JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

Python-based Action Recognition for Detecting Sign Language

[1] Gudiwaka Vijayalakshmi,[2] Balivada Sai Sree,
[1] Assistant professor, [2] MCA 2nd year,
[1]Master of Computer Applications,
[1]Sanketika Vidya Parishad Engineering College, Visakhapatnam, India

ABSTRACT
Sign language recognition is a complex task that involves a wide range of fields, such as computer vision, computer graphics, natural language processing, human-computer interaction, linguistics, and Deaf culture. Sign language is a primary mode of communication for the speech and hearing-impaired community, and for those who are not familiar with sign language, communicating without an interpreter can be difficult. Sign language recognition involves tracking and recognizing the meaningful gestures made by the human body, including head, arms, hands, fingers, and facial expressions.One technique used to facilitate communication between sign language users and non-sign language users is to translate sign language gestures into spoken language, which can be easily understood by listeners. This is particularly important for people who rely solely on gestural sign language for communication and need to communicate with someone who does not understand sign language.However, most sign language recognition systems face challenges in recognizing gestures accurately, particularly when it comes to variations in skin tone. To address this issue, filters can be introduced to help identify symbols regardless of skin tone.One popular approach for sign language recognition is to use convolutional neural networks (CNNs), which consist of four types of layers: convolution layers, pooling/subsampling layers, nonlinear layers, and fully connected layers. The purpose of these layers is to represent features that the system can learn and use to recognize sign language gestures accurately.In summary, sign language recognition is a complex task that requires expertise in multiple fields. The goal is to create systems that accurately recognize sign language gestures and can translate them into spoken language to facilitate communication between sign language users and non-sign language users.

Keywords: **Sign Language, Hand Gesture, Speech Recognition, Deaf and dump**

INTRODUCTION

Communication is the process of transmitting information from one person, place, or group to another. It involves three elements: the speaker, the message, and the listener, and it is considered successful only when the listener comprehends the message being conveyed. Communication can be categorized into different types, such as formal and informal, oral and written, non-verbal, grapevine[1], feedback, visual, and active listening.Formal communication follows predetermined channels, while grapevine communication is informal and spontaneous. Oral communication can be face-to-face or distant, and written communication involves letters, emails, and other written forms. Non-verbal communication uses gestures, facial expressions, and body language, while feedback communication involves giving feedback on a product or service. Visual communication is when information is obtained from visual sources like television, social media, and other sources. Active listening involves comprehending what the other person is trying to communicate.Sign language is an effective means of communication for people with hearing or speech impairments, but the lack of knowledge of sign language poses a communication barrier between people with normal hearing and those who use sign language. Technology-driven solutions can bridge this gap by translating sign language into a commonly spoken language like English.Research has been done in this field using data gloves, motion capturing systems, sensors, and vision-based sign language recognition (SLR) systems. The existing Indian Sign Language Recognition system was developed using machine learning algorithms, achieving high accuracy but lacking real-time capabilities.This paper aims to develop a real-time SLR system using TensorFlow object detection API and a dataset generated using a webcam. The paper discusses related work, data acquisition and generation, the methodology[2] used to develop the system, experimental evaluation, and concludes with future work

PROBLEM STATEMENT

The World Health Organization (WHO) estimates that approximately 63 million people in India have significant hearing impairments[3], which equates to about 6.3% of the population. Meanwhile, the National Association of the Deaf in India estimates that around 18 million people, or nearly 1% of the Indian population, are deaf. Therefore, there is a crucial need for a communication system to enable speech-impaired[4] and deaf individuals[5] to communicate effectively with others.

EXISTING SYSTEM

The previous system utilized a flex sensor module to aid communication for individuals with speech impairments. The user's hand is attached to the flex sensors, which react to the bending of each finger. The controller then responds with pre-recorded speech corresponding to each unique flex sensor[6] value. However, this system only supports a limited number of alphabet signs[7] and does not provide assistance for words or sentences. Additionally, the accuracy obtained is considerably low.

Figure: existing system

PROPOSED MODEL

The proposed system is designed to enable people with speech or hearing impairments[8] to communicate through sign language. The user inputs a gesture or sign image, which is processed using Matlab's image processing technique to classify the input based on a given dataset. The system then generates a voice output and displays the recognized sign language in text format. This prototype[9] is intended to demonstrate the feasibility of converting sign language into speech and text and provide a practical application for improving communication between deaf and mute individuals. The goal of this paper is to introduce an image processing algorithm-based solution that addresses the communication challenges faced by people with hearing and speech disabilities.

