



Enhancing Visual Speech Recognition for Deaf Individuals: A Hybrid LSTM and CNN 3D Model for Improved Accuracy

R. Shashidhar¹ · M. P. Shashank¹ · B. Sahana^{1,2}

Received: 2 June 2023 / Accepted: 5 October 2023 / Published online: 17 November 2023
© King Fahd University of Petroleum & Minerals 2023

Abstract

The ability of a person to communicate with other people and engage with the outside environment makes it crucial to a person's existence. It can be challenging for all the people who have hearing loss to communicate without the aid of lip reading, facial expression, or signals. Also, without instruction, it is challenging for them to comprehend lip movements, which makes it challenging for them to hear spoken words. The visual speech recognition devices for lip reading were developed to solve this issue and help the deaf understand speech. Recently, Visual Voice Recognition research has focused a lot on deep learning advancements. Here, we suggested the LSTM and CNN 3D hybrid model for MIRACL VC1 Dataset visual speech recognition. When a person's lip features are extracted for pattern detection, the text without audio is converted in this process. A model used to automatically recognize speech is called visual speech recognition. The proposed work stands apart from the previous work by incorporating the hybrid model, the combination of the models would yield a better and efficient accuracy rate comparatively and also the focal novelty of the proposed work was that the concatenation of the extracted features to provide an input to the system was a challenging one so has to manage the weights in the entire process of extraction for a huge number of dataset. It was suggested that two separate models, CNN 3D and LSTM, be combined to create a hybrid model in mandate to progress the overall accuracy in this case. In comparison with the individual CNN 3D model, whose testing accuracy is 79%, and the LSTM model, whose accuracy was 85% when we employed the English Dataset, the hybrid model's training, test, and validation accuracy were all 98, 85, and 86%, respectively.

Keywords Visual speech recognition · CNN 3D · LSTM · Hybrid model · Deep learning

1 Introduction

Visual speech recognition is the method that aids speech recognition in recognizing non-deterministic speech by using

the image processing capabilities of lip reading. Separately and methodically, the systems of speech recognition and lip reading are carried out, and at the level of feature fusion, the outcomes of each are combined. One of the key sources of information for face-to-face speech prediction is visual articulation. For a sizable portion of the hearing handicapped, determined is based on lip reading. When electrical signals are added to the lip-reading, it enables quick and effective comprehension of speech in well-trained people. Speech perception and prediction heavily rely on visual information.

The novelty of the work is the proposed work distinguishes itself from previous research by introducing a hybrid model that combines multiple models. This combination is expected to result in a significantly improved and efficient accuracy rate compared to existing approaches. A key novelty of the proposed work lies in the challenging task of concatenating the extracted features to create an input for the system. Managing the weights throughout the feature extraction process,

M. P. Shashank and B. Sahana these authors have contributed equally to this work.

✉ R. Shashidhar
shashidhar.r@sjce.ac.in

M. P. Shashank
shashisam124@gmail.com

B. Sahana
sahanab@rvce.ac.in

¹ Department of Electronics and Communication Engineering, JSS Science and Technology University, Mysuru, Karnataka 570006, India

² Department of Electronics and Communication Engineering, RV College of Engineering, Bengaluru, Karnataka 560059, India



especially for a large dataset, was a significant challenge that the researchers had to address.

Recognition of visual speech suggests to the detailed extraction of feature-based analysis on the lips and its surrounding region. The various aspects of feature extraction as to present since the external environment comes into picture and each and every bit in an image shows a significant role in speech prediction with the help of the visuals. This method comprises of many complex steps that are used for the resolve of feature extraction and image processing because of the presence of a large database. Which is used to comprehend the speech by perceiving the movement of the mouths, face and added societal hints. Listening to the voice of a speaker is challenging in noisy atmospheres and Visual speech recognition model helps in creating technologies for the same. The visual signal also provides supplementary and redundant information, even if the audio signal carries the majority of the information. Even in a noisy setting, the visual evidence will progress speech recognition accuracy greatly if the acoustic signal is unaffected.

Visual speech recognition is the method of identifying speech without the use of audio with the use of video. Deep learning or machine learning methods can be used to construct VSR representations. To achieve the maximum accuracy possible, we are using hybrid algorithms in this case.

Region of interest which is the lip is extracted from each instance's frames. To put it another way, lip reading is involved in extracting the region of interest. The data is divided into training, testing, and validating groups once the data pre-processing has been successfully completed. Typically, greater than 70% of the data used for training proposed system, the next 20% are used for testing the system, and the remaining 10% used for the purpose of validation.

Data are divided into three groups, and then the processed data are trained into a specific model that fits the data. In this case, a hybrid model made of two models is being used. The accuracy of each model is then noted after the data has been trained independently for two different models. Then, a hybrid model is created by integrating the two distinct simulations in mandate to advance correctness.

2 Related Work

This section presents a summary of earlier studies conducted in the domains of visual speech recognition and audiovisual speech recognition. It also explores different datasets, approaches for feature extraction, and strategies for classification that have been employed in these areas.

A model for AVSR Using Recurrent Neural Network Transducers was put up by Takaki Makino et al. Large-scale AVSR system built on a RNN transducer architecture

is demonstrated in this paper. A sizable audio-visual dataset is created of which segmented utterances taken from public YouTube films, resulting in 31k hours of audiovisual training content, in order to enable the creation of such a system [1]. Shankar et al. introduced a Deep Recurrent Neural Networks for AVSR, a deep RNN training method for automatic speech detection in audio and video. A deep RNN audio model is used for training. To train a fusion RNN, audio and visual information are combined. [2] Speech recognition system using the RNN method was anticipated by Hori et al. which examines the effectiveness of contemporary automatic voice recognition models. In the Wall Street Journal task, the multi-level technique significantly reduces error; however, because two separate LMs must be trained, the cost and memory use are increased. The proposed model achieves the lowest word error rate reported for automatic speech recognition systems on this benchmark. Specifically, it achieves a word error rate of 5.1% on the WSJ Eval 92 test set when the vocabulary capacity is increased twofold [3].

Deep learning model for the purpose visual speech recognition was put forth by Navin Kumar Mudaliar et al. Deep learning has been employed in the proposed visual speech recognition method to classify words at the word level. 3D convolutional layers are utilized as the decoder in the ResNet architecture. The entire video sequence served as the input for this method. The outcomes of the suggested strategy are acceptable. On the BBC data set, it obtains 90 percentage accuracy, and on the custom video data set, 88% [4]. A brand-new technique mentioned as a Class Specific Maximization of Mutual Information was put forward by Naeem Ahmed et al. a condensed dictionary for each class. The CSMMI captures the unique items for a particular class various actions. The idea of an accomplishment and motion detection method can be applied in a variety of contexts, including sign language recognition, facial gesture control, and eye tracking [5].

End to end VSR is a topic that Zuwei Li et al. addressed. This study shows light on an end-to-end LongShort Memory (LSTM) network-based visual speech recognition system which simultaneously acquires to do classification. LSTM is used to simulate the sequential crescendos in every stream [6].

Stavros Petridis et al provides a modern performance in visual speech classification while concurrently learning to get features from pixels and doing classification. An LSTM (Long Short-Term Memory) model is utilized to capture the temporal dynamics within each stream [7].

Large-scale visual speech recognition was suggested by Brendan Shillingford et al. This research offers a remedy for visual speech recognition with open vocabulary [8]. Greg Pottie and his colleagues proposed a method that compared the data requirements for training supervised and unsupervised hidden Markov models with those of LSTM. Their

study concluded that even without a substantial amount of labeled data, an unsupervised model can outperform other models, highlighting the effectiveness of this approach [9].

Audiovisual Speech Recognition System converging an End-to-End with Multitask Learning was the topic of a study by Fei Tao et al. This study suggests an innovative end-to-end ASR multitask learning system. They were able to create an audiovisual system that is reliable and accurate and is condition-neutral. The performance is improved of the AV-ASR by identifying segments. [10].

Abhinav Thanda and his team developed an Audio-Visual Speech Recognition (AVSR) model using Deep Recurrent Neural Networks (RNNs). They proposed a deep RNN training algorithm for the AVSR system [11]. The study's results indicated that incorporating the visual modality substantially decreased character mistake rates at varying noise levels, particularly when trained without noisy data. The researchers also compared feature fusion and decision fusion techniques and achieved an overall efficiency of approximately 90% with their model [12].

Weijiang Feng et al. They found that while existing deep learning models are successful at integrating visual data into speech recognition, none of them simultaneously take the properties of both audio and visual modalities into account. A multimodal RNN system with three parts, namely an audio part, a visual part, and a fusion portion, is offered as a solution to this problem. [7]. RNN based AVSR System was proposed by Kai-Xian Lau et al. The RNN-based AVSR system improves speech recognition performance accuracy in a noisy environment by using additional sensory modalities to offset damaged audio. To extract speech features, the proposed RNN-based AVSR system makes advantage of MFCC. Image and audio data are obtained from ANN and unidirectional multimodal LSTM, respectively [13, 14].

Marina Zimmermann and her team conducted a study in which image patches extracted from video data were processed using principal component analysis (PCA) to calculate weights for a two-stage convolutional network. The outcomes of the study exposed that their proposed approach outpaced baseline techniques for phrase recognition in the OuluVS2 audiovisual database. When utilizing frontal view cross-validation, the model achieved sentence accuracy rates of 79% and 73%, surpassing the performance of previous methods [15].

