Hindawi
Journal of Electrical and Computer Engineering
Volume 2019, Article ID 4167890, 12 pages
https://doi.org/10.1155/2019/4167890

Research Article

Static Hand Gesture Recognition Based on Convolutional Neural Networks

Raimundo F. Pinto Jr., Carlos D. B. Borges, Antônio M. A. Almeida, and Iális C. Paula Jr.

Universidade Federal do Ceará, Sobral, Ceará 62010-560, Brazil

Correspondence should be addressed to Raimundo F. Pinto Jr.; raimundo@pm.me

Received 27 December 2018; Revised 4 April 2019; Accepted 5 September 2019; Published 10 October 2019

Academic Editor: Sos S. Agaian

This paper proposes a gesture recognition method using convolutional neural networks. The procedure involves the application of morphological filters, contour generation, polygonal approximation, and segmentation during preprocessing, in which they contribute to a better feature extraction. Training and testing are performed with different convolutional neural networks, compared with architectures known in the literature and with other known methodologies. All calculated metrics and convergence graphs obtained during training are analyzed and discussed to validate the robustness of the proposed method.

1. Introduction

Humans are able to recognize body and sign language easily. This is possible due to the combination of vision and synaptic interactions that were formed along brain development [1]. In order to replicate this skill in computers, some problems need to be solved: how to separate objects of interest in images and which image capture technology and classification technique are more appropriate, among others.

The evolution of computing and the ease of access of new technologies motivated the development of equipment such as Kinect and Leap Motion, which are examples of innovation in input device technologies [2–4]. In this way, these devices are capable of capturing human gestures, developing a new medium of human-machine interaction. The uses of these devices are present in the most diverse areas, such as robotics, medicine, sign language translation, computer graphics, and augmented reality [5].

Gesture recognition methodologies are usually divided into two categories: static or dynamic [6]. Static gestures are those that only require the processing of a single image at the input of the classifier, the advantage of this approach is the lower computational cost. Dynamic gestures require the processing of image sequences and more complex gesture recognition approaches. In the literature, we can find several recognition methodologies based on supervised and unsupervised learning. We can cite some examples, such as neural networks [7–9], convolutional neural networks [10], support vector machines (SVM) [11, 12], nearest neighbours [13], graphs [14], distributed locally linear embeddings [15], and others [16].

Otiniano-Rodríguez et al. [12] present a methodology in which the features of gesture images are extracted through Hu and Zernike moments, while an SVM is used for classification. Another method is the use of neural networks to classify the data extracted from the images, as presented by the authors Tolba et al. [8], in which a special type of neural network is used, called learning vector quantization. The project from Nguyen et al. [9] was defined by principal component analysis (PCA) to select the best attributes and a neural network for classification. Oyedotun and Khashman [10] presented a methodology that uses several image processing operations to extract the shape of the hand and use it as input to compare two methods of classification: convolutional neural networks and stacked denoising autoencoder. Chevtchenko et al. [17] propose a method that uses Gabor features, Zernike moments, Hu moments, and contour-based descriptors to improve the features fed to the CNN, which is defined by feature fusion-based convolutional neural network (FFCNN). Ranga et al. [18] used

hybrid discrete wavelet transform-Gabor filter to extract the features and tests with different classifiers, adding a comparison with a CNN architecture.

In this work, we used two image bases of 24 gestures, some segmentation techniques and the use of convolutional neural networks (CNNs) for classification. Thus, with the proposed methodology, we demonstrated that with simple architectures of convolutional neural networks, it is possible to achieve excellent results for static gesture classification. We compared the proposed architectures with other existing networks in the literature and other gesture recognition methodologies. In the next sections, we present a brief description of the techniques we used, our proposed methodology, and the experiments we carried out. The final sections of this work show the results we obtained, a discussion and comparison with other works and, lastly, our conclusions and perspectives for future work.

2. Materials and Methods

This section discusses the techniques used in this work for image processing and data classification. In order to perform the training of a classifier, it is necessary to extract data. Besides, when dealing with images, it is important to extract features only from the regions of interest. Segmentation techniques, filters, and morphological operations are applied to enhance the relevant details to a particular application. With the use of convolutional neural networks, it is not necessary to extract feature vectors from the images since the input of this type of network is the image itself. With these considerations, a good preprocessing phase should separate well important image features from noise.

2.1. Morphological Operations. The use of morphological filters is common as a tool for extracting image components so that they are useful in the representation and description of forms. These filters are applied during image processing to remove or highlight features from a segmentation by closing holes and/or reducing noise. They are defined by two elementary operations: erosion and dilation; other important operations are opening and closing. These operations act with structuring elements that have the most diverse forms, thus altering the final result even when the same filter is used. Some examples of structuring elements may be a vector or a square matrix. The use of these elements in an erosion operation will remove lines and square regions from the image, respectively. The closing operation, however, will remove holes contained in the pixel sets of the image.

2.2. Contour Extraction and Polygon Approximation. It is possible to define and extract the outlines of objects in an image. One of the methods to accomplish this is by means of a technique described in Suzuki and Be [19], in which for each element in the image, a set of points is calculated around its contour. However, if the image quality is low, the resulting contour shape will probably be noisy.

To solve this problem, it is possible to apply a polygonal approximation technique, as described in Ramer [20]. This method recursively eliminates contour points whose distance from the mean contour curve is above a certain epsilon value (maximum distance between the original and simplified curve). Therefore, the epsilon parameter must be chosen so that the generated polygon does not have a shape too different from the original contour, but represents a subtle simplification.

2.3. Artificial Neural Networks. Artificial neural networks are structures widely used for classification tasks. When using this mechanism, the object to be classified is presented to the network through the activation of artificial neurons in the input layer. These activations are processed in the inner layers, and the result emerges as a pattern in the output layer. A network that has a single layer is known as a simple perceptron but is only capable of solving linearly separable problems. In order to solve nonlinearly separable problems, it is necessary to use a multilayer perceptron (MLP) neural network. The MLP consists of an input layer, a number of hidden layers, and an output layer, as can be seen in Figure 1.

The adequacy of the neural network to a classification task is determined by its weights. The training of a neural network consists of adapting the weights of the artificial synapses to a specific problem. In order to train an MLP, one of the most common methods is to use the backpropagation algorithm, in which the connection weights in the inner layers are modified as the error is propagated in the reverse direction, with the purpose of adapting the neural network to the resolution of the problem [22].

