Multimedia Tools and Applications (2023) 82:6807–6826
https://doi.org/10.1007/s11042-022-13423-9

A vision-based deep learning approach for independent-users Arabic sign language interpretation

Mostafa Magdy Balaha[1] · Sara El-Kady[1] · Hossam Magdy Balaha[1] · Mohamed Salama[1] · Eslam Emad[1] · Muhammed Hassan[1] · Mahmoud M. Saafan[1]

Received: 12 January 2021 / Revised: 31 March 2022 / Accepted: 2 July 2022 /
Published online: 10 August 2022

Abstract
More than 5% of the people around the world are deaf and have severe difficulties in communicating with normal people according to the World Health Organization (WHO). They face a real challenge to express anything without an interpreter for their signs. Nowadays, there are a lot of studies related to Sign Language Recognition (SLR) that aims to reduce this gap between deaf and normal people as it can replace the need for an interpreter. However, there are a lot of challenges facing the sign recognition systems such as low accuracy, complicated gestures, high-level noise, and the ability to operate under variant circumstances with the ability to generalize or to be locked to such limitations. Hence, many researchers proposed different solutions to overcome these problems. Each language has its signs and it can be very challenging to cover all the languages' signs. The current study objectives: (i) presenting a dataset of 20 Arabic words, and (ii) proposing a deep learning (DL) architecture by combining convolutional neural network (CNN) and recurrent neural network (RNN). The suggested architecture reported 98% accuracy on the presented dataset. It also reported 93.4% and 98.8% for the top-1 and top-5 accuracies on the UCF-101 dataset.

Keywords Arabic Sign Language (ASL) · Convolutional Neural Network (CNN) · Deep Learning (DL) · Recurrent Neural Network (RNN) · Sign Language Recognition (SLR) · Video recognition

1 Introduction

Sign language is always paired with deaf people who use it to communicate with each other but the problem arises when a deaf try to communicate with a normal person who has no prior experience with sign language [39]. This gap limits the interactions and social life

✉ Hossam Magdy Balaha
hossam.m.balaha@mans.edu.eg

[1] Computers and Systems Engineering Department, Faculty of Engineering, Mansoura University, Mansoura, Egypt

experience of deaf people as it requires an expert in sign language to ease the communication [71]. Scientists and researchers tried to shed light on this problem and search for such a solution to replace the intermediate human or the expert necessity with an automated interpreter that could convert the hand kinematics and facial expressions to words or phrases [65]. Despite these great efforts and tries in that field and the state-of-the-art development in artificial intelligence and deep learning techniques [13] to find a solution nevertheless, there is no optimal interpreter up to now due to the different challenges and difficulties that face them [58].

A sign language interpreter system (SLIS) accepts the human visual sign as a set of frames from any capturing medium such as cameras and outputs the corresponding meaning of that sign [70]. That sign can be represented as a text or sound as shown in Fig. 1. Training a SLIS requires a unique sign to represent each alphabet, number, and hence, it will result in a massive amount of data and signs required to be processed especially for each available language that exists. Any minimal differences in the signs can affect the interpreter's performance such as the performer himself as the interpreter can be designed to be user-dependent or user-independent. In the first case, the interpreter depends on the person while in the latter one, the user is not a problem anymore.

Different challenges in sign language recognition require to be solved in any SLIS starting from collecting the dataset to deploying the overall system [26, 56]. Some of these differences can be summarized as follows:

- **Viewpoint Variance**: Different people can capture the same sign with different poses and hand kinematics.
- **Environment**: The background, lighting, landmarks, and other elements can exist in the captured sign.
- **Complex Gestures**: A sign can be complex to be made by a person especially if the word is less used between people.
- **Facial Expressions**: A sign can include a facial expression. The face may include glasses, earrings, etc. and this may infer the system.
- **Non-Symmetric Signs**: A word can be expressed in different poses or styles between languages. Also, there are unique signs for each language.

In a-glance, over years scientists tried to find solutions for each challenge and problem to implement such interpreters using different methods and approaches. Mainly, we could classify these approaches into two groups; either sensor-based or vision-based. In the sensor-based approach, the user wears sensors such as colored-gloves or special-gloves, and a motion capture system captures the sign but this approach has different drawbacks [21]. For example, it was impractical in daily life situations as it obligates the user to

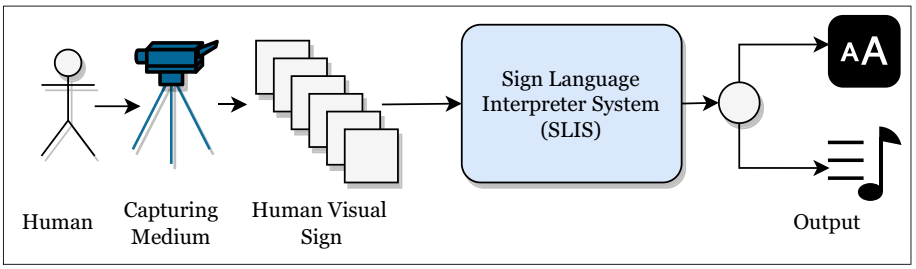

Fig. 1 A Sign Language Interpreter System (SLIS) Overview

wear such sensors that depend on the continuous power supply, wires, and other requirements. This reason was enough for the authors to cease this approach for experiments and work on the second approach. In the vision-based approach, the system relies on image processing and computer computations for processing images and videos in addition to machine learning and deep learning techniques to classify and predict the processing data [42]. Such approaches as Hidden-Markov-Model (HMM) [15], Artificial Neural Networks (ANN) [76], Convolutional Neural Network (CNN) [5, 53], and Recurrent Neural Network (RNN) [50]. The advantage of the second approach is the low-cost hardware. The capturing medium can be smartphones cameras.

In the current work, the authors depended on the vision-based approach and the contributions can be summarized as follows:

- Revising the literature related to the different built systems and frameworks.
- Proposing a new Arabic sign language dataset.
- Suggesting a deep learning framework using both CNNs and RNNs for Arabic sign language interpretation.
- Focusing on the working mechanism of user-independent approach and appliance of it.
- Performing different experiments and comparing the current work results with other published state-of-the-art results.

The rest of the paper is organized as follows, in Section 2, the related work and studies are discussed. In Section 3, the available Arabic sign language datasets are presented and the proposed dataset is discussed in detail. In Section 4, the pre-processing stages performed on the suggested dataset are discussed. In Section 5, the suggested deep learning architecture is presented and discussed in detail. In Section 6, the experiments and their corresponding results are reported. Finally, in Section 7, the presented work is concluded and the future work is presented.

2 Related work

Sign language recognition is studied by different researchers in different approaches since 1990 [1, 22, 62, 63]. In this section, the related literature is discussed. They include several works and methods throughout the years. Tamura et al. [67] assumed the sign word was composed of a time sequence of units called cheremes which consisted of handshape, movement, and location of the hand. They expressed the 3D features of these factors and converted them into 2D image features and classified the motion image of sign language with the 2D features.

Keskin et al. [40] created realistic 3D hand models that represented the hand with 21 different parts and trained Random Decision Forests (RDFs). They used the RDF to perform per-pixel classification and assigned each pixel to a hand part. It was then fed into a local mode finding algorithm to estimate the joint locations for the hand skeleton. They also described a support vector machine (SVM) model to recognize the Arabic sign language (ASL) digits based on this method. They achieved a high recognition rate on live depth images in real-time. Nandy et al. [52] created a video database for various signs of the Indian sign language. They used the direction histogram, which appealed for illumination and orientation invariance, as the features used in classification. They used two different approaches for recognition which were the Euclidean distance and K-nearest neighbor metrics.