A Multimodal Learning by the use of 3D Audio-Visual Data for AVSR was proposed by Rongfeng Suet al. The main problem is that the majority of systems rely on 2D AV corpora with a low video sampling rate. This research presents a solution to address the problem by introducing a new 3D audiovisual (AV) dataset with a higher video sample rate. The challenge of requiring both auditory and visual senses during system testing is also addressed. To create an

audiovisual speech recognition (AVSR) system with wider applicability, the study proposes a bimodal Convolutional Neural Network context that focuses on video feature generation [16].

The publication by Sonali et al illustrates the fundamental features and operation of ANNs. A highly interconnected set of processing components known as an ANN works together to tackle a particular problem, like pattern recognition or data classification, through a learning process. The infrastructure of neural networks is largely the same. Before to training, initial weights in the output layer are selected at random, and training then starts. In a lot less time, neural networks construct models that are more accurate representations of the data's structure [17].

Audio-Visual Speech Recognition by Kuniaki Noda et al involves a HMM system for AVSR is presented in this paper. Initially, noise-resistant audio features are acquired using a deep denoising autoencoder. Visual features from unprocessed photos of the mouth area are extracted using a convolutional neural network (CNN). Under 10dB of SNR ratio, denoised MFCCs can achieve a boost in word recognition rate of about 65% [18]. An article was published by Satoshi Tamur et al. This study with the help of visual features, applies AVSR features to enhance AVSR performance. With the aid of deep learning technology, numerous visual feature types are incorporated and then transformed into bottleneck features. They were able to read lips with accuracy of 73.66% employing the suggested features [19].

The article was published by Yuki Takashima et al. for a client severe hearing loss-related articulation issue, they suggested an AVSR. The speech pattern of a person with this kind of articulation impairment differs so significantly from that of a person who does not have hearing distance that a speaker independent acoustic prototypical for people with normal hearing is rarely helpful in identifying it. Despite the fact that effective feature integration is crucial for multimodal speech recognition, it is challenging due to the intrinsic differences between the features [20].

Fayek et al. introduced a real-time Speech Emotion Recognition (SER) system based on Deep Neural Networks (DNN) showcasing promising outcomes in terms of performance and efficiency [21]. Deep AVSR was the idea put out by Andrew Zisserman et al with the main objective to distinguish individual speakers from simultaneous multi-talker speech in movies. A deep audio-visual speech augmentation network that can distinguish a speaker's voice from lip regions in the accompanying video is proposed. Strong qualitative and quantitative performance is shown when the performance is examined for up to five simultaneous voices. Using a spatiotemporal residual network, a word-level lip reading challenge, visual features are retrieved from the input image frame sequence [22].

Deep learning-based audio-visual speech recognition was proposed by Yuki Yamaguchi et al. the datasets were used which contain noise as well as noise-corrupted data. To start, turn off the noise a noisy deep denoising auto encoder. After preparing the data for the model, robust audio features are created. The model is then trained to produce denoised audio features from matching noisy features. A Convolutional Neural Network (CNN) is utilized to capture visual features specifically from the mouth area of the image [18].

Oriol Vinyals et al. proposed the sequence-to-sequence loss—are primarily contrasted. They argued that their model will outperform all of the earlier attempts in the same [22]. Navdeep Jaitly et al. suggested the voice recognition technique presented in this research which immediately converts audio data to text without the need for phonotonic representation [24]. An application to identify, understand, and carry out speech commands across various applications and programs was proposed by Pooja et al. The complexity of providing voice support increases in the absence of a system where spoken commands to a computer are standardized, or at least restricted to a dialect. Vocal Aid and Control (VAAC) was created as a solution to this issue. By integrating it into all computing settings, the system seeks to provide adequate support to complement actions made using other input devices and improve device accessibility [25].

To enhance a noisy speech signal and obtain a clean speech waveform, it is often necessary to perform a transposal from the Mel-frequency field to the spectral field. However, this process can introduce falsification artifacts in the approximation and filtering of the spectrum. This is how the proposed speech enhancement method employs a Hidden Markov Model (HMM) framework combined with the Minimum Mean Square Error (MMSE) approach in the Mel-frequency domain. These parameters are estimated using a technique called Vector Taylor Series (VTS), which helps improve the estimation accuracy and ultimately enhance the speech quality [27].

shashidhar and team worked on the audiovisual speech recognition [28, 31] and visual speech recognition [32–33,35] using custom dataset for Indian English Language and Kannada Language using different machine learning and deep learning approaches. Saleh et al. worked on the traffic cognition and detection framework using deep learning [33].

3 Methodology

This chapter provides an overview of the system architecture and technical implementations involved in the development of Visual Speech Recognition. It outlines the various steps undertaken during the project's development process.

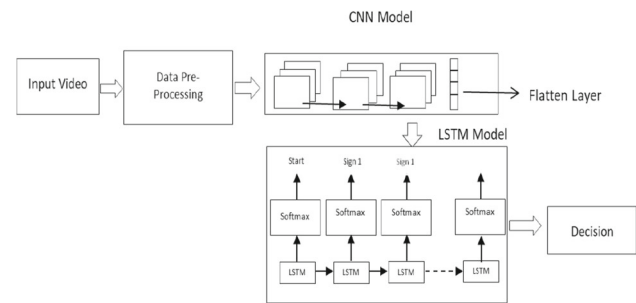


Fig. 1 Proposed flow Diagram

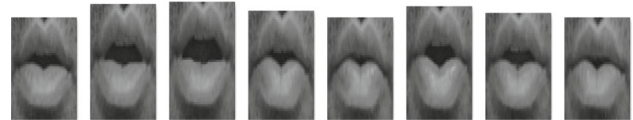


Fig. 2 Gray scale images of Lip localization

3.1 Block Diagram

There are three steps in the part of conduction. Figure 1 consists of block diagram which consists of all the methodologies is shown further

- (1) Pre-processing
- (2) Visual models
- (3) Hybridization of models

First we will get the Region of Interest from the frames and then using Deep Learning algorithms, two individual models are created and then the two models are combined to produce the hybrid model for Visual Speech Recognition.

3.1.1 Pre-processing

In the collection, there are videos of each speaker repeating each word ten times. To make the videos compatible with the desired specifications or constraints of 10 frames per second (FPS), the frames are resized during the pre-processing stage. This resizing step adjusts the dimensions of the video frames. The resulting videos maintain the same directory structure and contain around 10 to 15 frames for each instance.

The proposed work would capture a video input of the subject but the processing part demands only the use of visual input from which the lip region would be extracted and the further feature extraction would be done. So the audio from the video part would be separated and muted.

Nowadays, pre-processed datasets such as the MIRACL-VC1 dataset are becoming more accessible and widely utilized at no cost. The final aim of conversion to grayscale image has shown in Fig. 2 starts with the frames collected from these datasets are carefully examined individually.

Table 1 Words used to develop visual speech recognition

ID	Words	Resolution
1	Begin	640 × 480
2	Choose	640 × 480
3	Connection	640 × 480
4	Navigation	640 × 480
5	Next	640 × 480
6	Previous	640 × 480
7	Start	640 × 480
8	Stop	640 × 480
9	Hello	640 × 480
10	Web	640 × 480



Fig. 3 Frame Before mouth

Using the face landmark method, which is a function provided by the Dlib library, the frames are cropped to isolate the region of interest, specifically the mouth part. Additionally, to reduce the dataset’s size further, the images are converted to grayscale (Table 1).

3.1.2 Visual Models

By utilizing the Face landmark detection algorithm available in the Python 3 library called dlib, the mouth area is removed from individual frame of the movie database. Subsequently, to facilitate computations, the extracted lip region is transformed into grayscale. Following that, the coordinates corresponding to the outer lip are isolated and utilized. These coordinates typically fall within the range of 48 to 68.

The frame captured before the extraction of ROI is shown in Fig. 3. The localized lip information is stored as a Python tuple, containing the coordinates (x, y), thickness (w), and altitude (h). Afterward, using the Numpy library in Python, the lip coordinates (x, y) are saved as an independent array. Typically, 68 face landmarks are detected, but only landmarks 48 to 68 are utilized for lip localization, while the remaining landmarks are disregarded or hidden.

The dlib shape predictor is employed to identify the essential facial landmarks, and based on these landmarks, the photos are cropped and saved as shown in Fig. 4. The images are then loaded, resized, and converted to grayscale. Next,



Fig. 4 Extracting Mouth Region ROI extraction

the necessary words for training the preprocessed data are loaded. If the corresponding folder is not already present, it is created, and the directory path is updated to store all the data in that specific folder.

The processed data is subsequently utilized in the improvement of a Convolutional Neural Network model for Visual speech recognition. The information of this archetypal are explained below.

3.2 Dataset Description

The dataset The MIRACL-VC1 dataset is a comprehensive lip-reading dataset that includes both depth and color images. It is designed for various research fields, such as visual speech recognition, face detection, and biometrics. The dataset comprises recordings of fifteen speakers positioned within the field of view of an MS Kinect sensor. Each speaker utters a predefined set of ten words and ten phrases, with each utterance repeated ten times. For each instance in the dataset, there is a synchronized sequence of color and depth images, both having dimensions of 640 × 480 pixels. In total, the MIRACL-VC1 dataset contains 3000 instances, making it a valuable resource for researchers working on lip-reading and related tasks.