2.4. Convolutional Neural Networks. Convolutional neural networks, or CNNs, are widely used for image classification, object recognition, and detection [23]. Three types of layers can summarize its structure: convolution, pooling, and classification, as shown in Figure 2. The CNN architecture must be defined according to the application and is usually defined by the number of alternate convolution and pooling players, number of neurons in each layer, and choice of activation function. Some network architectures are already defined in the literature such as LeNet [24], InceptionResNetV2 [25], InceptionV3 [26], VGG16 [27], VGG19 [27], ResNet50 [28], and DenseNet201 [29].

In the context of image classification, the input for a CNN is an image represented by an arbitrary color model. At the convolution layer, every neuron is associated with a kernel window that is convolved with the input image during CNN training and classification. This convolution kernel is composed of the weights of each associated neuron. The output of this convolution step is a set of N images, one for each of the N neurons. Because of convolution, these new images can contain negative values. In order to avoid this issue, a rectified linear unit (ReLU) is used to replace negative values by zero. The outputs of this layer are called feature maps.

After a convolution layer, it is common to apply a pooling layer. This is important because pooling reduces the dimensionality of feature maps, which subsequently reduces the network training time. Some architectures alternate between convolution and pooling, for example, GoogLeNet

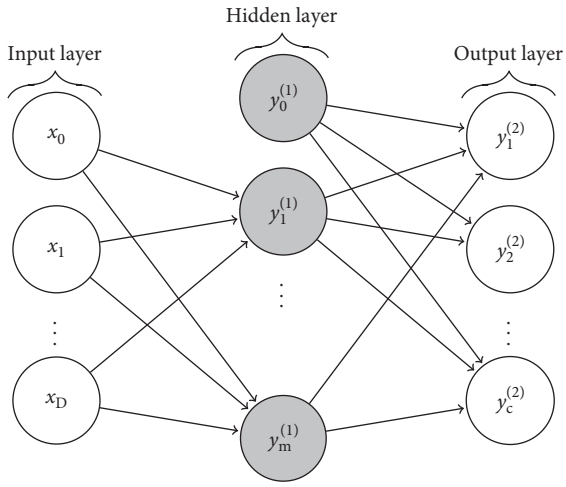


Figure 1: Example of MLP. Image adapted from Stutz [21].

[30] has five convolution layers followed by one pooling layer. At the end of the convolution and pooling architectures, there is a multilayer perceptron neural network that performs classification based on the feature maps computed by the previous layers.

Because of its large number of layers and successful applications, CNNs are one of the preferred techniques for deep learning. Its architecture allows automatic extraction of diverse image features, like edges, circles, lines, and texture. The extracted features are increasingly optimized in further layers. It is important to emphasize that the values of the kernel filters applied in the convolution layers are the result of backpropagation during CNN training.

2.5. Segmentation by Color. Segmentation subdivides an image into regions, so that it is possible to highlight regions that contain characteristics of interest [31, 32]. Therefore, segmentation algorithms can be implemented to separate colors, textures, points, lines, discontinuities, borders, among others. The segmentation process varies according to the problem.

In the case of gesture recognition, the entire background region of the image is not of interest, so only the set of pixels with the presence of the human hand must be maintained. One method for this segmentation is the implementation of background removal, where image samples are collected from the environment and then objects are added to the scene. In this way, the pixels of the new images are compared with the images of the scenario. The regions that show large amount of pixel differences, possibly containing the hand and gesture, are considered foreground. Although it is a good method, this type of segmentation is quite susceptible to variations in lighting.

An alternative is the color-segmentation technique, where it is possible to divide the images into regions that have previously defined color tones. In the case of the presented problem, the color tones to be segmented are similar to human skin tones. To solve this, we can train an MLP network that learns the skin color tones and then classifies which pixels in the image belong to the skin color sets. This method is more robust because it only depends on the correct learning of color sets.

3. Methodology

The proposed methodology can be visualized in the flow chart of Figure 3. The images are obtained from the database. Then, the images go through an image processing stage, in which the following operations occur: color segmentation using an MLP network, morphological operations of erosion and closing, contour generation, and polygonal approximation, to remove image noise.

After segmentation, binary images are obtained, so a logical AND operation is performed between these images and the originals, in order to preserve the information contained in the fingers and the surface of the hand. After these steps, the images are used to train a CNN and assess the performance of the technique with cross validation. Finally, the validation results are analyzed.

4. Experiments

The results of the experiments were obtained by classifying two image sets, a self-acquired dataset and another set available in the literature [33]. The self-acquired dataset was built by capturing the static gestures of the American Sign Language (ASL) alphabet, from 8 people, except for the letters J and Z, since they are dynamic gestures. To capture the images, we used a Logitech Brio webcam, with a resolution of 1920×1080 pixels, in a university laboratory with artificial lighting. By extracting only the hand region, we defined an area of 400×400 pixels for the final image of our dataset. To increase the dataset variety, we applied 30-degree rotations to all images, clockwise and counterclockwise, and a vertical scaling increase of 20%. Thus, the final image set is represented by 24 gestures and a total of 11100 samples. Such a large number of samples provided a variety of shapes and skin tones, as can be seen in Figure 4, which shows several samples of gesture A. Other samples of gestures can be seen in Figure 5.

At the beginning of the proposed methodology, we trained an MLP neural network to perform skin color segmentation. For this, we defined a network whose architecture has two hidden layers with 5 and 10 neurons, respectively. In the input layer, we had 3 attributes referring to the color tones of the RGB palette. The output layer has only one neuron for a binary decision: 0 for nonskin and 1 for skin. In our experiment, 5 images were used to train the MLP skin classifier. These images were used in previous works [34, 35] and they demonstrate robustness in the proposed segmentation methodology. An example of the training images used can be visualized in Figure 6.

After the segmentation training, during its use in the dataset images, we observed that some segmentations showed holes and noise in the hand region, as we can see in Figure 7.