Mehdi et al. [51] used 7-sensor glove of the 5DT Company. It was used to get the input data of the hands' movements with artificial neural networks (ANN). It was used as the classifier to recognize the signs' gestures. They achieved an accuracy value of 88%. López-Noriega et al. [47] followed their same approach and also offered a graphical user interface made with ".NET". Hidden Markov Model (HMM) based model was used and worked effectively in continuous and real-time sign language recognition tasks by Starner et al. [61]. They used gloves images as an input for the HMM. They proposed a recognition method based on the HMM. They used color gloves to capture hand shape, orientation, and trajectory. They represented HMM-based systems for recognizing the sentence-level ASL. They managed to get high word accuracy results.

Hienz et al. [35] used colored cotton gloves to make it easy to extract features. They converted the sequence of videos into feature vectors and then fed them to an HMM to classify them. They have achieved accuracy values from 92% to 94%. Grobel et al. [31] and Parcheta et al. [54] also followed the same approach. In a brief, these previous approaches were able to achieve high accuracy values. But, they could not be used in real daily life as they required the wearing of gloves and were limited to a fixed environment which isn't natural. Actually, many of them were user-dependent which means that they must be trained on each user which isn't logical and unnatural. Due to the previous reasons; Youssif et al. [77] tended to generalize and proposed a model based on the HMM that did not depend on users nor require gloves. On the other hand, their model fell into the trap of low accuracy as it reached a value of 82%.

CNN is widely used in the field of image recognition and classification. Researchers made many studies using it with the SLR. Masood et al. [49] proposed a CNN model for ASL's character recognition. They were able to use CNN to achieve an overall accuracy of 96% on a 2,524 ASL gestures image dataset. Wadhawan et al. [72], Bheda et al. [16] and Tao et al. [68] offered a CNN architecture to classify different languages' signs alphabet with accuracies 99%, 82.5% and 100% respectively.

CNN uses a frame-by-frame manner in its work. Coupling CNN with RNN can keep information over time, especially in videos. Due to this ability, dynamic signs can be recognized more accurately. Yang et al. [75] proposed an effective continuous sign language recognition method. It was based on the combination of CNN and long short-term memory (LSTM). They achieved remarkable accuracies in the experiments on their self-built dataset. 3D Convolutional Neural Network (3D-CNN) based models, instead of 2D-CNN, require another phase of the RNN to keep information over time. 3D-CNN was able to take multi-frames of a video at once which helped to learn the sequence between frames without the need for RNN. Huang et al. [37] and Al-Hammadi et al. [3] proposed models based on that approach. The approach proposed in the current study is based on the CNN-RNN approach, specifically, the authors use double CNN as features extractors and for the RNN, Bi-directional long short-term memory (BiLSTM) layers are used. The BiLSTM layers are used to identify the complex sequences in videos to overcome the conflicts between different classes.

3 Arabic sign language datasets

Many available datasets in Arabic sign language that focus on letters or words are based on specific conditions such as: (i) the user must wear gloves or (ii) many images refer to static words [3, 40, 49]. So that, the major goal, which is the independence of unnecessary features related to specific users or the surrounding environment, can be achieved. This section starts

with presenting the available datasets in sign language and after that, the proposed dataset is presented in detail.

3.1 Available sign language datasets

Latif et al. [43] presented an **Arabic Alphabets Sign Language Dataset** named "ArASL". It consisted of 54,049 images. It was compiled by more than 40 volunteers for the 32 standard Arabic signs and alphabets. They mentioned that the number of images per class was not the same. It differs fr om one class to another. They created a Comma-Separated Values (CSV) file that contained the Label of each image. It is available online at https://data.mendeley.com/datasets/y7pckrw6z2/1.

Sign Language Digits Dataset is prepared by "Turkey Ankara Ayrancı Anadolu High School Students" [82]. Each image size is (100 × 100) pixels in the Red-Green-Blue (RGB) color space. It consists of 10 classes (Digits from 0 to 9). The total number of images is 2,062. It was collected from 218 students where the number of samples per student is 10. It is available online at https://www.kaggle.com/ardamavi/sign-language-digits-dataset and https://github.com/ardamavi/Sign-Language-Digits-Dataset.

Another dataset for the alphabets in the **American Sign Language** [83] which is available online at https://www.kaggle.com/grassknoted/asl-alphabet and https://github.com/SouravJain01/ASL_SIGN_PREDICTOR. The training dataset contains 87,000 images. Each image has a size of (200 × 200) pixels. There are 29 classes (26 for the letters from "A" to "Z" and 3 classes for "SPACE", "DELETE", and "NOTHING"). The test dataset contains a mere 29 images.

UCF-101 [60] is an action recognition dataset. It contains 13,320 realistic action YouTube videos. The number of its categories is 101. The UCF-101 categories can be divided into five different types: (i) Human-Object Interaction, (ii) Body-Motion Only, (iii) Human-Human Interaction, (iv) Playing Musical Instruments, and (v) Sports. Apply Eye Makeup, Apply Lipstick, Archery, Baby Crawling, Balance Beam, and Band Marching are examples of these categories. It is available online at https://www.crcv.ucf.edu/data/UCF101.php.

Shohieb et al. [59] developed a dataset for the Arabic sign language manual and non-manual signs named **SignsWorld Atlas**. Their captured postures, gestures, and motions were applied under different lighting and background conditions. Their dataset contained 500 elements and included (1) Arabic alphabets, (2) numbers from 0 to 9, (3) hand-shapes, (4) signs in isolation, (5) movement in continuous sentences, (6) lip movement for a set of Arabic sentences, and (7) facial expressions. Table 1 summarizes the existing and discussed datasets.

Table 1 Summary of the Existing Datasets

Work	Dataset Size	Language	Variety
Arabic Alphabets Sign Language Dataset (ArASL) [43]	54,049	Arabic	32 Standard Signs and Alphabets
Turkey Ankara Ayrancı Anadolu High School Students [82]	2,062	–	Digits from 0 to 9
American Sign Language [83]	87,000	English	29 Categories
UCF-101 [60]	13,320	–	101 Categories (5 Types)
SignsWorld Atlas [59]	500	Arabic	7 Types

Table 2 Signs with the Corresponding Count of each Video

#	Word (English)	Word (Arabic)	Count
1	Baby	طفل	430
2	Eat	يأكل	410
3	Father	اب	451
4	Finish	ينهي	440
5	Good	جيد	436
6	Happy	سعيد	445
7	Hear	يسمع	433
8	House	بيت	421
9	Important	مهم	446
10	Love	يحب	435
11	Mall	مول	414
12	Me	انا	430
13	Mosque	مسجد	427
14	Mother	ام	406
15	Normal	عادي	410
16	Sad	حزين	420
17	Stop	توقف	426
18	Thanks	شكرا	412
19	Thninking	يفكر	366
20	Worry	قلق	409
Total			8, 467

3.2 The proposed dataset

Creating such a dataset to fit the natural circumstances and environments is one of the main objectives of the current study. Based on statistics by 2020 [80, 81], almost everyone has his own smartphone with a camera. Following this concept, the dataset is created using smartphone videos. Videos are recorded natively using the authors' mobile phones without using any stabilization tool either hardware or software. Videos are captured with different resolutions and different locations, places, and backgrounds. 8,467 videos are recorded for 20 signs from 72 volunteers. The followed recording criteria is that each volunteer has to do each sign for at least 5 times (i.e., around 100 videos from each volunteer). The volunteers were males and females in an age range from 20 to 24. Table 2 shows each sign with the corresponding count of each video. Figure 2 summarizes the statistics of each word in the suggested dataset. The dataset calculated average (i.e., mean) is 423.35 and the standard deviation is 18.58. Figure 3 shows sample frames from each word in the proposed dataset.