3.3 Visual Speech Recognition Using CNN 3D Model

When dealing with tasks that involve analyzing features or relationships in a three-dimensional (3D) space, such as in visual speech recognition, 3D convolutions are utilized. These convolutions operate similarly to 2D convolutions but involve moving a kernel in three dimensions instead of two. This enables a more comprehensive capturing of dependencies within the 3D space. The output dimensions of the convolutional operation are different from those in 2D convolutions. To ensure the kernel can traverse all three dimensions, its depth should be smaller than the depth of the feature map being convolved.

Convolutional Neural Networks (CNNs or ConvNets) are popular deep learning models extensively utilized for analyzing visual images. Unlike traditional neural networks that rely on matrix multiplications, CNNs utilize a unique

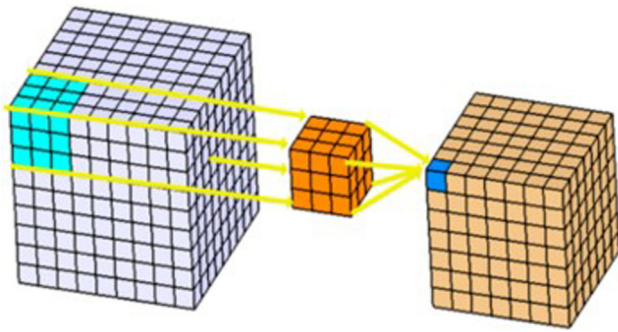


Fig. 5 Convolution 3D dept dimension

technique known as convolution. In mathematical terms, convolution involves the process implemented on two functions to produce a third function that describes how one function's shape is altered by the other.

Convolutional Neural Networks (CNNs) contain of various layers of artificial neurons, mimicking the structure of biological neurons. These artificial neurons perform mathematical computations by calculating the weighted sum of inputs and generating an activation value. In CNNs, when an image is served into the network, each layer produces multiple activation functions, which are then passed on to the subsequent layers. This process enables the network to extract and learn hierarchical features at different levels, capturing progressively intricate patterns and structures within the image.

CNNs employ filters or kernels to identify specific landscapes, such as edges, present in an image. The CNN operations can be summarized into four steps. The first step is convolution, second layer is non linearity, third layer is pooling layer and the four layer is classification layer.

3.3.1 Convolution

A 3D CNN shares similarities with a 2D CNN but with a few distinctions. In a 2D Convolution Layer, the input and filters are 2D matrices, and the convolution operation involves multiplying corresponding entries. However, in a 3D Convolution Layer, this operation is performed on multiple pairs of 2D matrices. 3D convolutions utilize a 3D filter that transfers in three orders (x, y, z) to estimate low-level features within a dataset. The output of 3D convolutions is a 3D volume space as shown in Fig. 5, which can be visualized as a cube or cuboid. While they are commonly employed for event recognition in videos and three-dimensional medical images, 3D convolutions can also be functional to two-dimensional inputs like images.

Assuming we have a square neuron layer of size $N \times N$, tracked by a convolutional layer, if we apply an $m \times m$ mesh ω , the productive of the convolutional layer will

have dimensions. To compute the pre-nonlinearity input for a specific unit x_{ij} in our layer, we sum up the influences from the previous layer cells, weighted by the corresponding filter components.

$$x_{ij}^l = \sum_{a=0}^{m-1} \sum_{b=0}^{m-1} w_{ab} Y_{(i+a)(j+b)}^{l-1} \quad (1)$$

where,

x_{ij}^l represents the activation at position (i,j) in the convolutional layer.

$Y_{(i+a)(j+b)}^{l-1}$ denotes the input activation at position (i + a, j + b) in the m th input channel.

$W(ab)$ is the weight parameter related with the association among the m th input channel and the feature map at position (a,b).

The 3D CNN prototypical contains of three convolutional layers and three fully connected layers. Each layer utilizes filters with sizes of 64, 128, and 256, respectively. The pool size and kernel size for these layers are both set to 3. To address overfitting, dropout regularization is applied with a rate of 0.5.

3.3.2 Max-Pooling Layers

After each convolutional layer, there is a possibility of having a pooling layer. The purpose of the pooling layer is to extract a single output from small rectangular blocks taken from the convolutional layer through subsampling. Different pooling methods exist, such as average pooling, maximum pooling, or using a learned linear combination of neurons within the block. In our scenario, only max-pooling will be used in the pooling layers, where the highest value within each pooling block is selected.

Max-pooling layers are simple and do not contain parameters that can be learned. Their operation involves taking an input region of size $k \times k$ and producing a single output value, which is determined as the maximum value within that region. To illustrate, if the input layer has dimensions $N \times N$, the output layer generated by the max-pooling layer will have dimensions $N/k \times N/k$. This reduction in size is due to the fact that each $k \times k$ block in the input is condensed into a single value using the maximum function.

To calculate the dimensions of the output volume when applying a pooling layer with kernel/filter size F and stride S to an input volume with width $W1$, height $H1$, and depth $D1$, the following formulas can be used:

$$W^2 = \frac{(W^1 - F)}{S} + 1 \text{ Defines the Output Width} \quad (2)$$

$$H^2 = \frac{(H^1 - F)}{S} + 1 \text{ Defines the Output Height} \quad (3)$$

$$D2 = D1 \text{Defines the Output Depth,} \tag{4}$$

These equations determine the dimensions of the output volume after applying the pooling layer over the input volume.

3.3.3 Fully-Connected

Following multiple convolutional and max pooling layers, the neural network engages in higher-level reasoning by employing fully connected layers. In a fully connected layer, all neurons from the previous layer are connected, regardless of whether they originated from fully connected, pooling, or convolutional layers. In these fully connected layers, the neurons lose their spatial organization and can be envisioned as one-dimensional. As a result, it is not possible to have any convolutional layers following a fully connected layer.

In a fully connected layer, the input data matrix, usually obtained from the previous layer, is multiplied with a weights matrix using the dot product operation. This multiplication enables the layer to learn and compute intricate relationships between the input data and the associated weights. By adjusting the weights during training, the fully connected layer can capture and model complex patterns and interactions within the data.

$$y = X.W \tag{5}$$

In a fully connected layer, the input data is usually in the form of (batch_size, n_features), while the weight matrix has the shape of (n_features, n_units). This equation is important because it indicates that the output shape will be (batch_size, n_units). If we desire to include a bias vector with a size of n_units, the equation for the fully connected layer can be adjusted to incorporate the bias term. In this scenario, the output shape will remain unchanged, but each output value will be added to its corresponding bias value.

$$y = XW + b \tag{6}$$

By assuming the activation function as an arbitrary function, f , the complete operation of a fully connected layer can be expressed as follows:

$$y = f(XW + b) \tag{7}$$

During the training process, both the weight matrix (W) and the bias vector (b) will be learned in order to optimize the model’s fit to the training data (X, y).

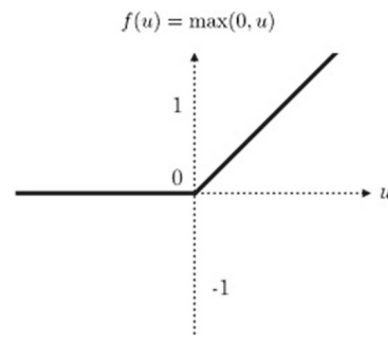


Fig. 6 ReLU activation curve

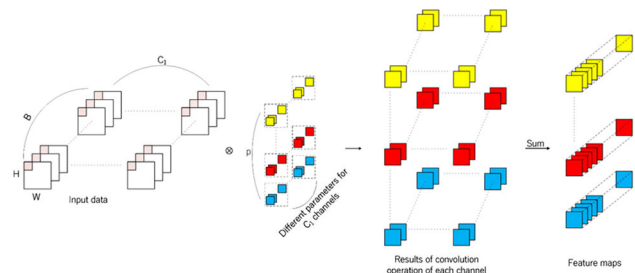


Fig. 7 Step by step CNN model operation

3.3.4 Activation Function

The activation functions used in our model are ReLU (Rectified Linear Unit) activation functions. ReLU is a significant advancement in the field of deep learning and has surpassed previous activation functions like sigmoid or tanh. It is both simple and superior in performance. The formula for the ReLU activation function is,

$$f(x) = \max(0, x) \tag{8}$$

The derivative of the ReLU function is also monotonic, meaning it maintains the same order or direction. When the ReLU function receives a negative input, it returns a value of 0. However, for any positive input value x , it proceeds the same value as shown in Fig. 6. As a result, the ReLU function produces outputs within the range of 0 to infinity.

The 3D convolutional layer has p kernels, each with a size of $k \times k \times k$, resulting in $p \times k \times k \times k \times C1$ weight parameters in the network. The determination of the 3D convolution process is to excerpt features from the input data, resulting in a feature map that contains more spectral information.

The CNN model is illustrated in Fig. 7 operates in a step-by-step manner, where the input data is processed through convolutional layers, typically followed by activation functions such as ReLU, pooling layers for downsampling, and fully connected layers for classification. This sequential processing allows the model to learn hierarchical representations

and extract relevant features from the input data, enabling accurate classification.

In the complete 3D convolution process, represented by a kernel size of $k \times k \times k$, the last dimension of k denotes the extent of the convolution kernel across the spectral dimension in each convolution operation. If padding is not applied and the stride is set to 1, the resulting output data dimension for a single kernel can be computed as $(W - k + 1) \times (H - k + 1) \times (B + k + 1)$. The output information dimension in this context is determined by the input data dimensions: W for width, H for height, and B for depth (or time) dimension. When performing the convolution operation without padding and with a stride of 1, the output dimension is calculated by subtracting the kernel size (k) minus 1 from the corresponding input dimension and adding 1 to the depth dimension (B).