To correct these problems, a morphological erosion operation for noise removal is applied using a horizontal line

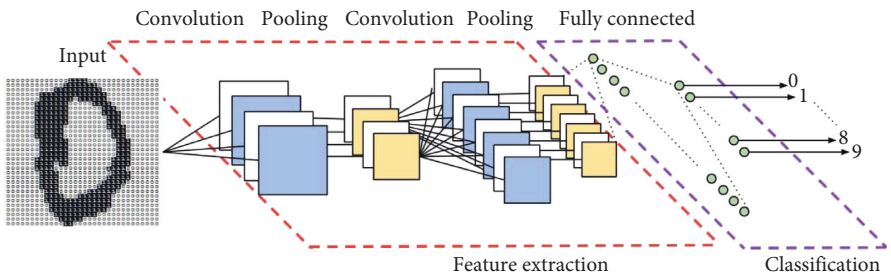


Figure 2: Example of CNN and its layers. Image adapted from Vargas et al. [23].

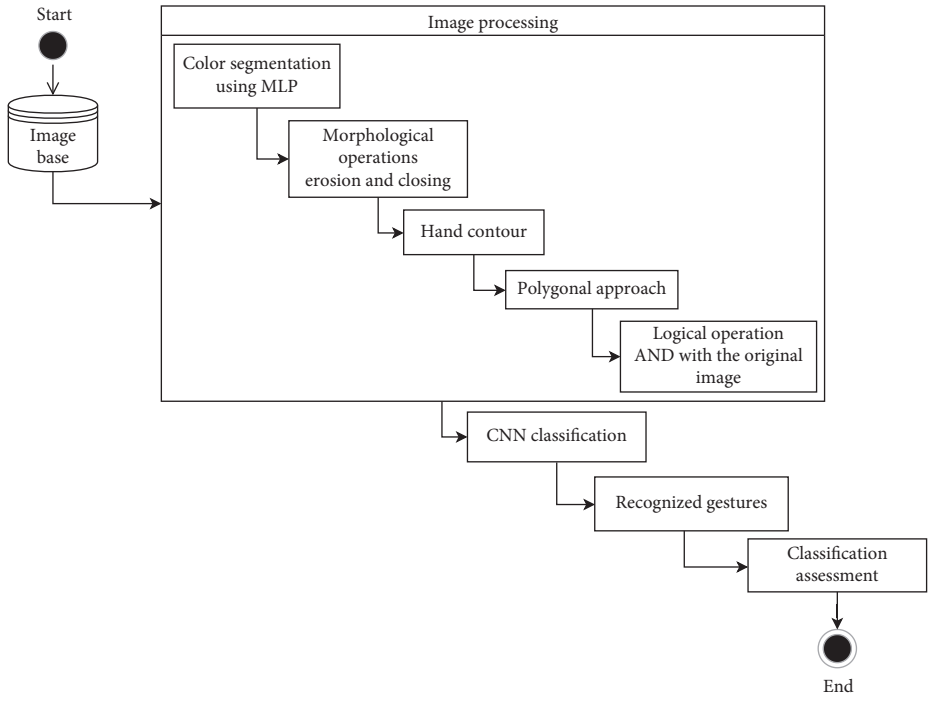
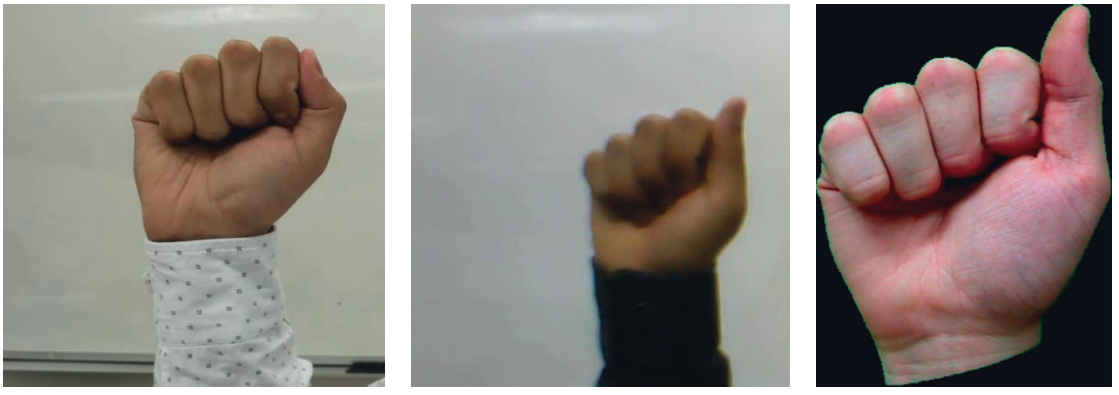


Figure 3: The sequence of processes in our methodology.

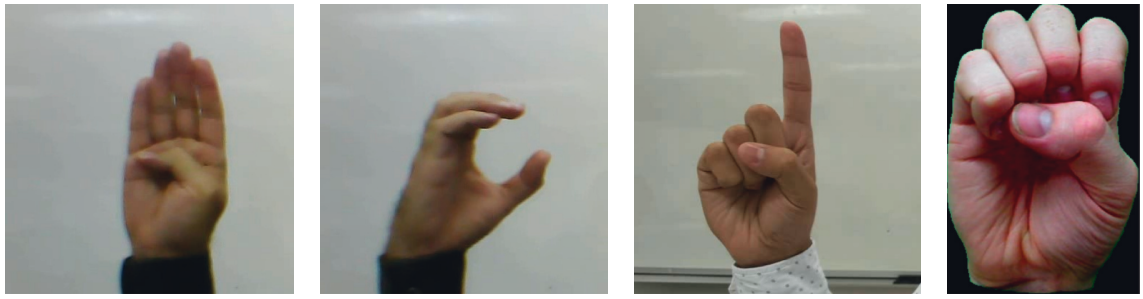


(a) (b) (c)

Figure 4: Samples for gesture A.