4 Dataset pre-processing

In this section, the pre-processing stages made on the raw data are presented. As mentioned in the previous section, the proposed dataset videos were captured by mobile cameras, not a

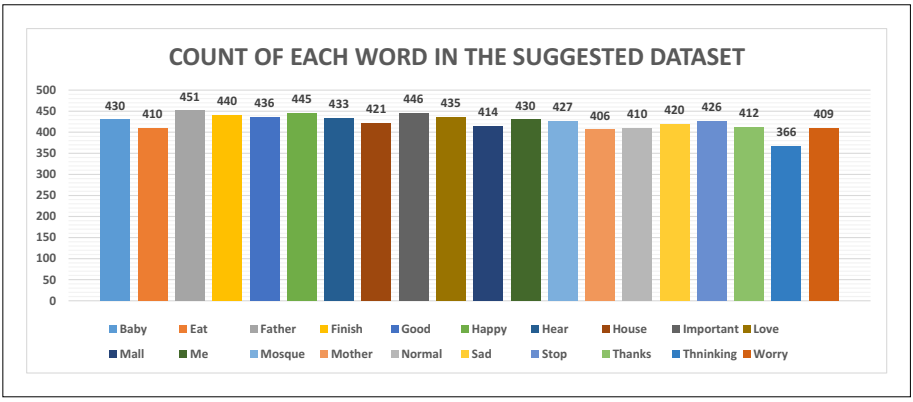

Fig. 2 Graphical Statistics of the Dataset Words

professional camera nor even a fixed camera; hence, the videos are affected by a noticeable amount of noise. By following the rules of feature selection [4, 41, 45], a suitable way should be found to extract just the necessary movement out from each frame, so the model can generalize on any signer under any circumstances [46]. Raw video passes through three stages before it could be used with the proposed model (discussed in Section 5).

First Stage: The first stage is to reduce each frame's dimensions and to convert the frames into grayscale. The benefits behind this stage are to (1) reduce the processing time and (2) achieve less overall complexity.

Second Stage: The output of the first stage is then passed to a difference function as shown in Fig. 4. The difference function subtracts every two consecutive frames to find the motion as shown in Equation (1). If the resultant frame was totally white or black, it is discarded. An adaptive threshold [38] is applied to the resultant frame. This approach will hold the most important information out of the frames. By applying this to the whole video's frames, $(n-1)$ frames will be retrieved, where n is the number of video frames.

Fig. 3 Sample Frames from each Word in the Proposed Dataset

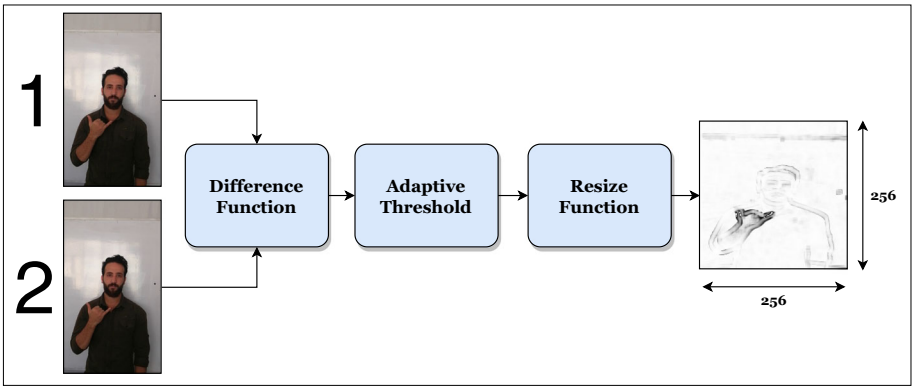

Fig. 4 A Sample Preview on the Second Pre-Processing Stage

The output single frame can be resized optionally using a resizing function. Figure 5 shows a sample preview after the pre-processing of the second stage on a sample video.

$$frame_{diff} = frame_i - frame_{(i-1)} \tag{1}$$

Third Stage: The third and last stage is about unifying each class's features and adding a unique factor to each class's videos. The output is only 30 frames out from $(n-1)$ frames where each unified frame combines (3×3) frames as shown in Fig. 6. These frames aren't selected randomly but instead, it is related to the index of the currently formed frame. The main purpose of the last stage is to reduce redundancy but without dropping any frame and keeping all information of all frames in the 30 frames. This can reduce conflicts between signs of similar movements' positions but with different operations sequences as these frames track the hands' positions through time.

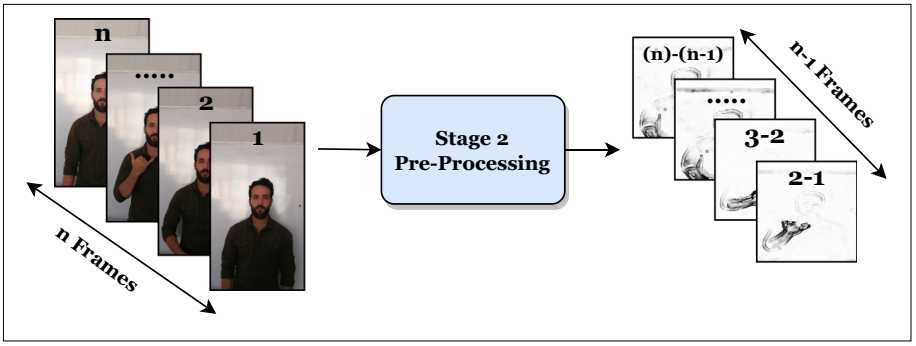

Fig. 5 Sample Preview after the Second Stage on a Sample Video

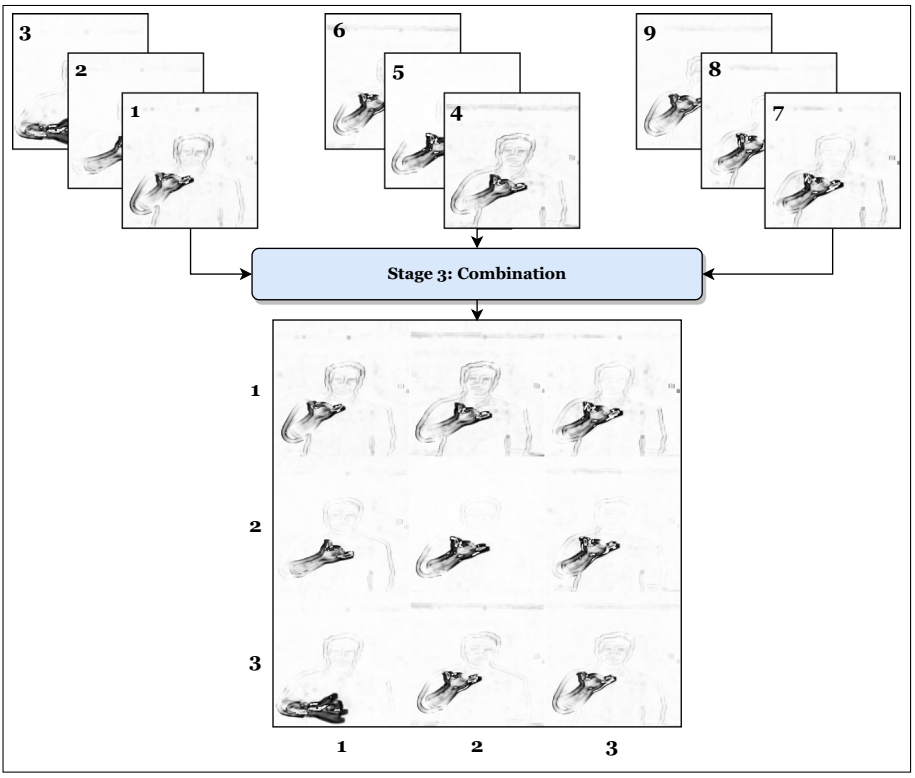

Fig. 6 Stage 3 Pre-Processing: Sample Preview

Algorithm 1 summarizes the three dataset pre-processing stages with their inner steps.