In the complete 3D convolution process, represented by a kernel size of $k \times k \times k$, the last dimension of k denotes the extent of the convolution kernel across the spectral dimension in each convolution operation. If padding is not applied and the stride is set to 1, the resulting output data dimension for a single kernel can be computed as $(W - k + 1) \times (H - k + 1) \times (B + k + 1)$. For every kernel utilized in the 3D convolution operation, this process is repeated, leading to multiple output feature maps with respective dimensions. These feature maps play a crucial role in capturing significant spatial and spectral information, enabling the extraction of meaningful features from the input data. The model structure used in the project is shown in Fig. 8 that is used for English dataset as shown below

3.4 Convolutional Layer

Convolutional layers in a neural network consist of a grid-like arrangement of neurons, forming a rectangular shape. For the convolutional layer to work effectively, the preceding layer must also be organized as a rectangular grid of neurons. Figure 9 shows the setup, each neuron in the convolutional layer receives inputs from a corresponding rectangular region in the previous layer. Importantly, the weights associated with this rectangular region are shared by all the neurons within the convolutional layer.

Spatial features in signal data refer to the patterns and structures found within specific locations or areas. For example, in images, spatial features can include edges, corners, textures, shapes, or local patterns. By incorporating spatial features, a model can learn to recognize unique visual elements that define particular classes. In tasks like image recognition, spatial features are essential for identifying objects or shapes that distinguish one class from another.

On the other hand, temporal features capture patterns and variations that evolve over time in sequential data, such as

```

Model: "sequential"
-----
Layer (type)                Output Shape                Param #
-----
conv3d (Conv3D)              (None, 20, 98, 98, 64)    1792
max_pooling3d (MaxPooling3D) (None, 10, 49, 49, 64)    0
conv3d_1 (Conv3D)            (None, 8, 47, 47, 128)    221312
max_pooling3d_1 (MaxPooling3D) (None, 4, 23, 23, 128)    0
conv3d_2 (Conv3D)            (None, 3, 22, 22, 256)    262400
max_pooling3d_2 (MaxPooling3D) (None, 1, 11, 11, 256)    0
flatten (Flatten)           (None, 30976)              0
dense (Dense)                (None, 4096)               126881792
dropout (Dropout)           (None, 4096)               0
dense_1 (Dense)              (None, 2048)               8390656
dropout_1 (Dropout)         (None, 2048)               0
dense_2 (Dense)              (None, 10)                 20490
-----
Total params: 135,778,442
Trainable params: 135,778,442

```

Fig. 8 Model structure of CNN 3D model for the Dataset

audio signals. For audio signals, temporal features are crucial in identifying phonetic patterns, speech intonations, or musical rhythms.

By incorporating both spatial and temporal features, a model gains contextual understanding of the signal data, considering relationships between different parts of the signal and how they change over time. This comprehensive approach enables the model to make more informed classification decisions in various signal processing tasks, such as speech recognition, action recognition in videos, and other sequential data classification tasks.

3.5 Visual Speech Recognition Using LSTM Model

Long Short-Term Memory networks are a specific form of recurrent neural network that demonstrate strong performance in recognizing patterns within sequential data. They are particularly effective in jobs like machine interpretation and speech identification, where understanding the sequential order of the input is vital. The architecture of the network comprises a chain-like arrangement of interconnected neural networks, working in conjunction with specialized memory blocks called cells. These cells serve the purpose of storing and retaining information, while the network's gates regulate the flow of information within the memory blocks. An LSTM network consists of three types of gates: The Forget Gate, the Input Gate, and the Output Gate. Each gate plays a pivotal role in determining which information to forget, which new

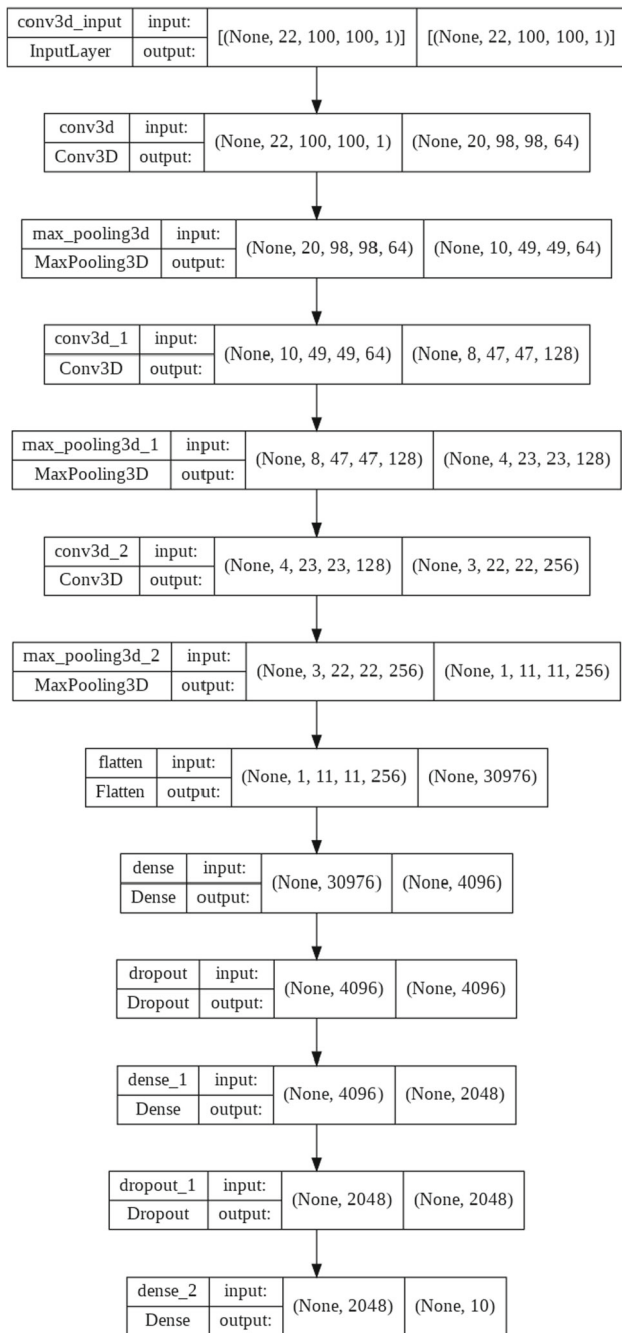


Fig. 9 Summary of model structure of convolutional model for the dataset

information to incorporate, and which information to output from the network.

To summarize, the initial layer of the model consists of an LSTM (Long Short-Term Memory) with 128 hidden units and operates on sequences with 8 timestamps. The processes intimate LSTM cell are as follows

$$\hat{c}^{<t>} = \tanh(W_c[a^{<t-1>}, x^{<t>}] + b_c) \tag{9}$$

Model: "sequential_12"

Layer (type)	Output Shape	Param #
lstm_24 (LSTM)	(None, 22, 128)	5186048
batch_normalization_24 (Batch Normalization)	(None, 22, 128)	512
lstm_25 (LSTM)	(None, 256)	394240
batch_normalization_25 (Batch Normalization)	(None, 256)	1024
dense_36 (Dense)	(None, 512)	131584
dropout_21 (Dropout)	(None, 512)	0
dense_37 (Dense)	(None, 1024)	525312
dropout_22 (Dropout)	(None, 1024)	0
dense_38 (Dense)	(None, 10)	10250

=====
 Total params: 6,248,970
 Trainable params: 6,248,202
 Non-trainable params: 768
 =====

Fig. 10 Model structure for LSTM model for custom dataset

$$\Gamma_u = \sigma\left(W_u\left[a^{<t-1>}, x^{<t>}\right] + b_u\right) \tag{10}$$

$$\Gamma_f = \sigma\left(W_f\left[a^{<t-1>}, x^{<t>}\right] + b_f\right) \tag{11}$$

$$\Gamma_o = \sigma\left(W_o\left[a^{<t-1>}, x^{<t>}\right] + b_o\right) \tag{12}$$

$$c^{<t>} = \Gamma_u * \hat{c}^{<t>} + \Gamma_f * c^{<t-1>} \tag{13}$$

$$a^{<t>} = \Gamma_o \tanh(c^{<t>}) \tag{14}$$

In the LSTM cell as in Fig 10, we have various components denoted by symbols: c for the memory cell, t for the timestamp, for the update gate, F for the forget gate, for the output gate, for the sigmoid function, W_u for the weights of the update gate, W_f for the weights of the forget gate, for the weights of the output gate, b for bias, and for the candidate cell variables..

A standard LSTM network consists of memory blocks called cells. These cells transfer two states, namely the cell state and the hidden state, to the next cell in the network. The main purpose of these memory blocks is to retain information, and the memory manipulation occurs through three key mechanisms known as gates. The overall structure is shown in Fig 10, followed by its internal structure in Figs. 11, 12.