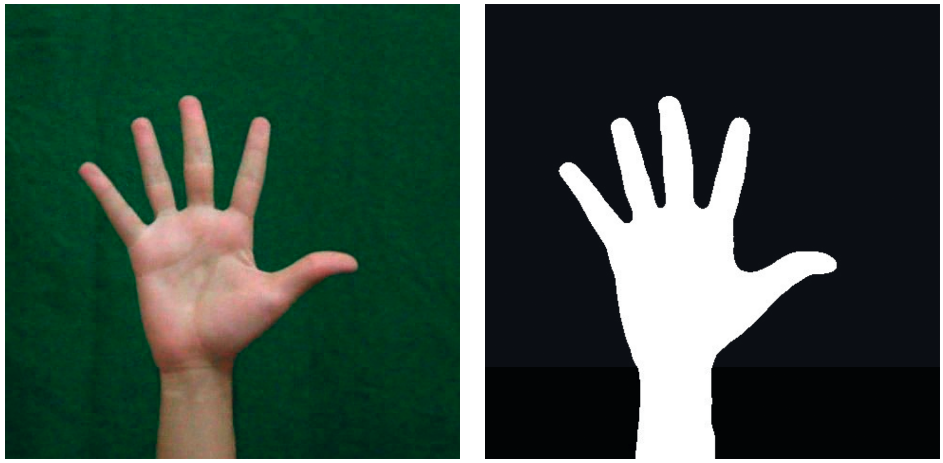
structuring element with the size of 9 pixels. In sequence, a closing operation is performed using a square element with a 13-pixel dimension to remove the holes in the hand region. The use of these two operations in sequence presented good results, as can be analyzed in Figure 8.

Despite the good results obtained after applying the morphological operations, a small subset of images still had holes in the inside of the palm and noise at the edges. To solve these problems, we computed a polygonal approximation [22] of the hand region through its generated



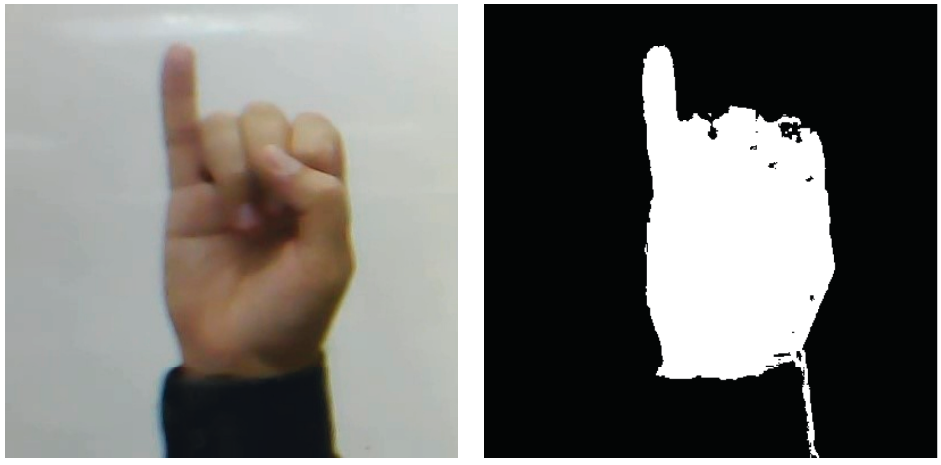
(a) (b) (c) (d)

Figure 5: Samples from the adopted dataset used, from left to right: gestures b, c, d, and e.



(a) (b)

Figure 6: Image (a) and mask (b) used to train the MLP skin color classifier.



(a) (b)

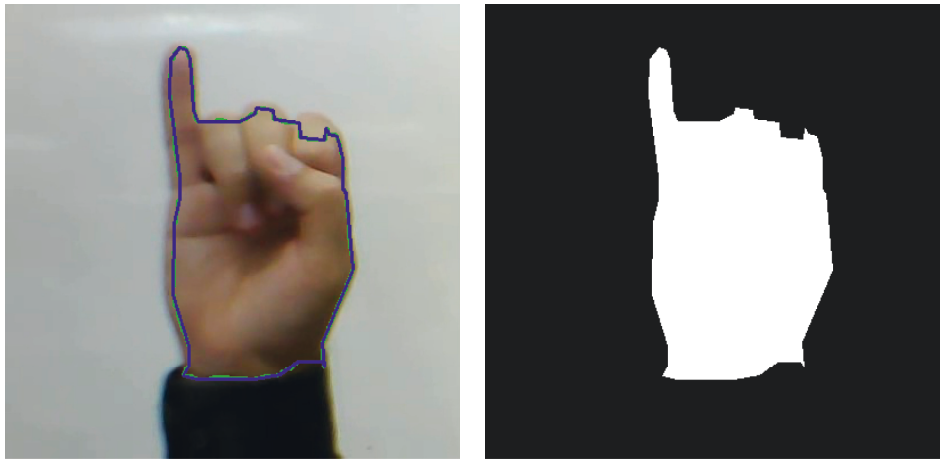
Figure 7: Original image (a) and faulty segmentation (b).

contour [19] (Figure 9). The final segmentation result is the region contained within the polygonal approximation.

After analyzing all the segmented images classes contained in the dataset, we noticed that, for some gestures, there is great similarity in its shape, as we can see in Figures 10 and 11. It is possible to notice the similarity between the shapes of gestures A and E and gestures S and T, so we came to the conclusion that if we trained a CNN with these binary images, this high shape similarity would confuse the CNN.

(a) (b)

Figure 8: Segmentation after erosion (a) and closing (b).



(a) (b)

Figure 9: Image and its hand contours (a) and the final hand segmentation (b).

(a) (b)

Figure 10: Similar segmented images: gestures A (a) and E (b).

(a) (b)

Figure 11: Similar segmented images: gestures S (a) and T (b).

One measure to reduce the similarity between gestures was to use the segmented images as masks and thus apply a logical AND operation between the segmentations and the original images in grayscale. In this way, the information on the shape gesture and the characteristics of the palms features and the fingers positions are preserved, as shown in Figures 12 and 13. Thus, gestures A and E, for example, have less similarity due to the position of the thumb in the image.

After performing the previous steps on all dataset images, we used them to train the static gesture classifier. Thus, we trained and tested four proposed CNN architectures, defined in Table 1, where we present the convolution and pooling layers and the sizes of the convolutional kernels.

The proposed architectures use the activation rectified linear units (ReLUs) function in their convolution and pooling layers. In order to classify the features extracted by the defined CNN architectures, we adopted an MLP with 400 and 800 neurons in its two intermediate layers, using a ReLU activation function and a softmax output layer with 24 neurons.

Other tests were performed with CNN architectures known in the literature, such as LeNet, InceptionResNetV2, InceptionV3, VGG16, VGG19, ResNet50, and DenseNet201.