Algorithm 1 The Three Dataset Pre-processing Stages Pseudocode.

function PREPROCESS(video) \\ The pre-processing function. It accepts the video and returns the pre-processed and combined frames.
  frames ← **ExtractFrames**(video) \\ Extract the frames from the video.
  resizedFrames ← **Resize**(frames) \\ Resize the frames.
  grayFrames ← **GrayscaleConversion**(resizedFrames) \\ Convert the frames to grayscale.
  diffFrames ← **DifferenceFunction**(grayFrames) \\ Apply the difference function to every two consecutive frames.
  adptFrames ← **AdaptiveThreshold**(diffFrames) \\ Apply the adaptive threshold to the (n − 1) frames.
  combinedFrames ← **CombineRefine**(adptFrames) \\ Apply the combinations on the (n − 1) frames.
  return combinedFrames \\ Return the pre-processed and combined frames.

5 The proposed architecture

This paper contributes with an architecture for recognizing videos and classify them in the video classification field specifically sign language recognition. The main idea behind the proposed model (i.e., architecture) is to train two different CNN independently using the same architecture but on different portions of data. The input to it is the frames that are

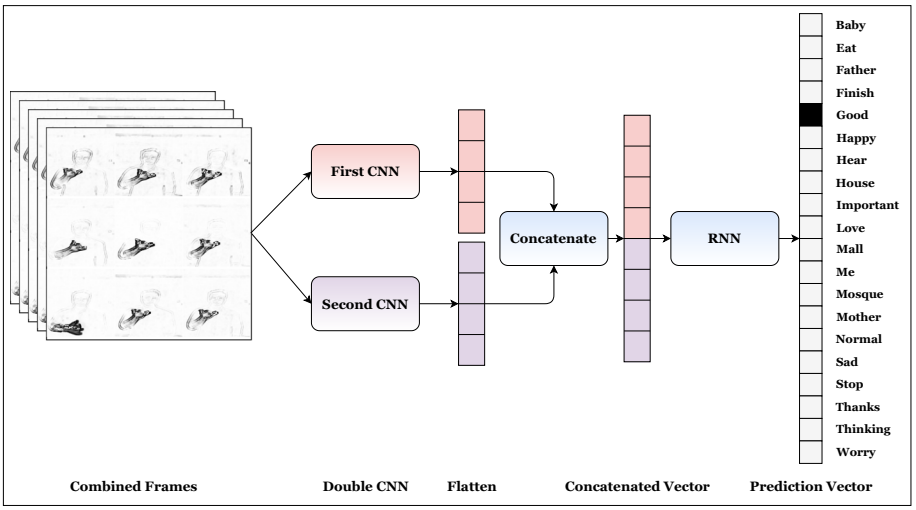

Fig. 7 Overview on the Proposed Architecture

pre-processed in the pre-processing phase. The output from each CNN is concatenated into one single vector with a size of (1×512). It is then passed to an RNN, which has a great ability to identify sequences in videos. RNN can learn from the changes over time in each sequence and be able to generalize it over the classes. The authors made the RNN sequence size be (30×512). This approach can help the network to identify different features for the same input and improves its overall confidence and accuracy. Figure 7 shows an overview of the suggested model.

5.1 Convolutional Neural Network (CNN)

The CNN is used to extract spatial features in the proposed architecture. Mainly, the convolutional layers [10, 57] are used to extract the features and detect different patterns in multiple sub-regions (i.e., kernels). The pooling layers [13, 64] are used to keep the most important features and progressively reduce the input spatial size to reduce the number of parameters and computation cost in the architecture and hence it can control the overfitting issue [9, 33]. There are different types of the pooling layers such as max-, min-, and average (i.e., mean) pooling layers [7]. The max-pooling and min-pooling layers take the maximum and minimum values from the previous layer respectively while the average layer takes the average. The max-pooling is a commonly used pooling type [8].

Figure 8 shows the building blocks of the used CNN architecture. The input layer accepts frames where each frame is sized $(128 \times 128 \times 3)$. After that, it has four "Conv-Pool-Drop" blocks, a global average pooling (GAP) layer [36], and a prediction network. Each "Conv-Pool-Drop" block of the first four blocks has two convolutional layers, one max-pooling layer, and followed by a dropout layer [6, 14] with a ratio of 0.5. The dropout layer is used to reduce the overfitting and increase its network's ability to generalize. All blocks almost have the same dimensions except for depth. They are as follows 128, 256, 512, and 256 respectively from the left to the right. The global average pooling layer is used to reduce the spatial dimensions. However, GAP layers apply a more extreme dimensionality reduction approach where the input is reduced in size to have dimensions of $(1 \times 1 \times d)$. They reduce

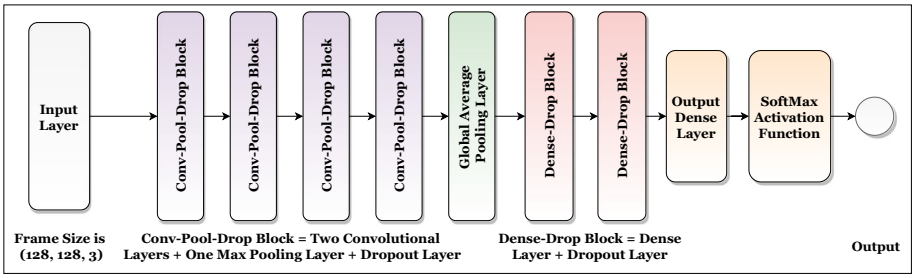

Fig. 8 The Building Blocks of the CNN Architecture

each feature map to a single value by simply applying the average of all feature map values [12, 29].

The prediction network is composed of two "Dense-Drop" blocks and one Fully-Connected (FC) layer [79]. It takes the output of the CNN network, flattens it (i.e., converts from multi-dimensions to a one-dimensional vector [8]), and uses it to classify the input to its class. The two "Dense-Drop" blocks contain a dense layer and a dropout layer. Each dense layer has 1024 neurons and each dropout layer has a dropout ratio of 0.2. The used activation function is Rectified Linear Unit (ReLU) [2, 11] in the hidden layers. ReLU is one of the common activation functions that returns 0 for negative inputs and the value itself for positive inputs. It is helpful for specific interaction effects and non-linearities. Equation (2) shown the used ReLU equation [32].

$$\mathrm{ReLU}(input) = \max(0, input) \tag{2}$$

The last FC layer contains 20 neurons with a SoftMax activation function [24]. The used batch size for the CNN network is 64. Table 3 shows the internal layers in detail. Figure 9 shows the internal structure of the "Conv-Pool-Drop" and "Dense-Drop" blocks.