The concatenation method is used where, the spatial and temporal features are combined by concatenating them along a specific dimension, such as the feature dimension. For instance, if the spatial features are represented as a vector

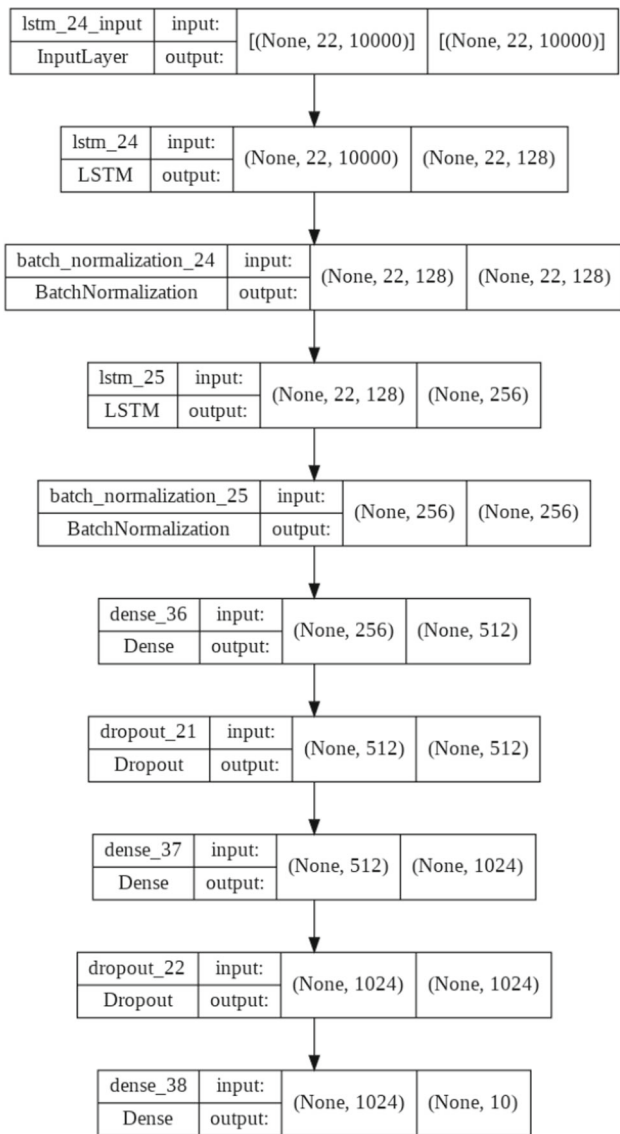


Fig. 11 Summary of model structure of LSTM model for the dataset

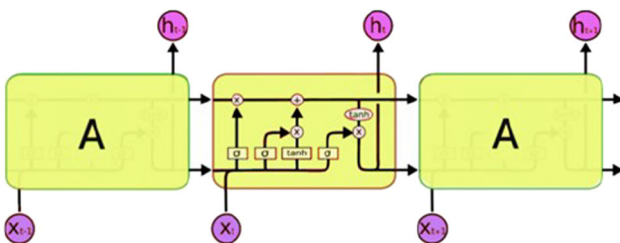


Fig. 12 Internal structure of LSTM cell

with a size of N_{spatial} , and the temporal features are represented as a sequence of vectors with T_{temporal} time steps, the fused feature tensor would have a shape of $(T_{\text{temporal}}$,

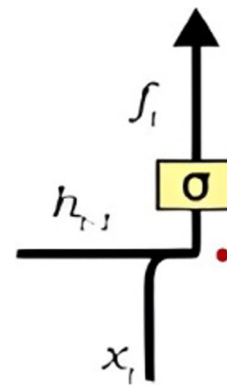


Fig. 13 Structure of Forget Gate

$N_{\text{spatial}} + N_{\text{temporal}}$). This concatenated representation is then used as the input to the Bidirectional Long Short-Term Memory (BiLSTM) network.

3.5.1 Forget Gate

The use of forget gate in an LSTM model is to selectively discard information from the cell state. It serves to discard unwanted information or less important for the LSTM to understand the input sequence effectively. The forget gate achieves this by multiplying a filter with the existing cell state. This filtering process enhances the efficiency of the LSTM network by managing the preservation or elimination of particular data. The agram demonstrates the organization and constituents of the forget gate in the LSTM model.

The mentioned gate receives two inputs: h_{t-1} and x_t . h_{t-1} denotes the previous cell’s hidden state or output, while x_t represents the input at the current time step. These inputs undergo multiplication with weight matrices and the addition of a bias. Then, the resulting value is passed through the sigmoid function. The sigmoid function generates a vector with values between 0 and 1, representing each element in the cell state. This Forget gate structural illustration in is seen in Fig. 13.

In essence, the sigmoid function plays a crucial role in making decisions by determining which values should be kept and which should be discarded. A value of ‘0’ signifies that the forget gate wants the cell state to completely disregard that specific information. Conversely, a value of ‘1’ indicates that the forget gate intends to retain the entire information. The output vector obtained from the sigmoid function is then multiplied element-wise with the cell state, modifying its contents accordingly.

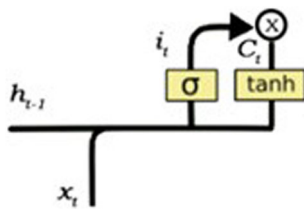


Fig. 14 Structure of Input Gate

3.5.2 Input Gate

The input gate assumes the responsibility of incorporating new information into the cell state. The provided diagram illustrates the composition of the Input Gate, outlining its structure and components which can be observed.

- a. The cell state’s information addition process is regulated by a sigmoid function, similar to the forget gate. This function serves as a filter for information originating from h_{t-1} and x_t .
- b. A vector is created to encompass all possible values that can be added to the cell state, considering h_{t-1} and x_t . This vector is generated using the tanh function, producing values ranging from -1 to $+1$.
- c. The regulatory filter, obtained from the sigmoid gate, is multiplied by the vector produced in step (b) using the tanh function as shown in Fig. 14. The resulting value, representing valuable information, is added to the cell state through an addition operation.

3.5.3 Output Gate:

The output gate assumes the role of selecting and presenting relevant information from the current cell state as an output. The operation of the output gate can be divided into three distinct steps:

- a. A vector is created by applying the tanh function to the cell state, which scales the values within the range of -1 to $+1$.
- b. A filter is generated using the values of h_{t-1} and x_t to control which values should be output from the vector created in step 1. This filter employs a sigmoid function.
- c. The regulatory filter obtained in step 2 is multiplied by the vector generated in step 1. The resulting value serves as both the output of the current cell and the hidden state for the subsequent cell. This can be visualized with the help of Fig. 15.

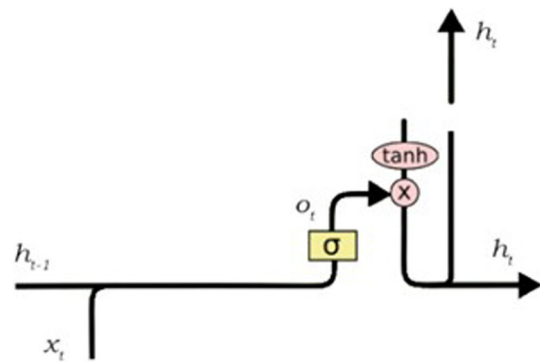


Fig. 15 Structure of Output Gate

3.5.4 Dense Layer

In a model’s dense layer, the neurons receive input from every neuron in the preceding layer. The computation carried out by the neurons in the dense layer involves matrix–vector multiplication. During this process, the output from the preceding layer is treated as a row vector, and the weights of the dense layer are treated as a column vector. It is essential to emphasize that in matrix–vector multiplication, the row vector must have the same number of elements as the column vector.

$$y = \sigma(Wx + b) \tag{15}$$

where

- The input to the dense layer is denoted by x , which is commonly a vector.
- The weight matrix of the dense layer is represented by W , where each column corresponds to the weights associated with a specific neuron in the layer.
- The bias vector of the dense layer is represented by b , with each element corresponding to the bias value for a specific neuron. σ represents the activation function applied element-wise to the result of $Wx + b$.
- y represents the output vector of the dense layer, which is obtained after applying the activation function.

3.6 Visual Speech Recognition Using Hybrid Model

The CNN-LSTM model was designed to address visual time series prediction tasks and generate textual descriptions based on sequences of images. It combines the strengths of convolutional neural network (CNN) layers for feature extraction with the sequential prediction capabilities of LSTM (Long Short-Term Memory) networks. The structure of hybrid model implemented using CNN and LSTM can be looked into detail in Fig. 16

Model: "sequential_1"

Layer (type)	Output Shape	Param #
conv3d_3 (Conv3D)	(None, 20, 98, 98, 32)	896
max_pooling3d_3 (MaxPooling 3D)	(None, 10, 49, 49, 32)	0
conv3d_4 (Conv3D)	(None, 8, 47, 47, 64)	55360
max_pooling3d_4 (MaxPooling 3D)	(None, 4, 23, 23, 64)	0
conv3d_5 (Conv3D)	(None, 2, 21, 21, 128)	221312
max_pooling3d_5 (MaxPooling 3D)	(None, 1, 10, 10, 128)	0
reshape_1 (Reshape)	(None, 128, 100)	0
lstm_1 (LSTM)	(None, 128, 64)	42240
flatten_1 (Flatten)	(None, 8192)	0
dense_3 (Dense)	(None, 2048)	16779264
dropout_2 (Dropout)	(None, 2048)	0
dense_4 (Dense)	(None, 1024)	2098176
dropout_3 (Dropout)	(None, 1024)	0
dense_5 (Dense)	(None, 10)	10250

 Total params: 19,207,498
 Trainable params: 19,207,498
 Non-trainable params: 0

Fig. 16 Shows the structure of hybrid model implemented using CNN and LSTM

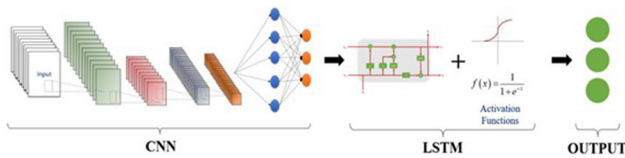


Fig. 17 Structure of CNN-LSTM model

In the CNN-LSTM design shown by Fig. 17, CNN layers are employed to capture important features from the spatial input data. These extracted features are then fed into the LSTM network to generate captions or predictions based on the sequence. This integration of CNN and LSTM allows the model to leverage the spatial information captured by the CNN layers and incorporate it into the sequential prediction process of the LSTM network.