Our experiments used the holdout cross-validation method. Therefore, we adopted a division of 75% and 25% for training and testing, respectively, as defined in Kohavi [36]. From the 75% of data used for training, 5% were reserved for validation during training. Also, ten rounds of training and testing were adopted, in which the training and test sets were randomly permuted. The holdout metrics were obtained by averaging the results of the 10 rounds, using accuracy, recall, and F1 score.

The tests were run on a server with configuration Intel (R) Core i7-6800K @ 3.40 GHz CPU, 64 GB of RAM, and two NVIDIA Titan Xp GPUs.

5. Results and Discussion

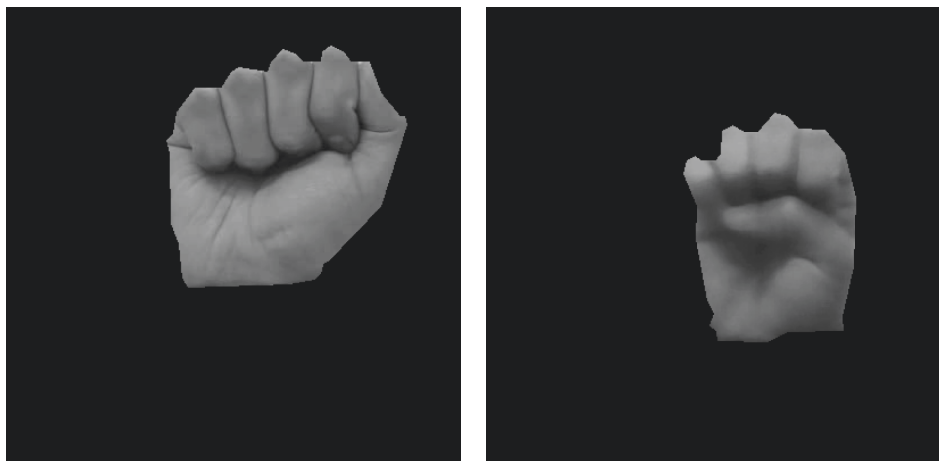
The results obtained with the proposed architectures are shown in Table 2, in which we present their accuracy, precision, recall, and F1 score. The proposed architectures presented good results, with average rates of success of 96%. For CNN 1, with only two layers of convolution, it presented an accuracy rate of 94.7%, and for CNN 2, 3 and 4 accuracies remained above 96%.

It is clear that starting from 3 layers of convolution, together with pooling layers, modifying this type of neural network architecture does not increase the feature extraction and classification capacities of the network. Therefore, there is no significant increase in accuracy, but only in the computational cost of the network. However, if we analyse the network convergence times during training, in Figures 14 and 15, it is seen that increasing the number of convolution layers reduces the number of epochs necessary for the network to converge, thus extracting the data faster. We can see that CNN 1 (see Figure 14) converges at epoch 16, while CNN 3 (see Figure 15) converges at epoch 7.

The architectures already defined in the literature presented accuracy rates with values up to 99%, as we can see in Table 3. However, they are more complex than the proposed architectures, some of them are more than 200 layers deep. Thus, these architectures are also capable of extracting characteristics of the images more quickly, presenting the smallest numbers of epochs for convergence, as is the case of InceptionV3 and ResNet50, which converged at epochs 5 and 7, respectively, as can be seen in Figures 16 and 17, respectively.

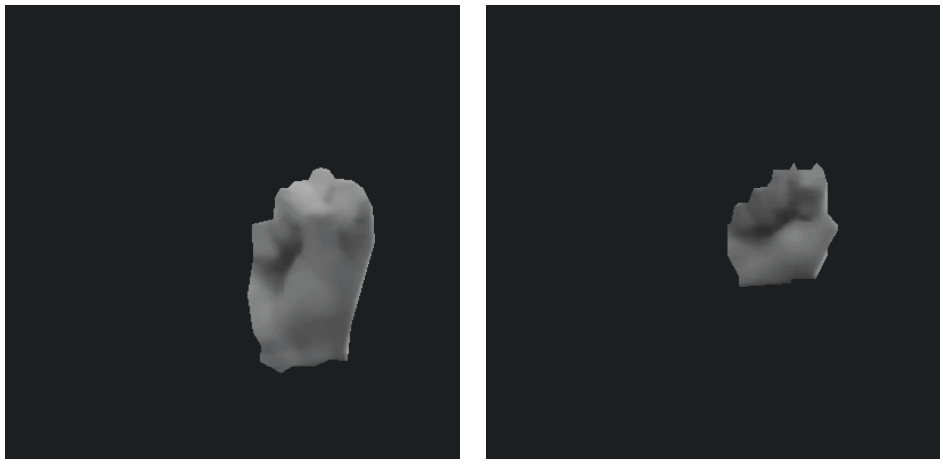
Results were also generated using the individual image datasets: our own dataset and two others available in the literature [33, 37]. These results are shown in Tables 4 and 5. It is possible to conclude that the CNNs are able to extract the features and classify the patterns well enough to reach correct answers close to 100%. Due to this behavior, we adopted the use of two datasets together, our own dataset and that in [33], demonstrating robustness of the methodology independent of the base used.

In addition to comparing the results obtained by the proposed methodology with other CNN architectures from the literature, we also make a comparison with other related



(a) (b)

Figure 12: Gestures A (a) and E (b) after AND logical operation with the hand segmented image.



(a) (b)

Figure 13: Gestures S (a) and T (b) after AND logical operation with the hand segmented image.

Table 1: Proposed CNN architectures.

Depth	CNN 1	CNN 2	CNN 3	CNN 4
1	Convolutional (5 × 5)	Convolutional (5 × 5)	Convolutional (5 × 5)	Convolutional (5 × 5)
2	Max pooling (2 × 2)	Max pooling (2 × 2)	Convolutional (7 × 7)	Convolutional (7 × 7)
3	Convolutional (7 × 7)	Convolutional (7 × 7)	Max pooling (2 × 2)	Convolutional (9 × 9)
4	Max pooling (2 × 2)	Max pooling (2 × 2)	Convolutional (5 × 5)	Max pooling (2 × 2)
5		Convolutional (5 × 5)	Convolutional (7 × 7)	Convolutional (5 × 5)
6		Convolutional (7 × 7)	Max pooling (2 × 2)	Convolutional (7 × 7)
7		Max pooling (2 × 2)	Convolutional (5 × 5)	Convolutional (9 × 9)
8			Convolutional (7 × 7)	Max pooling (2 × 2)
9			Convolutional (9 × 9)	Convolutional (5 × 5)
10			Max pooling (2 × 2)	Convolutional (7 × 7)
11				Convolutional (9 × 9)
12				Max pooling (2 × 2)

works that were trained and tested with the same gestures defined by the ASL. In [12], the Zernike moments are used for training and classification by means of an SVM; in [8, 9], different methodologies are presented for feature extraction, but neural networks are used for classification, in [10, 17, 18], different CNN architectures are used. We observed that the use of CNN combined with efficient image processing yields excellent performance, with higher

Table 2: Results obtained with the proposed architectures.