5.2 Recurrent Neural Network (RNN)

The RNNs make use of the information in the sequence for the recognition tasks. Traditional RNNs suffer from vanishing gradients which caused them not to learn so much [25]. Long Short-Term Memory (LSTM) is a variant of RNN, which is designed to efficiently solve the vanishing and exploding gradients problems [30]. Bi-directional LSTMs (BiLSTMs) are an extension of traditional LSTMs which improve model performance on sequence classification problems [69]. BiLSTMs train two LSTMs instead of one LSTM in the input sequence, when all time steps of the input sequence are available. This can provide additional context to the network and result in faster and even fuller learning on the current task.

In the suggested RNN model, the output is combined from the two CNNs and is fed to five cascaded layers of 512 BiLSTM units. Every one of these layers is followed by a dropout layer with a dropout rate of 0.9 to avoid network overfitting. These layers are followed by an FC layer with a SoftMax activation function which is used to predict the output. Decreasing the number of BiLSTM layers with keeping the same number of BiLSTM units is experimented and using only 3 BiLSTM layers with 2048, 1024, 2048 units respectively is also experimented. We tested different configurations (try-and-error tries) on the suggested dataset and found that 5 BiLSTM layers with 512 hidden units performed the best. Figure 10 shows the building blocks of the used RNN architecture. The used activation function is ReLU [2] in the hidden layers. The used batch size for the RNN network is 64.

Table 3 The Internal In-Detail Blocks of the CNN Architecture

Layer	Number of Kernels	Kernel Size	Stride	Output size
Input	3	–	–	128 × 128 × 3
Convolutional 2D	128	5	1	124 × 124 × 128
Convolutional 2D	128	5	1	120 × 120 × 128
Max Pooling	128	3	2	59 × 59 × 128
Dropout	–	–	–	59 × 59 × 128
Convolutional 2D	256	5	1	55 × 55 × 256
Convolutional 2D	256	5	1	51 × 51 × 256
Max Pooling	256	2	2	25 × 25 × 256
Dropout	–	–	–	25 × 25 × 256
Convolutional 2D	512	3	1	23 × 23 × 512
Convolutional 2D	512	3	1	21 × 21 × 512
Max Pooling	512	2	2	10 × 10 × 512
Dropout	–	–	–	10 × 10 × 512
Convolutional 2D	256	3	1	8 × 8 × 256
Convolutional 2D	256	3	1	6 × 6 × 256
Max Pooling	256	3	3	2 × 2 × 256
Dropout	–	–	–	2 × 2 × 256
Global Average Pooling 2D	–	–	–	256
Fully Connected	–	–	–	1024
Dropout	–	–	–	1024
Fully Connected	–	–	–	1024
Dropout	–	–	–	1024
Fully Connected	–	–	–	20

To train the model, the Adaptive Moment (Adam) parameters optimizer technique is used [17]. It is an optimization algorithm that is used to update the network's weights (i.e., parameters) iterative based on the training instead of the classical stochastic gradient descent procedure [18]. Adam combines the heuristics of both the Momentum and the RMSProp

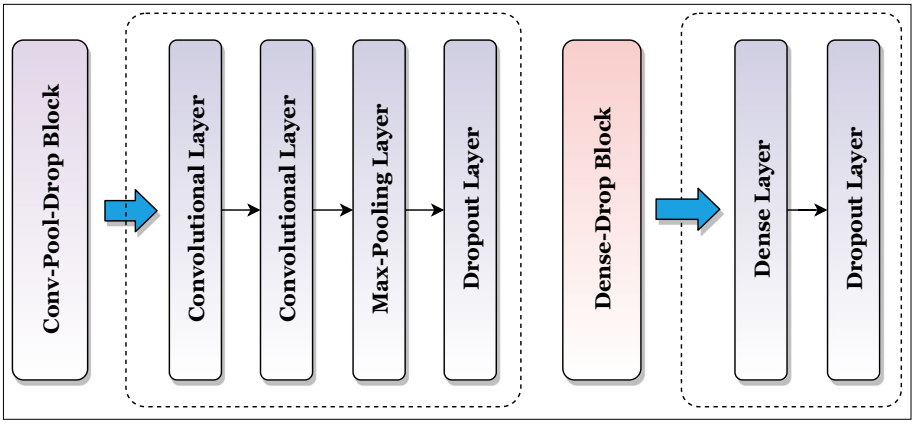

Fig. 9 The Internal Structure of the "Conv-Pool-Drop" and "Dense-Drop" Blocks

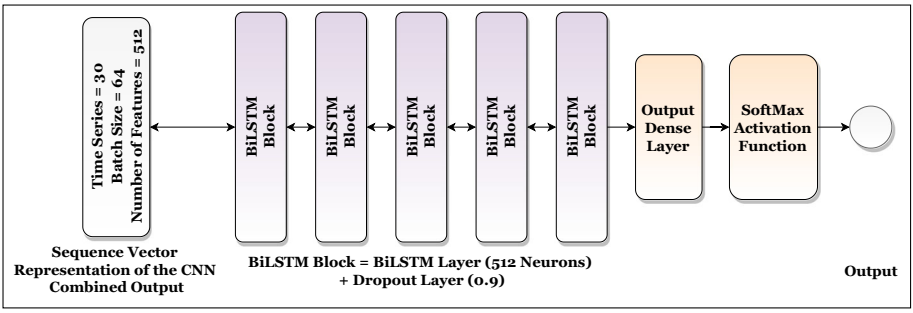

Fig. 10 The Building Blocks of the RNN Architecture

and hence has the advantage that it can handle spare gradients on noisy problems [78] as shown in Equation (3).

$$w_{t+1} = w_t - \eta \times \frac{v_t}{\sqrt{s_t + \epsilon}} \times g_t \tag{3}$$

where η is the initial learning rate (10^{-4} in the current study), g_t is the gradient at time t, v_t is the exponential average of gradients, s_t is the exponential average of square gradients, and ϵ is a very small value to avoid the division by zero (it can be 10^{-10}). Adam is used with a 10^{-6} decay rate.

6 Experimental results and discussion

In the first subsection, the experiments' configurations are presented. In the second subsection, Two types of experiments are performed. The first is performed on the suggested dataset while the second is applied to the UCF-101 dataset.

Table 4 summarizes the experiments configurations.

Table 4 Experiments Common Configurations Summarization

Configuration	Values
Dataset (X and Y)	A Suggested Dataset and UCF-101
Categories	20 for the Suggested Dataset and 101 for the UCF-101
Batch Size	64
Parameters Optimizer	Adam
Learning Rate	10^{-4}
Decay Rate	10^{-6}
"Dense-Drop" Dropout Ratio	0.2
RNN Dropout Ratio	0.9
Activation Function	ReLU
Number of Epochs	64
Performance Metrics	Accuracy, Loss, Confusion Metrics, and Top-1 Accuracy
Training Environment	Windows 10, GPU $4GB$, RAM $32GB$, and Intel Core i7 Processor

Table 5 Top-1 Accuracies of the Proposed Dataset

Class Name	Validation (%)	Test (%)
Baby	99%	97%
Eat	97%	93%
Father	100%	93%
Finish	97%	91%
Good	93%	79%
Happy	97%	98%
Hear	98%	78%
House	98%	99%
Important	100%	97%
Love	97%	96%
Mall	100%	98%
Me	95%	89%
Mosque	97%	98%
Mother	99%	84%
Normal	99%	92%
Sad	100%	97%
Stop	99%	95%
Thanks	97%	65%
Thinking	97%	72%
Worry	100%	100%
Average Accuracy	98%	92%
Standard Deviation	1.8%	9.61%

6.1 The suggested dataset experiments

The results of the proposed dataset are shown in Table 5 and presented graphically in Fig. 11. The top-1 accuracies are reported for each class on validation and test sets. By observing the results, it could be noticed that four classes have very low accuracies relative to other classes.