4 Result for Visual Speech Recognition Using CNN 3D Model

4.1 Dataset Details

The MIRACL VC1 datasets is used wherein 15 speakers out of which 10 are female speakers and remaining 5 are male

Table 2 Details of the dataset we used in the research work

Words	Number of speakers	Total no. of words repeated	Total no. of samples
Begin	15 (10 F + 5 M)	10	15*10 = 150
Choose	15 (10 F + 5 M)	10	15*10 = 150
Connection	15 (10 F + 5 M)	10	15*10 = 150
Navigation	15 (10 F + 5 M)	10	15*10 = 150
Next	15 (10 F + 5 M)	10	15*10 = 150
yPrevious	15 (10 F + 5 M)	10	15*10 = 150
Start	15 (10 F + 5 M)	10	15*10 = 150
Stop	15 (10 F + 5 M)	10	15*10 = 150
Hello	15 (10 F + 5 M)	10	15*10 = 150
Web	15 (10 F + 5 M)	10	15*10 = 150

where *F*—Female, *M*—Male

```

Epoch 37/45
42/42 [=====] - 20s 474ms/step - loss: 0.4251 - accuracy: 0.8600 - val_loss: 0.6396 - val_accuracy: 0.7867
Epoch 38/45
42/42 [=====] - 20s 465ms/step - loss: 0.4164 - accuracy: 0.8676 - val_loss: 0.6065 - val_accuracy: 0.7733
Epoch 39/45
42/42 [=====] - 20s 474ms/step - loss: 0.3921 - accuracy: 0.8705 - val_loss: 0.5786 - val_accuracy: 0.7933
Epoch 40/45
42/42 [=====] - 20s 465ms/step - loss: 0.3788 - accuracy: 0.8762 - val_loss: 0.6118 - val_accuracy: 0.7600
Epoch 41/45
42/42 [=====] - 20s 474ms/step - loss: 0.3444 - accuracy: 0.8848 - val_loss: 0.5798 - val_accuracy: 0.7867
Epoch 42/45
42/42 [=====] - 20s 474ms/step - loss: 0.3275 - accuracy: 0.8981 - val_loss: 0.6401 - val_accuracy: 0.7467
Epoch 43/45
42/42 [=====] - 20s 474ms/step - loss: 0.3407 - accuracy: 0.8876 - val_loss: 0.5687 - val_accuracy: 0.7408
Epoch 44/45
42/42 [=====] - 20s 474ms/step - loss: 0.3289 - accuracy: 0.8952 - val_loss: 0.5621 - val_accuracy: 0.8133
Epoch 45/45
42/42 [=====] - 20s 466ms/step - loss: 0.2993 - accuracy: 0.9152 - val_loss: 0.6112 - val_accuracy: 0.7867

Training time : 897.988000154952 secs.
    
```

Fig. 18 Training with CNN 3D model

speakers. Each of the speakers spell 10 words 10 times and hence the total dataset we have used becomes 1500 (Table 2).

4.2 Result:

Training of Dataset is done for 45 epochs which is observed in Fig. 18. Achieved training accuracy of 91.52%, test accuracy of 78.67% and validation accuracy of 79.00%,

Figures 19, 20 represent model accuracy and loss, respectively. The confusion matrix is a valuable tool for analyzing

Table 3 Classification matrix for CNN 3D model

	Precision	Recall	F1-Score	Support
Begin	0.84	0.70	0.76	30
Choose	0.94	0.97	0.95	30
Connection	0.86	0.80	0.83	30
Navigation	0.91	0.70	0.79	30
Next	0.73	0.80	0.76	30
Previous	0.92	0.77	0.84	30
Start	0.65	0.87	0.74	30
Stop	0.56	0.83	0.67	30
Hello	0.89	0.83	0.86	30
Web	0.82	0.60	0.69	30
Accuracy			0.79	300
Macro Avg	0.81	0.79	0.79	300
Weighted Avg	0.81	0.79	0.79	300

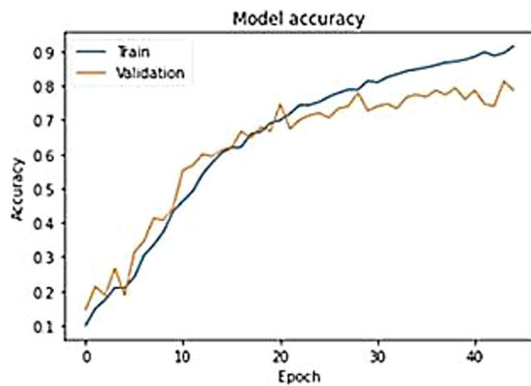


Fig. 19 Model Accuracy plot for CNN 3D model

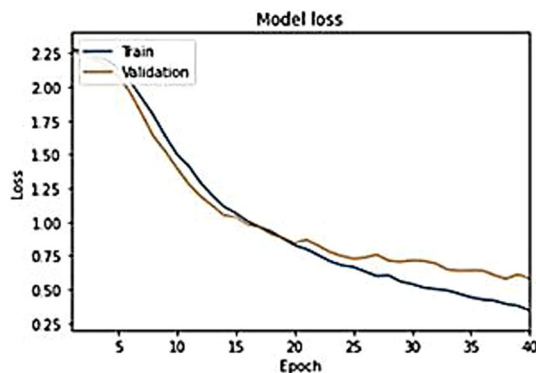


Fig. 20 Loss plot for CNN 3D model

how words are classified into different classes. It provides a summary of the number of words that are correctly or incorrectly classified for each class in the dataset as shown in Fig. 21. This matrix helps evaluate the accuracy of word classification. To calculate the accuracy percentage for each

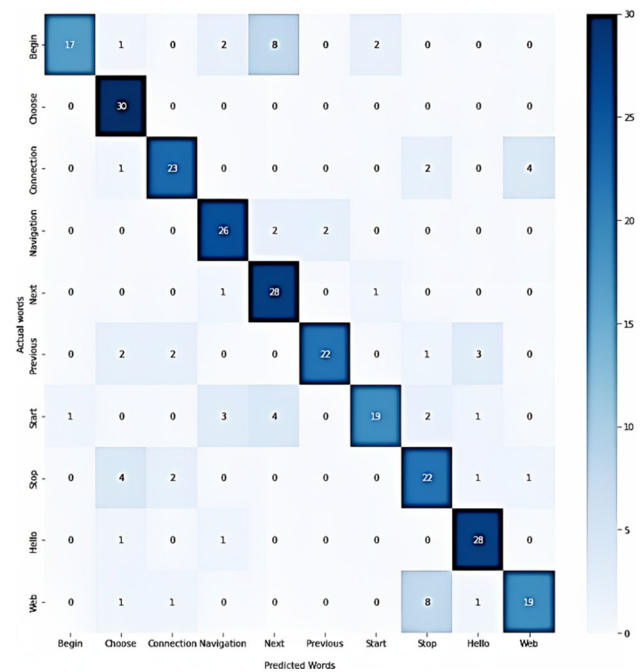


Fig. 21 Confusion matrix for CNN 3D model

word, a specific formula is required, which has not been provided in the given statement. However, the common method entails dividing the count of accurately classified words by the total number of words and then multiplying the outcome by 100 to derive a percentage. By analyzing the confusion matrix and computing the accuracy for each word, we can evaluate the performance of the classification model as shown in Table 3 and identify any trends or areas for improvement in the accuracy of word classification.

$$Accuracy(\%) = \frac{w}{W} \times 100 \tag{16}$$

```

Epoch 37/45
42/42 [=====] - 3s 64ms/step - loss: 1.0341 - accuracy: 0.6114 - val_loss: 1.3269 - val_accuracy: 0.5600
Epoch 38/45
42/42 [=====] - 3s 65ms/step - loss: 0.9924 - accuracy: 0.6152 - val_loss: 0.9917 - val_accuracy: 0.6000
Epoch 39/45
42/42 [=====] - 3s 65ms/step - loss: 0.9273 - accuracy: 0.6543 - val_loss: 1.1860 - val_accuracy: 0.5733
Epoch 40/45
42/42 [=====] - 3s 65ms/step - loss: 0.9163 - accuracy: 0.6514 - val_loss: 0.8796 - val_accuracy: 0.6733
Epoch 41/45
42/42 [=====] - 3s 65ms/step - loss: 0.8208 - accuracy: 0.6848 - val_loss: 0.8709 - val_accuracy: 0.6333
Epoch 42/45
42/42 [=====] - 3s 66ms/step - loss: 0.8627 - accuracy: 0.6790 - val_loss: 1.1231 - val_accuracy: 0.6000
Epoch 43/45
42/42 [=====] - 3s 65ms/step - loss: 0.8400 - accuracy: 0.6629 - val_loss: 0.8888 - val_accuracy: 0.6867
Epoch 44/45
42/42 [=====] - 3s 66ms/step - loss: 0.7847 - accuracy: 0.6876 - val_loss: 0.9235 - val_accuracy: 0.6333
Epoch 45/45
42/42 [=====] - 3s 66ms/step - loss: 0.8092 - accuracy: 0.6943 - val_loss: 0.8692 - val_accuracy: 0.6933
Training time : 144.789377450943 secs.
    
```

Fig. 22 Training with LSTM model

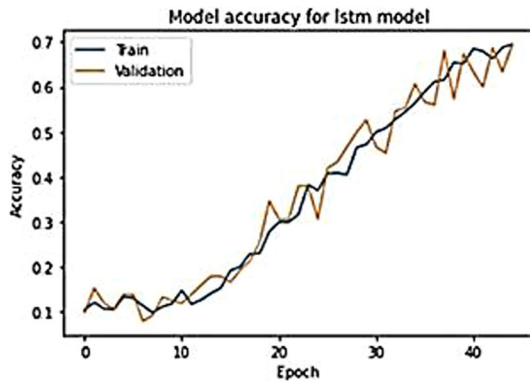


Fig. 23 Accuracy Plot with LSTM Model

w- total number of correctly predicted instances for a word in the dataset

W- total number of instances for a word used in dataset.