	CNN 1	CNN 2	CNN 3	CNN 4
Accuracy	94.71%	96.5%	96.83%	96.83%
Precision	94.77%	96.54%	96.86%	96.86%
Recall	94.7%	96.49%	96.82%	96.82%
F1 score	94.7%	96.49%	96.82%	96.82%

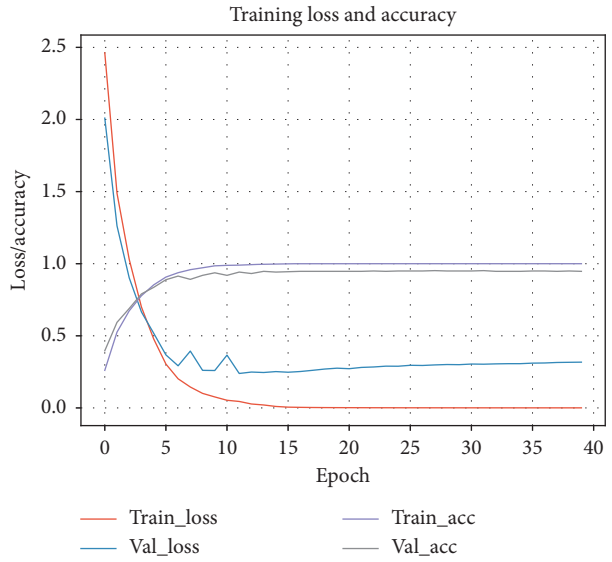


Figure 14: Accuracy and loss during CNN 1 training and validation.

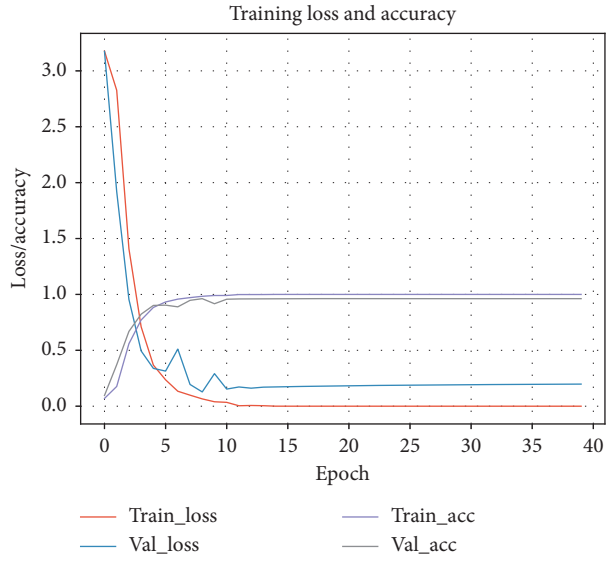


Figure 15: Accuracy and loss during CNN 3 training and validation.

Table 3: Results obtained with architectures available in the literature.

	InceptionV3	Inception ResNetV2	ResNet50	Dense Net201	VGG16	VGG19	LeNet
Accuracy	99.46%	99.62%	97.0%	98.38%	97.54%	97.29%	93.54%
Precision	99.46%	99.62%	97.07%	98.47%	97.57%	97.32%	93.61%
Recall	99.45%	99.62%	96.99%	98.37%	97.54%	97.29%	93.53%
F1 score	99.45%	99.62%	97.0%	98.37%	97.54%	97.3%	93.54%

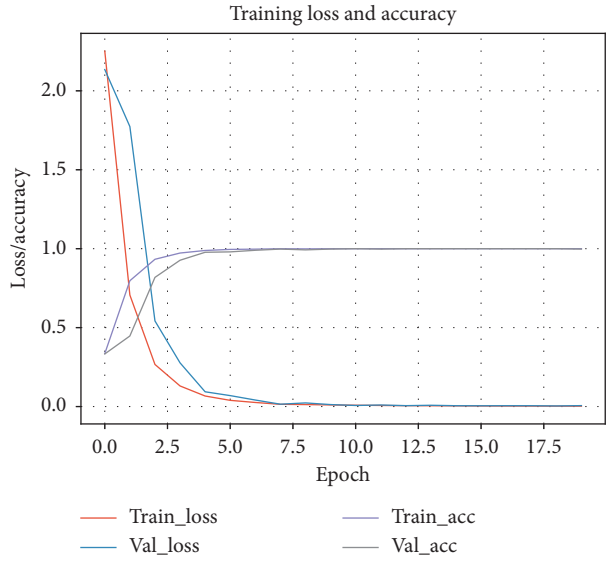


Figure 16: Accuracy and loss during inception V3 training and validation.

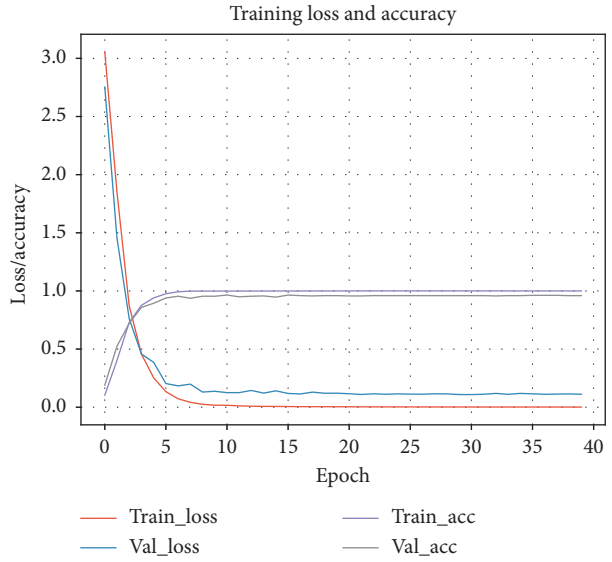


Figure 17: Accuracy and loss during ResNet50 training and validation.

