Assaleh, T. Shanableh, M. Fanaswala, F. Amin, and H. Bajaj, "Continuous Arabic Sign Language Recognition in User Dependent Mode," *J. Intell. Learn. Syst. Appl.*, vol. 02, no. 01, pp. 19–27, Mar. 2010.

[21] S. Aly and S. Mohammed, "Arabic Sign Language Recognition Using Spatio-Temporal Local Binary Patterns and Support Vector Machine," in *International Conference on Advanced Machine Learning Technologies and Applications*, Cairo, Nov. 2014.

[22] M. Abid, E. Petriu, and E. Amjadian, "Dynamic Sign Language Recognition for Smart Home Interactive Application Using Stochastic Linear Formal Grammar," *IEEE Trans. Instrum. Meas.*, vol. 64, no. 3, pp. 596–605, Mar. 2015.

[23] O. Kopuklu, N. Kose, and G. Rigoll, "Motion Fused Frames: Data Level Fusion Strategy for Hand Gesture Recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Salt Lake City, Jun. 2018.

[24] P. Molchanov, S. Gupta, K. Kim, and J. Kautz, "Hand Gesture Recognition With 3D Convolutional Neural Networks," in *IEEE Conference on Computer Vision and Pattern Recognition*, Massachusetts, Jun. 2015.

[25] G. Poon, K. C. Kwan, and W.-M. Pang, "Occlusion-robust bimanual gesture recognition by fusing multi-views," *Multimed. Tools Appl.*, vol. 78, no. 16, pp. 23469–23488, Aug. 2019.

[26] X. Chen and K. Gao, "DenseImage Network: Video Spatial-Temporal Evolution Encoding and Understanding," *arXiv Prepr. arXiv:1805.07550*, 2018.

[27] Y. Kim and W. Yoon, "Generating Task-Oriented Interactions of Service Robots," *IEEE Trans. Syst. Man, Cybern. Syst.*, vol. 44, no. 8, pp. 981–994, Aug. 2014.

[28] K. Assaleh and M. Al-Rousan, "Recognition of Arabic Sign Language Alphabet Using Polynomial Classifiers," *EURASIP J. Adv. Signal Process.*, vol. 2005, no. 13, pp. 2136–2145, Dec. 2005.

[29] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Kauai, Jun. 2001.

[30] A. Özaslan, M. Y. İşcan, İ. Özaslan, H. Tuğcu, and S. Koç, "Estimation of stature from body parts," *Forensic Sci. Int.*, vol. 132, no. 1, pp. 40–45, Mar. 2003.

[31] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning Spatiotemporal Features with 3D Convolutional Networks," in *IEEE International Conference on Computer Vision*, Santiago, Dec. 2015.

[32] G. Dahl, T. Sainath, and G. Hinton, "Improving deep neural networks for LVCSR using rectified linear units and dropout," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, Vancouver, May 2013.

[33] I. Jindal, M. Nokleby, and X. Chen, "Learning Deep Networks from Noisy Labels with Dropout Regularization," in *IEEE 16th International Conference on Data Mining*, Barcelona, Dec. 2016.

[34] F. Ordóñez, D. Roggen, F. J. Ordóñez, and D. Roggen, "Deep Convolutional and LSTM Recurrent Neural Networks for Multimodal Wearable Activity Recognition," *Sensors*, vol. 16, no. 1, 115, Jan. 2016.

[35] Y. J. Fan, "Autoencoder node saliency: Selecting relevant latent representations," *Pattern Recognit.*, vol. 88, pp. 643–653, Apr. 2019.

[36] I. Goodfellow, Y. Bengio, and A. Courville, "Autoencoders," in *Deep Learning*, 1st ed., Massachusetts, USA: MIT Press, 2016, pp. 502–525.

[37] M. Al-Hammadi, G. Muhammad, W. Abdul, M. Alsulaiman, and M. S. Hossain, "Hand Gesture Recognition Using 3D-CNN Model," IEEE Consumer Electronics Magazine, vol. 9, no. 1, pp. 95-101, January 2020.

[38] S. Wold, K. Esbensen, and P. Geladi, "Principal component analysis," *Chemom. Intell. Lab. Syst.*, vol. 2, no. 1–3, pp. 37–52, Aug. 1987.

[39] L. Maaten and G. Hinton, "Visualizing Data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, pp. 2579–2605, Nov. 2008.

[40] G. Muhammad, M. F. Alhamid, M. Alsulaiman, and B. Gupta, "Edge Computing with Cloud for Voice Disorders Assessment and Treatment," *IEEE Communications Magazine*, vol. 56, issue 4, pp. 60-65, April 2018.

[41] Z. Ali, G. Muhammad, and M.F. Alhamid, "An Automatic Health Monitoring System for Patients Suffering from Voice Complications in Smart Cities," *IEEE Access*, vol. 5, no. 1, pp. 3900-3908, 2017.