To calculate the overall parameters like precision Recall and f1score the below formula is used

$$\text{Precision} = \frac{\text{TruePositive}}{(\text{TruePositive} + \text{FlasePositive})} \quad (17)$$

$$\text{Recall} = \frac{\text{TruePositive}}{(\text{TruePositive} + \text{FlaseNegative})} \quad (18)$$

$$f1\text{score} = \frac{2 * (\text{Recall} * \text{Precision})}{(\text{Recall} + \text{Precision})} \quad (19)$$

4.3 Result For Visual Speech Recognition Using LSTM Model

Dataset is trained for 45 epochs which is observed in Fig. 22. Achieved training accuracy of 69.43%, test accuracy of 66.00% and validation accuracy of 69.33%

Figures 23, 24 represent model accuracy and loss, respectively, for an LSMT modeled network. The performance evaluation of the classification model as shown in Table 4.

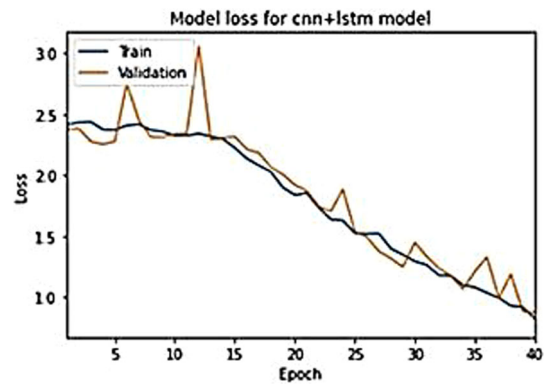


Fig. 24 Loss Plot with LSTM Model

```

Epoch 38/45
42/42 [=====] - 9s 211ms/step - loss: 0.0293 - accuracy: 0.9933 - val_loss: 0.7234 - val_accuracy: 0.8267
Epoch 39/45
42/42 [=====] - 9s 210ms/step - loss: 0.0242 - accuracy: 0.9933 - val_loss: 0.7037 - val_accuracy: 0.8200
Epoch 40/45
42/42 [=====] - 9s 210ms/step - loss: 0.0286 - accuracy: 0.9895 - val_loss: 0.5227 - val_accuracy: 0.8667
Epoch 41/45
42/42 [=====] - 9s 211ms/step - loss: 0.0582 - accuracy: 0.9829 - val_loss: 0.6826 - val_accuracy: 0.8200
Epoch 42/45
42/42 [=====] - 9s 210ms/step - loss: 0.0275 - accuracy: 0.9914 - val_loss: 0.7782 - val_accuracy: 0.8467
Epoch 43/45
42/42 [=====] - 9s 211ms/step - loss: 0.0251 - accuracy: 0.9914 - val_loss: 0.6373 - val_accuracy: 0.8467
Epoch 44/45
42/42 [=====] - 9s 210ms/step - loss: 0.0191 - accuracy: 0.9943 - val_loss: 0.9593 - val_accuracy: 0.8400
Epoch 45/45
42/42 [=====] - 9s 216ms/step - loss: 0.0180 - accuracy: 0.9952 - val_loss: 0.6936 - val_accuracy: 0.8667
    
```

Fig. 25 Training dataset with hybrid model

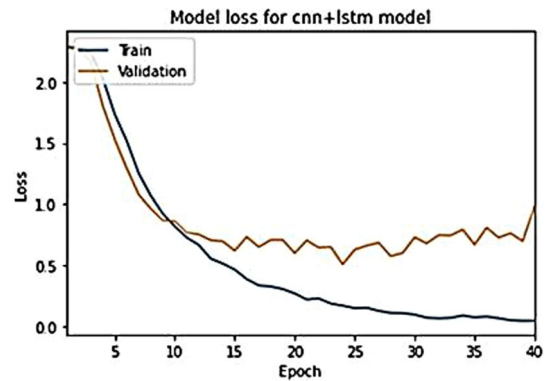


Fig. 26 Accuracy plot with hybrid model

4.4 Result for Visual Speech Recognition Using Hybrid Model

Training is done for 45 epochs which is observed in Fig. 25. Achieved training accuracy of 99.52%, test accuracy of 85.00% and validation accuracy of 86.67%

Figures 26, 27 represent model accuracy and model loss, respectively, for an LSMT and CNN combined modeled network.

The confusion matrix shown in Fig. 28 displays the count of words classified into different available classes. It serves

Table 4 Classification matrix with LSTM Model

	Precision	Recall	F1-Score	Support
Begin	0.68	0.50	0.58	30
Choose	0.84	0.87	0.85	30
Connection	0.60	0.90	0.72	30
Navigation	0.61	0.90	0.73	30
Next	0.60	0.70	0.65	30
Previous	0.61	0.37	0.46	30
Start	0.60	0.70	0.65	30
Stop	0.50	0.27	0.35	30
Hello	0.76	0.87	0.81	30
Web	0.75	0.50	0.60	30
Accuracy			0.66	300
Macro avg	0.66	0.66	0.64	300
Weighted avg	0.66	0.66	0.64	300

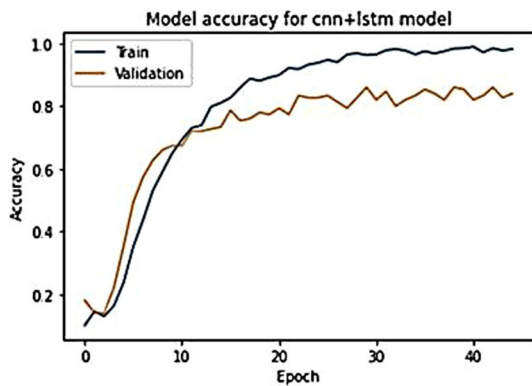


Fig. 27 Loss plot with hybrid model

to evaluate the accuracy of each word utilized in the dataset. Table 5 describes the comparison of the proposed work with existing work.

5 Conclusion

A Visual Speech Recognition (VSR) system is currently under development with the aim of interpreting videos. The project has made significant progress in achieving the majority of its objectives in a methodical manner. Video recognition was performed individually for a CNN 3D model and an LSTM model using a custom dataset. These two models were then combined into a hybrid model, resulting in a significant improvement in training accuracy. The dataset achieved testing accuracies of 79% for the CNN 3D model, 66% for the LSTM model, and 85% for the hybrid model (CNN + LSTM). This indicates that the proposed methodology outperforms existing approaches, and the hybrid model

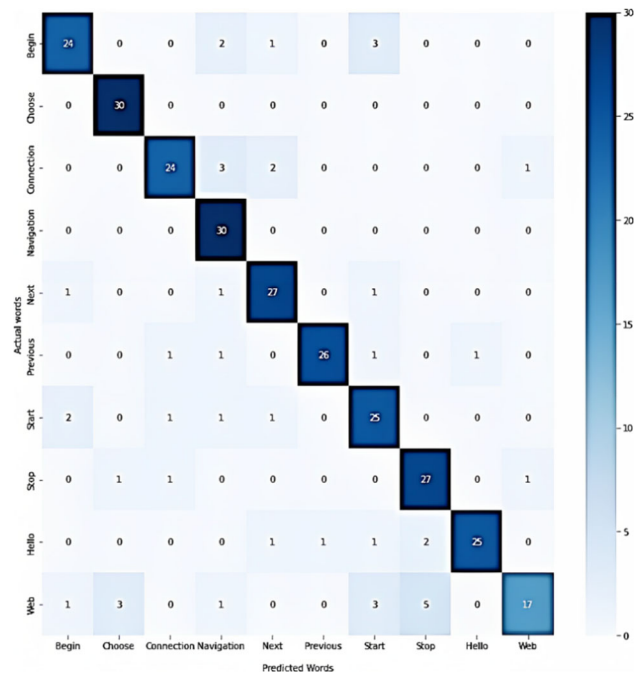


Fig. 28 Confusion matrix for hybrid model

offers a promising opportunity to achieve high performance in Visual Speech Recognition.

Visual speech recognition is valuable as it enables machines to understand spoken languages even in noisy environments, with applications ranging from enhanced hearing aids to biometric authentication. The VSR model utilizes open-source module libraries, making it accessible to everyone for recognizing spoken words.

However, there are some limitations to the VSR model. Since the accuracy is not 100%, there may be errors in interpreting words using Visual Speech Recognition alone.

Table 5 Summary of the proposed system

Method	Dataset	Accuracy (%)	Error (%)
HMM[17]	MIRACL VC1	65	35
CNN + GRU [10]	MIRACL VC1	90	10
CNN 3D [proposed]	MIRACL VC1	78.67	21.33
LSTM [proposed]	MIRACL VC1	66.00	34
Hybrid Model CNN 3D + LSTM [proposed]	MIRACL VC1	85.00	15

The training for the words would fetch a higher accuracy comparatively but the usage of concatenated words in-terms of sentences would be harder and consume a longer duration also would give a lower accuracy. Incorporating audio along with the video can improve accuracy and make the system more compatible compared to relying solely on Visual Speech Recognition.