Table 4: Results obtained for each dataset individually using the proposed architectures.

	CNN 1	CNN 2	CNN 3	CNN 4
Own dataset	97.53%	97.74%	98.32%	98.24%
Barczak et al. [33]	98.09%	98.99%	98.99%	99.40%
Moeslund's [37]	99.61%	99.41%	99.61%	99.41%

accuracy rates than the related works used for this comparison, as can be seen in Table 6.

The proposed method results were superior to other methods that use the same method of classification of gestures, such as [10]. Thus, the success rate of 96.83% shows the robustness of the presented methodology, the importance of preparing the images before the classification process and the need to study and analyse the types of convolutional neural network architectures. When using only dataset [33], we obtained higher accuracy rates than [17, 18], as we can see in Tables 4 and 6. In the proposed methodology with a combined dataset, when we use our own set and [33] to increase the diversity of hands, we obtained a slightly lower accuracy. But, in this way, we can demonstrate that our methodology is able to adapt to a greater variety of hand data.

Table 5: Results obtained for each dataset individually using the architectures available in the literature.

	InceptionV3	Inception ResNetV2	ResNet50	Dense Net201	VGG16	VGG19	LeNet
Own dataset	98.44%	99.35%	98.89%	99.27%	98.10%	98.32%	86.15%
Barczak et al. [33]	99.23%	99.74%	99.11%	98.89%	99.58%	99.53%	98.15%
Moeslund's [37]	99.41%	99.80%	98.24%	99.02%	99.61%	99.61%	97.65%

Table 6: Comparison with related work.

	Proposed method—combined dataset	Proposed method—dataset [33]	Otiniano-rodríguez et al. [12]	Nguyen et al. [9]	Tolba et al. [8]	Oyedotun and Khashman [10]	Chevtchenko et al. [17]	Ranga et al. [18]
Accuracy	96.83%	99.4%	96.27%	94.3%	90.83%	91.33%	98.06%	97.01%

6. Conclusions

One of the problems in gesture recognition is dealing with the image background and the noise often present in the regions of interest, such as the hand region. The use of neural networks for color segmentation, followed by morphological operations and a polygonal approximation, presented excellent results as a way to separate the hand region from the background and to remove noise. This step is important because it removes image objects that are not relevant to the classification method, allowing the convolutional neural network to extract the most relevant gesture features through their convolution and pooling layers and, therefore, to increase network accuracy. The proposal to make a logical AND operation with the segmentation masks and the original images provided the relevant information of the palms and fingers. Thus, the proposed CNN architectures achieved high success rates at a relatively low computational cost. It was superior to methodologies mentioned in related works, confirming the robustness of the presented method. In addition, the proposed architectures reached accuracies very similar to the architectures already defined in the literature, although they are much simpler and have a lower computational cost. This is possible due to the proposed image processing methodology, in which unnecessary information is removed, allowing improved feature extraction by the CNN. The proposed methodology and CNN architecture open the door to a future implementation of gesture recognition in embedded devices with hardware limitations. The proposed methodology approaches only cases of gestures present in static images, without hand detection and tracking or cases of hand occlusion. In the future, we intend to work on these particular cases in a new data preprocessing methodology, investigating other techniques of color segmentation [38–40] and deep learning architectures [41, 42].

Data Availability

In this article, two image databases were used, they were used together. One of them is known in the literature and has been referenced. The other was produced by the authors, and it cannot be made publicly available at this time. However, the data used are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

Thanks are due to Fundação Cearense de Apoio ao Desenvolvimento Científico e Tecnológico (FUNCAP), Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES), and Universidade Federal do Ceará (UFC) for financial support.

References

[1] National Research Council, *How People Learn: Brain, Mind, Experience, and School: Expanded Edition*, The National Academies Press, Washington, DC, USA, 2000.

[2] M. W. Cohen, N. B. Zikri, and A. Velkovich, "Recognition of continuous sign language alphabet using leap motion controller," in *Proceedings of the 2018 11th International Conference on Human System Interaction (HSI)*, pp. 193–199, Gdańsk, Poland, July 2018.

[3] A. Betancourt, P. Morerio, C. S. Regazzoni, and M. R. Auterberg, "A structure for deoxyribose nucleic acid," *Circuits and Systems for Video Technology*, vol. 25, no. 5, pp. 744–760, 2015.

[4] S. Riofrío, D. Pozo, J. Rosero, and J. Vásquez, "Gesture recognition using dynamic time warping and kinect: a practical approach," in *Proceedings of the 2017 International Conference on Information Systems and Computer Science (INCISCOS)*, pp. 302–308, Quito, Ecuador, November 2017.

[5] Technavio, *Global Robotics Market 2015–2019*, Research and Markets, Dublin, Ireland, 2015.

[6] P. K. Pisharady and M. Saerbeck, "Recent methods and databases in vision-based hand gesture recognition: a review," *Computer Vision and Image Understanding*, vol. 141, no. 4356, pp. 152–165, 2015.

[7] N. A. Ming-Hsuan Yang and M. Tabb, "Extraction of 2d motion trajectories and its application to hand gesture recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 8, pp. 1061–1074, 2002.

[8] A. S. Tolba, M. A. Elsoud, and O. A. Elnaser, "LVQ for hand gesture recognition based on DCT and projection features," *Journal of Electrical Engineering*, vol. 60, no. 4, pp. 204–208, 2009.

[9] T.-N. Nguyen, H.-H. Huynh, and J. Meunier, "Static hand gesture recognition using principal component analysis

combined with artificial neural network," *Journal of Automation and Control Engineering*, vol. 3, no. 1, 2015.

[10] O. K. Oyedotun and A. Khashman, "Deep learning in vision-based static hand gesture recognition," *Neural Computing and Applications*, vol. 28, no. 12, pp. 3941–3951, 2017.

[11] D.-Y. Huang, W.-C. Hu, and S.-H. Chang, "Gabor filter-based hand-pose angle estimation for hand gesture recognition under varying illumination," *Expert Systems with Applications*, vol. 38, no. 5, pp. 6031–6042, 2011.