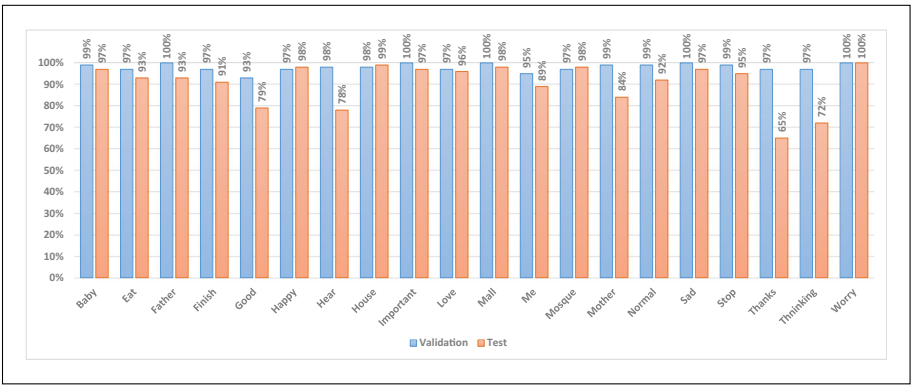

Fig. 11 Graphical Summarization of the Results of the Suggested Dataset

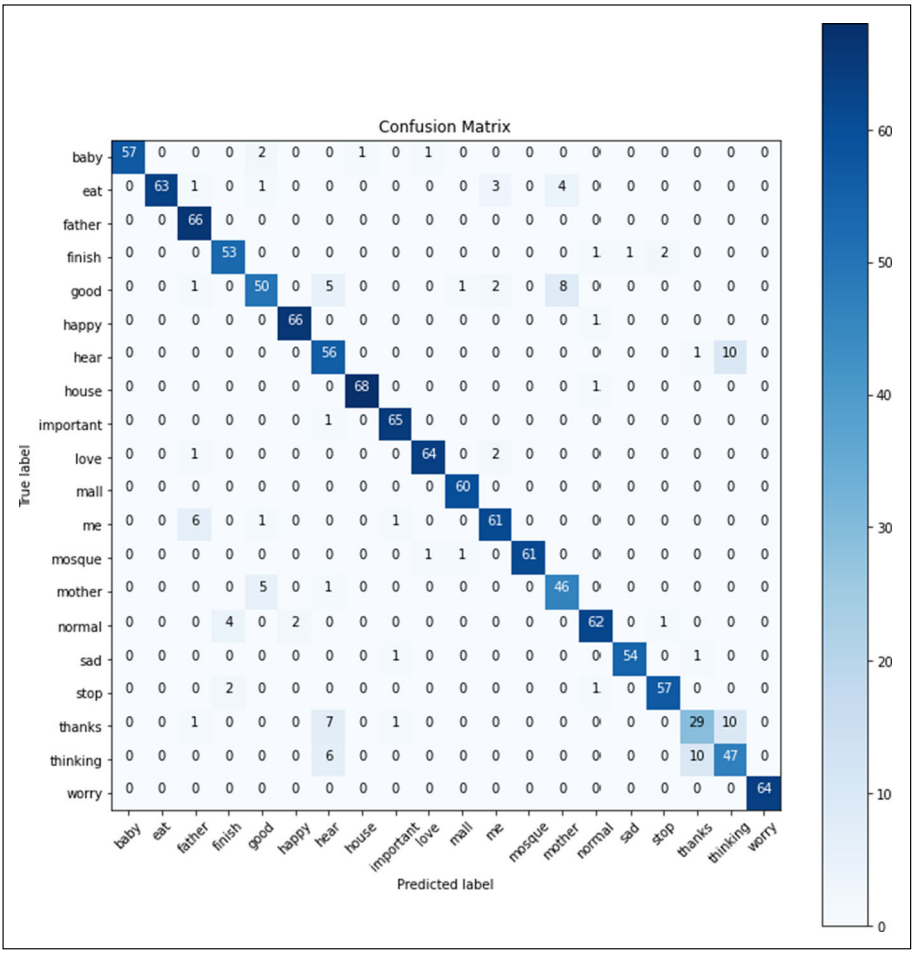

Fig. 12 The Confusion Matrix on the Test Data of the Suggested Dataset

They are "Good", "Hear", "Thanks", and "Thinking". The reason behind these results is that almost all of them are similar in the kinematic movement and the sign performance. That led to a conflict between these signs which in turn led to these low accuracies for these classes.

The conflicts can be clear also by observing the confusion matrix on test data shown in Fig. 12. As mentioned, the conflict between these few classes occurred due to the lack of experience of performers. Also, it seems that there is a great conflict between the classes "Thanks" and "Thinking" signs as they are almost similar and they need to be performed correctly and accurately to be recognized correctly.

6.2 The UCF-101 dataset experiments

To check the ability of our model to behave on other datasets and how it could generalize, we have applied preprocess stages and trained the model on the UCF-101 dataset. Table 6

Table 6 Top-1 and Top-5 Accuracies of the UCF-101 Dataset

Method	Backbone	Pretrained Weights	Top-1	Top-5
TSN-7seg [74]	InceptionV3 [66]	ImageNet [23]	73.9%	91.1%
TSM-8seg [44]	ResNet50 [34]	ImageNet [23]	72.8%	–
SlowOnly-8x8 [27]	ResNet101 [34]	–	75.9%	–
SlowFast-8x8 [27]	ResNet101 [34]	–	77.9%	93.2%
I3D-64x1 [20]	Inception [66]	ImageNet [23]	72.1%	90.3%
NL-128x1 [73]	ResNet101 [34]	ImageNet [23]	77.7%	93.3%
SlowOnly-8x8 [27]	ResNet101 [34]	ImageNet [23]	77.9%	93.2%
LGD-3D (RGB) [55]	ResNet101 [34]	ImageNet [23]	79.4%	94.4%
STDFB [48]	ResNet152 [34]	ImageNet [23]	78.8%	93.6%
irCSN-32x2 [28]	irCSN-152 [28]	IG-65M [28]	82.6%	95.3%
Proposed Model	–	–	93.4%	98.8%

shows a comparison between the different models and the proposed model. The presented results are reported after performing cross-validation using 3-folds [19].

As the table shows, our proposed model has achieved state-of-the-art results on the UCF-101 dataset. The current study reported accuracies that were better than 10 previous studies on the UCF-101 dataset. These results confirm that the proposed model can be used on different action recognition datasets not only the sign language datasets including the suggested one.

7 Conclusions and future work

In this paper, we proposed an Arabic sign language dataset with 8,467 videos of 20 signs for different volunteers. The captured videos did not require any tools but just a mobile phone. Also, we suggested a new approach (i.e., architecture) for video classification and recognition using a combination of CNN and RNN besides the pre-processing performed on the captured videos. We used double CNNs as feature extractors out from videos' frames and concatenate these features together as a sequence. RNN was used to identify the relationship between the sequences and produce the overall prediction. Concerning that approach, we reached state-of-the-art results as we achieved 98% and 92% on the validation and testing subsets respectively on the suggested dataset. The suggested approach also achieved very promising accuracies on the UCF-101 dataset. They were 93.40% and 98.80% on top-1 and top-5 respectively.