Author contributions All the authors have immensely contributed for the research work.

Funding Not applicable.

Data availability Not applicable.

Code availability The code available on reasonable request to the corresponding author.

Declarations

Conflicts of interest None.

References

1. E Tatulli; T Hueber, (2017) Feature extraction using multimodal convolutional neural networks for visual speech recognition. In: IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 10.1109/ICASSP.2017.7952701
2. YH Goh; KX Lau and YK Lee; (2019), Audio-visual speech recognition system using recurrent neural network. In: 4th International Conference on Information Technology (InCIT), 38–43, <https://doi.org/10.1109/INCIT.2019.8912049>.
3. Tadayon; Manie Pottie; Greg (2020) Comparative analysis of the hidden markov model and LSTM: a simulation approach <https://arxiv.org/abs/2008.03825>. <https://doi.org/10.48550/arXiv.2008.03825>.
4. Shillingford; Brendan; Yannis Assael; Matthew W. Hoffman; Thomas Paine; Cian Hughes; Utsav Prabhu; Hank Liao et al.(2018) ,Large-scale visual speech recognition. arXiv preprint [arXiv:1807.05162](https://arxiv.org/abs/1807.05162). <https://doi.org/10.48550/arXiv.1807.05162>
5. W Feng; N Guan; Y Li, X Zhang; Z Luo,(2017), Audio visual speech recognition with multimodal recurrent neural networks. In: 2017 International Joint Conference on Neural Networks (IJCNN), 681–688, <https://doi.org/10.1109/IJCNN.2017.7965918>.
6. P Zhou; W Yang; W Chen; Y Wang; J Jia, (2019), In: Modality attention for end-to-end audio-visual speech recognition. In: ICASSP 2019–2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 6565–6569, <https://doi.org/10.1109/ICASSP.2019.8683733>.
7. S Petridis, Z Li and M Pantic,(2017), End-to-end visual speech recognition with LSTMs. In: 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2592–2596, <https://doi.org/10.1109/ICASSP.2017.7952625>.
8. Zimmermann M; Mehdipour Ghazi M; Ekenel HK; Thiran JP (2017), Visual speech recognition using PCA networks and LSTMs in a Tandem GMM-HMM System. In: Chen CS., Lu J., Ma KK. (eds) Computer Vision – ACCV 2016 Workshops. ACCV. Lecture Notes in Computer Science, 10117, Springer, Cham. https://doi.org/10.1007/978-3-319-54427-4_2.
9. F. Tao and C. Busso, (2021), "End-to-End Audiovisual Speech Recognition System With Multitask Learning," in IEEE Transactions on Multimedia, vol. 23, pp. 1–11, doi: <https://doi.org/10.1109/TMM.2020.2975922>.
10. NK Mudaliar; K Hegde; A Ramesh; V Patil, (2020), Visual speech recognition: a deep learning approach. In: 5th International Conference on Communication and Electronics Systems (ICCES), 1218–1221, <https://doi.org/10.1109/ICCES48766.2020.9137926>.
11. Thanda, A.; Venkatesan, S.M.: Audio visual speech recognition using deep recurrent neural networks. In: Schwenker, F.; Scherer, S. (Eds.) Multimodal pattern recognition of social signals in human computer interaction MPRSS 2016. Lecture Notes in Computer Science, Springer, Cham (2017). https://doi.org/10.1007/978-3-319-59259-6_9
12. T Makino et al., (2019), Recurrent neural network transducer for audio-visual speech recognition. In: IEEE Automatic Speech Recognition and Understanding Workshop (ASRU), 905–912, <https://doi.org/10.1109/ASRU46091.2019.9004036>.
13. YH Goh; KX Lau; YK Lee, (2019), Audio-visual speech recognition system using recurrent neural network. In: 4th International Conference on Information Technology (InCIT), 38–43, <https://doi.org/10.1109/INCIT.2019.8912049>.
14. T Hori; J Cho; S Watanabe, (2018), End-to-end speech recognition with word-based rnn language models. In: IEEE Spoken Language Technology Workshop (SLT), 2018, 389–396, <https://doi.org/10.1109/SLT.2018.8639693>.
15. Amberkar; P Awasarmol; G Deshmukh; P Dave,(2018) Speech recognition using recurrent neural networks. In: International Conference on Current Trends towards Converging Technologies (ICCTCT), 1–4, <https://doi.org/10.1109/ICCTCT.2018.85-51185>.
16. R Su, L Wang and X Liu,(2017), Multimodal learning using 3D audio-visual data for audio-visual speech recognition. In: International Conference on Asian Language Processing (IALP), 40–43, <https://doi.org/10.1109/IALP.2017.8300541>.
17. Maind SB; Priyanka Wankar. (2014), Research paper on basics of artificial neural network. In: International Journal on Recent and Innovation Trends in Computing and Communication, 2(1): 96–100. <https://doi.org/10.1017/s10499-019-0069-10>
18. Noda, K.; Yamaguchi, Y.; Nakadai, K., et al.: Audio-visual speech recognition using deep learning. Appl. Intell. **42**, 722–737 (2015). <https://doi.org/10.1007/s10489-014-0629-7>
19. S Tamura et al.,(2015), Audio-visual speech recognition using deep bottleneck features and high-performance lipreading. In: 2015 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA), 575–582, <https://doi.org/10.1109/APSIPA.2015.7415335>.
20. Takashima Yuki; Ryo Aihara; Tetsuya Takiguchi; Yasuo Ariki; Nobuyuki Mitani; Kiyohiro Omori; Kaoru Nakazono. (2016), Audio-visual speech recognition using bimodal-trained bottleneck features for a person with severe hearing loss. In: Interspeech, 277–281.



21. HM Fayek; M Lech; L Cavedon, (2015), Towards real-time speech emotion recognition using deep neural networks, In: 9th International Conference on Signal Processing and Communication Systems (ICSPCS), <https://doi.org/10.1109/ICSPCS.2015.7391796>.
22. T Afouras; JS Chung; A Senior; O Vinyals; A Zisserman, (2018), Deep Audio-visual Speech Recognition, In: IEEE Transactions on Pattern Analysis and Machine Intelligence, <https://doi.org/10.1109/TPAMI.2018.2889052>.
23. Thanda, A.; Venkatesan, S.M.: Audio Visual Speech Recognition Using Deep Recurrent Neural Networks. In: Schwenker, F.; Scherer, S. (Eds.) Multimodal pattern recognition of social signals in human computer interaction MPRSS 2016. Lecture Notes in Computer Science, Springer, Cham (2017)
24. Graves Alex; Navdeep Jaitly. (2014), Towards end-to-end speech recognition with recurrent neural networks. In: International conference on machine learning, 1764–1772. PMLR, 2014.
25. T. Afouras JS; Chung A Senior; O Vinyals; A Zisserman,(2018), Deep audiovisual speech recognition, In: IEEE Transactions on Pattern Analysis and Machine Intelligence, <https://doi.org/10.1109/TPAMI.2018.2889052>.
26. Naeem Ahmed; Champa H, (2015), Applying hidden markov model technique in CSMMI for action and gesture recognition, Int. J. Eng. Res. Technol. (IJERT) ICESMART, 3(19), <https://doi.org/10.1109/ICSPCS.2015.7391796>.
27. Priyanka TB; Sindhu UD; Varshitha SM; Manjula G (2021), Speech enhancement using hidden markov models and mel-frequency, Int. J. Eng. Res. Technol. (IJERT) NCCDS – 2021 <https://doi.org/10.1007/s10489-014-0629-7>
28. Shashidhar, R.; Patilkulkarni, S.; Puneeth, S.B.: Combining audio and visual speech recognition using LSTM and deep convolutional neural network. Int. j. inf. tecnol. **14**, 3425–3436 (2022). <https://doi.org/10.1007/s41870-022-00907-y>
29. Shashidhar, R.; Patilkulkarni, S.: Visual speech recognition for small scale dataset using VGG16 convolution neural network. Multimed Tools Appl **80**, 28941–28952 (2021). <https://doi.org/10.1007/s11042-021-11119-0>
30. Sooraj, V.; Hardhik, M.; Murthy, N.S.; Sandesh, C.; Shashidhar, R.S.: Lip-reading techniques: a review. Int. J. Sci. Technol. Res. **9**(02), 1–6 (2020)
31. Shashidhar R; S Patilkulkarni; Puneeth SB, (2020) Audio visual speech recognition using feed forward neural network architecture, In: 2020 IEEE International Conference for Innovation in Technology (INOCON), Bangluru, India, 1-5, <https://doi.org/10.1109/INOCON50539.2020.9298429>
32. Rudregowda, S.; Kulkarni, S.P.; Gururaj, H.L.; Ravi, V.; Krichen, M.: Visual speech recognition for Kannada language using VGG16 convolutional neural network. Acoustics **5**(1), 343–353 (2023). <https://doi.org/10.3390/acoustics5010020>
33. Basalamah, S.; Khan, S.D.; Felemban, E.; Naseer, A.; Rehman, F.U.: Deep learning framework for congestion detection at public places via learning from synthetic data. J. King Saud Univ. Comput. Inform. Sci. **35**(1), 102–114 (2023)

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.