[12] K. C. Otiniano-Rodríguez, G. Camara-Chávez, and D. Menotti, "Hu and zernike moments for sign language recognition," in *Proceedings of the International Conference on Image Processing, Computer Vision, and Pattern Recognition (IPCV)*, November 2012.

[13] M. V. den Bergh, D. Carton, R. De Nijs et al., "Real-time 3D hand gesture interaction with a robot for understanding directions from humans," in *Proceedings of the 2011 20th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, pp. 357–362, Atlanta, GA, USA, July 2011.

[14] J. Triesch and C. von der Malsburg, "A system for person-independent hand posture recognition against complex backgrounds," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 12, pp. 1449–1453, 2001.

[15] S. S. Ge, Y. Yang, and T. H. Lee, "Hand gesture recognition and tracking based on distributed locally linear embedding," in *Proceedings of the 2006 IEEE Conference on Robotics, Automation and Mechatronics*, vol. 171, no. 4356, pp. 1–6, Orlando, FL, USA, June 2006.

[16] P. K. Pisharady and M. Saerbeck, "Recent methods and databases in vision-based hand gesture recognition: a review," *Computer Vision and Image Understanding*, vol. 141, pp. 152–165, 2015.

[17] S. F. Chevtchenko, R. F. Vale, V. Macario, and F. R. Cordeiro, "A convolutional neural network with feature fusion for real-time hand posture recognition," *Applied Soft Computing*, vol. 73, pp. 748–766, 2018.

[18] V. Ranga, N. Yadav, and P. Garg, "American sign language fingerspelling using hybrid discrete wavelet transform-gabor filter and convolutional neural network," *Journal of Engineering Science and Technology*, vol. 13, no. 9, pp. 2655–2669, 2018.

[19] S. Suzuki and K. Be, "Topological structural analysis of digitized binary images by border following," *Computer Vision, Graphics, and Image Processing*, vol. 30, no. 1, pp. 32–46, 1985.

[20] U. Ramer, "An iterative procedure for the polygonal approximation of plane curves," *Computer Graphics and Image Processing*, vol. 1, no. 3, pp. 244–256, 1972.

[21] D. Stutz, "Recognizing handwritten digits using a two-layer perceptron and the mnist dataset," 2018.

[22] M. J. Bufo, "Aplicação de rede neural artificial como auxiliar na predição do desempenho de um land farming," *Dissertation (Master in Agricultural Engineering)*, Universidade Estadual de Campinas, São Paulo, Brazil, 2000.

[23] A. C. G. Vargas, A. M. P. Carvalho, and C. N. Vasconcelos, "Um estudo sobre redes neurais convolucionais e sua aplicação em detecção de pedestres," in *Proceedings of the SIBGRAPI—Conference on Graphics, Patterns and Images*, Sao Paulo, Brazil, October 2016.

[24] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, *Gradient-Based Learning Applied to Document Recognition*, Elsevier, Amsterdam, Netherlands, 1998.

[25] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," in *Thirty-First AAAI Conference on Artificial Intelligence*, San Francisco, CA, USA, February 2017.

[26] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2818–2826, Las Vegas, NV, USA, June 2016.

[27] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," http://arxiv.org/abs/1409.1556, 2014.

[28] K. He, X. Z. S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Las Vegas, NV, USA, June 2016.

[29] G. Huang, Z. Liu, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4700–4708, Honolulu, HI, USA, July 2017.

[30] C. Szegedy, W. Liu, Y. Jia et al., "Going deeper with convolutions," in *Proceedings of the Computer Vision And Pattern Recognition (CVPR)*, Boston, MA, USA, June 2015.

[31] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, Prentice-Hall, Upper Saddle River, NJ, USA, 3rd edition, 2006.

[32] R. Szeliski, *Computer Vision: Algorithms and Applications*, Springer, Berlin, Germany, 2010.

[33] A. L. C. Barczak, N. H. Reyes, M. Abastillas, A. Piccio, and T. Susnjak, "A new 2D static hand gesture colour image dataset for ASL gestures," *Research Letters in the Information and Mathematical Sciences*, vol. 15, no. 4356, pp. 12–20, 2011.

[34] R. F. Pinto Júnior, C. D. B. Borges, A. S. Nascimento, and I. C. de Paula Júnior, "Reconhecimento de gestos usando momentos invariantes de HU e redes neurais profundas," in *Proceedings of the X Encontro Unificado de Computação (ENUCOMP 2017)*, Teresina, PiauÍ, Brazil, December 2017.

[35] R. F. Pinto Júnior, C. D. B. Borges, A. S. Nascimento, and I. C. de Paula Júnior, *Detecção de Gestos Através da Extração de Características da Mão Humana*, Escola Regional de Informática do Piauí (ERIPI 2016), Edinburgh, UK, 2016.

[36] R. Kohavi, "A study of cross-validation and bootstrap for accuracy estimation and model selection," in *Proceedings of the 14th International Joint Conference on Artificial Intelligence IJCAI'95*, vol. 2, pp. 1137–1143, Morgan Kaufmann Publishers, San Francisco, CA, USA, August 1995.

[37] T. Moeslund's, *Gesture Recognition Database*, 2002.

[38] G. Xu, Y. Xiao, S. Xie, and S. Zhu, "Face detection based on skin color segmentation and adaboost algorithm," in *Proceedings of the 2017 IEEE 2nd Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*, pp. 1756–1760, Chongqing, China, March 2017.

[39] Y. Lei, W. Yuan, H. Wang, Y. Wenhu, and W. Bo, "A skin segmentation algorithm based on stacked autoencoders," *IEEE Transactions on Multimedia*, vol. 19, no. 4, pp. 740–749, 2017.

[40] H. Zuo, H. Fan, E. Blasch, and H. Ling, "Combining convolutional and recurrent neural networks for human skin detection," *IEEE Signal Processing Letters*, vol. 24, no. 3, pp. 286–293, 2017.

[41] S. Nie, M. Zheng, and Q. Ji, "The deep regression bayesian network and its applications: probabilistic deep learning for computer vision," *IEEE Signal Processing Magazine*, vol. 35, no. 1, pp. 101–111, 2018.

[42] A. Hassan and A. Mahmood, "Convolutional recurrent deep learning model for sentence classification," *IEEE Access*, vol. 6, pp. 13949–13957, 2018.