In the future, we can enlarge the suggested dataset with new signs and more users. We can shed the light on phrases not just words. We can also modify the proposed model to adapt to the new videos with the ability to implement grammatically right phrases. Different architectures, approaches, networks, methods can be used also. More experiments can be conducted on other Arabic datasets.

Acknowledgements We would like to express gratitude and appreciation to the volunteers who decided to cooperate in the dataset construction.

Funding Open access funding provided by The Science, Technology & Innovation Funding Authority (STDF) in cooperation with The Egyptian Knowledge Bank (EKB). No funding was received for this work.

Declarations

Conflict of Interests No conflict of interest exists. We wish to confirm that, there are no known conflicts of interest associated with this publication and there has been no significant financial support for this work that could have influenced its outcome.

References

1. Abdulazeem Y, Balaha HM, Bahgat WM, Badawy M (2021) Human action recognition based on transfer learning approach. IEEE Access 9:82058–82069
2. Agarap AF (2018) Deep learning using rectified linear units (relu). arXiv:1803.08375
3. Al-Hammadi M et al (2020) Hand gesture recognition for sign language using 3dcnn. IEEE Access 8:79491–79509
4. Al-Tashi Q, Abdulkadir SJ, Rais HM, Mirjalili S, Alhussian H (2020) Approaches to multi-objective feature selection: a systematic literature review. IEEE Access 8:125076–125096
5. Albawi S, Mohammed TA, Al-Zawi S (2017) Understanding of a convolutional neural network. In: 2017 International conference on engineering and technology (ICET). IEEE, pp 1–6
6. Bahgat WM, Balaha HM, AbdulAzeem Y, Badawy MM (2021) An optimized transfer learning-based approach for automatic diagnosis of covid-19 from chest x-ray images. PeerJ Comput Sci 7:e555
7. Balaha HM, Ali HA, Badawy M (2021) Automatic recognition of handwritten arabic characters: a comprehensive review. Neural Comput Applic 33(7):3011–3034
8. Balaha HM, Ali HA, Saraya M, Badawy M (2021) A new arabic handwritten character recognition deep learning system (ahcr-dls). Neural Comput Applic 33(11):6325–6367
9. Balaha HM, Balaha MH, Ali HA (2021) Hybrid covid-19 segmentation and recognition framework (hmb-hcf) using deep learning and genetic algorithms. Artif Intell Med 119:102156
10. Balaha HM, El-Gendy EM, Saafan MM (2021) Covh2sd: a covid-19 detection approach based on harris hawks optimization and stacked deep learning. Expert Syst Appl 186:115805
11. Balaha HM, El-Gendy EM, Saafan MM (2022) A complete framework for accurate recognition and prognosis of covid-19 patients based on deep transfer learning and feature classification approach. Artif Intell Rev, 1–46
12. Balaha HM, Saif M, Tamer A, Abdelhay EH (2022) Hybrid deep learning and genetic algorithms approach (hmb-dlgaha) for the early ultrasound diagnoses of breast cancer. Neural Comput Applic, 1–25
13. Balaha HM et al (2021) Recognizing arabic handwritten characters using deep learning and genetic algorithms. Multimed Tools Appl 80(21):32473–32509
14. Baldi P, Sadowski PJ (2013) Understanding dropout. Adv Neural Inf Process Syst 26:2814–2822
15. Beal MJ, Ghahramani Z, Rasmussen CE (2002) The infinite hidden markov model. In: Advances in neural information processing systems, pp 577–584
16. Bheda V, Radpour D (2017) Using deep convolutional networks for gesture recognition in american sign language. arXiv:1710.06836
17. Bock S, Goppold J, Weiß M (2018) An improvement of the convergence proof of the adam-optimizer. arXiv:1804.10587
18. Bock S, Weiß M (2019) A proof of local convergence for the adam optimizer. In: 2019 International joint conference on neural networks (IJCNN). IEEE, pp 1–8
19. Browne MW (2000) Cross-validation methods. J Math Psychol 44(1):108–132
20. Carreira J, Zisserman A (2017) Quo vadis, action recognition? A new model and the kinetics dataset. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)
21. Cheok MJ, Omar Z, Jaward MH (2019) A review of hand gesture and sign language recognition techniques. Int J Mach Learn Cybern 10(1):131–153

22. Cooper H, Holt B, Bowden R (2011) Sign language recognition in visual analysis of humans. Springer, pp 539–562
23. Deng J et al (2009) Imagenet: a large-scale hierarchical image database. In: 2009 IEEE Conference on computer vision and pattern recognition. IEEE, pp 248–255
24. Dunne RA, Campbell NA (1997) On the pairing of the softmax activation and cross-entropy penalty functions and the derivation of the softmax activation function. In: Proc. 8th Aust. conf. on the neural networks, vol 181. Citeseer, Melbourne, p 185
25. ElSaid A, Wild B, Higgins J, Desell T (2016) Using lstm recurrent neural networks to predict excess vibration events in aircraft engines. In: 2016 IEEE 12th International conference on e-science (e-science). IEEE, pp 260–269
26. Er-Rady A, Faizi R, Thami ROH, Housni H (2017) Automatic sign language recognition: a survey in 2017 International Conference on Advanced Technologies for Signal and Image Processing (ATSIP). IEEE, pp. 1–7
27. Feichtenhofer C, Fan H, Malik J, He K (2019) Slowfast networks for video recognition. In: Proceedings of the IEEE international conference on computer vision, pp 6202–6211
28. Ghadiyaram D et al (2019) Large-scale weakly-supervised pre-training for video action recognition. CoRR abs/1905.00561
29. Gong W, Chen H, Zhang Z, Zhang M, Gao H (2020) A data-driven-based fault diagnosis approach for electrical power dc-dc inverter by using modified convolutional neural network with global average pooling and 2-d feature image. IEEE Access 8:73677–73697
30. Graves A (2012) Long short-term memory in Supervised sequence labelling with recurrent neural networks. Springer, pp 37–45
31. Grobel K, Assan M (1997) Isolated sign language recognition using hidden markov models. In: 1997 IEEE International conference on systems, man, and cybernetics. Computational cybernetics and simulation, vol 1. IEEE, pp 162–167
32. Hara K, Saito D, Shouno H (2015) Analysis of function of rectified linear unit used in deep learning. In: 2015 International joint conference on neural networks (IJCNN). IEEE, pp 1–8
33. Hawkins DM (2004) The problem of overfitting. J Chem Inf Comput Sci 44(1):1–12
34. He K, Zhang X, Ren S, Sun J (2015) Deep residual learning for image recognition. CoRR abs/1512.03385
35. Hienz H, Bauer B, Kraiss KF (1999) Hmm-based continuous sign language recognition using stochastic grammars. In: International gesture workshop. Springer, pp 185–196
36. Hsiao TY, Chang YC, Chou HH, Chiu CT (2019) Filter-based deep-compression with global average pooling for convolutional networks. J Syst Archit 95:9–18
37. Huang J, Zhou W, Li H, Li W (2015) Sign language recognition using 3d convolutional neural networks. In: 2015 IEEE international conference on multimedia and expo (ICME). IEEE, pp 1–6
38. Jie G, Ning L (2012) An improved adaptive threshold canny edge detection algorithm. In: 2012 International conference on computer science and electronics engineering, vol 1. IEEE, pp 164–168
39. Johnston T, Schembri A (2007) Australian sign language (Auslan): an introduction to sign language linguistics. Cambridge University Press, Cambridge
40. Keskin C, Kıraç F, Kara YE, Akarun L (2013) Real time hand pose estimation using depth sensors in consumer depth cameras for computer vision. Springer, pp 119–137
41. Kira K, Rendell LA (1992) A practical approach to feature selection in Machine learning proceedings 1992. Elsevier, pp 249–256
42. Koller O, Zargaran O, Ney H, Bowden R (2016) Deep sign: hybrid cnn-hmm for continuous sign language recognition. In: Proceedings of the British machine vision conference 2016
43. Latif G, Mohammad N, Alghazo J, AlKhalaf R, AlKhalaf R (2019) Arasl: Arabic alphabets sign language dataset. Data Br 23:103777
44. Lin J, Gan C, Han S (2019) Tsm: temporal shift module for efficient video understanding. In: Proceedings of the IEEE international conference on computer vision, pp 7083–7093
45. Liu S, He T, Dai J (2021) A survey of crf algorithm based knowledge extraction of elementary mathematics in Chinese. Mobile Netw Applic, 1–13
46. Liu S, Wang S, Liu X, Lin CT, Lv Z (2020) Fuzzy detection aided real-time and robust visual tracking under complex environments. IEEE Trans Fuzzy Syst 29(1):90–102
47. López-Noriega JE, Fernández-Valladares MI, Uc-Cetina V (2014) Glove-based sign language recognition solution to assist communication for deaf users. In: 2014 11th International conference on electrical engineering, computing science and automatic control (CCE). IEEE, pp 1–6
48. Martinez B, Modolo D, Xiong Y, Tighe J (2019) Action recognition with spatial-temporal discriminative filter banks. In: Proceedings of the IEEE international conference on computer vision, pp 5482–5491

49. Masood S, Thuwal HC, Srivastava A (2018) American sign language character recognition using convolution neural network in Smart Computing and Informatics. Springer, pp 403–412
50. Medsker LR, Jain L (2001) Recurrent neural networks. Design and Applications, 5
51. Mehdi SA, Khan YN (2002) Sign language recognition using sensor gloves. In: Proceedings of the 9th international conference on neural information processing, 2002. ICONIP'02, vol 5. IEEE, pp 2204–2206
52. Nandy A, Prasad JS, Mondal S, Chakraborty P, Nandi GC (2010) Recognition of isolated indian sign language gesture in real time. In: International conference on business administration and information processing. Springer, pp 102–107
53. O'Shea K, Nash R (2015) An introduction to convolutional neural networks. arXiv:1511.08458
54. Parcheta Z, Martínez-Hinarejos CD (2017) Sign language gesture recognition using hmm. In: Iberian conference on pattern recognition and image analysis. Springer, pp 419–426
55. Qiu Z, Yao T, Ngo CW, Tian X, Mei T (2019) Learning spatio-temporal representation with local and global diffusion. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 12056–12065
56. Rastgoo R, Kiani K, Escalera S (2020) Sign language recognition: a deep survey. Expert Systems with Applications, 113794
57. Sainath TN, Mohamed Ar, Kingsbury B, Ramabhadran B (2013) Deep convolutional neural networks for lvcsr. In: 2013 IEEE international conference on acoustics, speech and signal processing. IEEE, pp 8614–8618
58. Sandler W, Lillo-Martin D (2006) Sign language and linguistic universals. Cambridge University Press, Cambridge
59. Shohieb SM, Elminir HK, Riad A (2015) Signsworld atlas; a benchmark arabic sign language database. J King Saud Univ - Comput Inf Sci 27(1):68–76
60. Soomro K, Zamir AR, Shah M (2012) UCF101: a dataset of 101 human actions classes from videos in the wild. CoRR abs/1212.0402
61. Starner TE (1995) Visual recognition of american sign language using hidden markov models. Massachusetts Inst Of Tech Cambridge Dept Of Brain And Cognitive Sciences. Technical report
62. Starner T, Pentland A (1997) Real-time american sign language recognition from video using hidden markov models in Motion-based recognition. Springer, pp 227–243
63. Starner T, Weaver J, Pentland A (1998) Real-time american sign language recognition using desk and wearable computer based video. IEEE Trans Pattern Anal Mach Intell 20(12):1371–1375
64. Sun M, Song Z, Jiang X, Pan J, Pang Y (2017) Learning pooling for convolutional neural network. Neurocomputing 224:96–104
65. Sutton-Spence R, Woll B (1999) The linguistics of British sign language: an introduction. Cambridge University Press, Cambridge
66. Szegedy C, Vanhoucke V, Ioffe S, Shlens J, Wojna Z (2015) Rethinking the inception architecture for computer vision. CoRR abs/1512.00567
67. Tamura S, Kawasaki S (1988) Recognition of sign language motion images. Pattern Recogn 21(4):343–353
68. Tao W, Leu MC, Yin Z (2018) American sign language alphabet recognition using convolutional neural networks with multiview augmentation and inference fusion. Eng Appl Artif Intell 76:202–213
69. Ullah A, Ahmad J, Muhammad K, Sajjad M, Baik SW (2017) Action recognition in video sequences using deep bi-directional lstm with cnn features. IEEE Access 6:1155–1166
70. Upendran S, Thamizharasi A (2014) American sign language interpreter system for deaf and dumb individuals. In: International conference on control, instrumentation, communication and computational technologies (ICCICCT). IEEE, pp 1477–1481
71. Valli C (2000) Lucas, C. Gallaudet University Press, Washington
72. Wadhawan A, Kumar P (2020) Deep learning-based sign language recognition system for static signs. Neural Comput Applic, 1–12
73. Wang X, Girshick R, Gupta A, He K (2018) Non-local neural networks. In: proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)
74. Wang G, Lai J, Huang P, Xie X (2019) Spatial-temporal person re-identification. In: Proceedings of the AAAI conference on artificial intelligence, vol 33, pp 8933–8940
75. Yang S, Zhu Q (2017) Continuous chinese sign language recognition with cnn-lstm. In: Ninth international conference on digital image processing (ICDIP 2017). (International Society for Optics and Photonics), vol 10420, p 104200F
76. Yegnanarayana B (2009) Artificial neural networks. (PHI Learning Pvt. Ltd.)
77. Youssif A, Aboutabl AE, Ali HH (2011) Arabic sign language (arsl) recognition system using hmm. International Journal of Advanced Computer Science and Applications (IJACSA) 2(11)

78. Zhang Z (2018) Improved adam optimizer for deep neural networks. In: 2018 IEEE/ACM 26th international symposium on quality of service (IWQoS). IEEE, pp 1–2
79. Zhang Q, Liang D (2020) Visualization of fully connected layer weights in deep learning ct reconstruction. arXiv:2002.06788
80. 39+ smartphone statistics you should know in 2020 (https://review42.com/smartphone-statistics). Accessed 25 December 2020
81. Number of smartphone users worldwide from 2016 to 2021 (https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide). Accessed 25 December 2020
82. Turkey ankara ayrancı anadolu high school's sign language digits dataset (https://www.kaggle.com/ardamavi/sign-language-digits-dataset). Accessed 25 December 2020
83. Dataset for the alphabets in the american sign language (https://www.kaggle.com/grassknoted/asl-alphabet). Accessed 25 December 2020