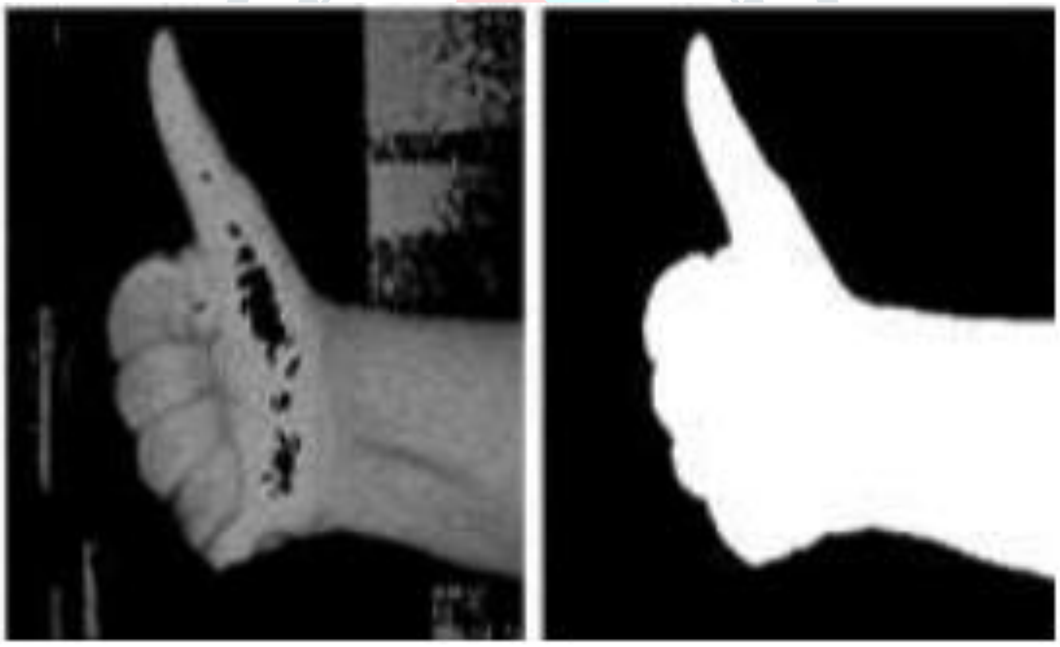


Figure: proposed model

Pre-Processing

Understanding aspect ratios

Aspect ratio describes the proportional relationship between the width and height of an image, defining its shape. It is represented as a width-to-height formula. For instance, a square image has an aspect ratio of 1:1, as its height and width are equal. This ratio remains constant regardless[10] of the size of the image, be it 500px x 500px or 1500px x 1500px. On the other hand, a portrait-style image may have an aspect ratio of 2:3, with the height being 1.5 times longer than the width. In this case, the image could have dimensions of 500px by 750px, 1500px by 2250px, and so on.

Cropping to an aspect ratio

Aspect ratio describes the proportional relationship between the width and height of an image, defining its shape. It is represented as a width-to-height formula. For instance, a square image has an aspect ratio of 1:1, as its height and width are equal. This ratio remains constant regardless of the size of the image, be it 500px x 500px or 1500px x 1500px. On the other hand, a portrait-style image may have an aspect ratio of 2:3, with the height being 1.5 times longer than the width. In this case, the image could have dimensions of 500px by 750px, 1500px by 2250px, and so on.

Image scaling

Image scaling[11] refers to the process of resizing a digital image[12] in computer graphics or digital imaging. In video technology, upscaling[13] or resolution enhancement[14] is used to describe the magnification[15] of digital content. Scaling a vector graphic image involves using geometric transformations to scale the visual primitives that make up the image without compromising image quality. On the other hand, scaling a raster graphics image requires generating a new image with either a larger or lower number of pixels. When scaling down, there is often a noticeable decrease in image quality. Scaling raster graphics is an example of two-dimensional sample-rate conversion, which is the conversion of a discrete signal from one sampling rate to another

Segmentation

Image segmentation[16] is a crucial process in computer vision that involves partitioning a digital image into multiple segments or regions, often represented as sets of pixels or image objects. This is done to simplify and modify an image's representation, making it easier to analyze and understand. Today, modern image segmentation methods are powered by deep learning technology. But why is image segmentation so important? For instance, autonomous vehicles rely on sensory input devices like cameras, radar, and lasers to help the vehicle understand the environment and create a digital map. Without image segmentation, which involves picture classification and segmentation, autonomous driving is not possible. So how does image segmentation work? Essentially, it involves converting an image into a collection of pixel segments[17], represented by a mask or a labeled image. By segmenting an image, we can process only the important parts of the image instead of the entire image. There are several techniques for image segmentation, including identifying abrupt discontinuities in pixel values that often represent edges that define a region, or searching for similarities in different parts of an image, followed by strategies such as region expansion, clustering, and thresholding. Over the years, various approaches to image segmentation have been developed, all of which rely on domain-specific information to solve segmentation problems in particular application areas.

Algorithm

In this case we are using CNN algorithm

CNN Algorithm

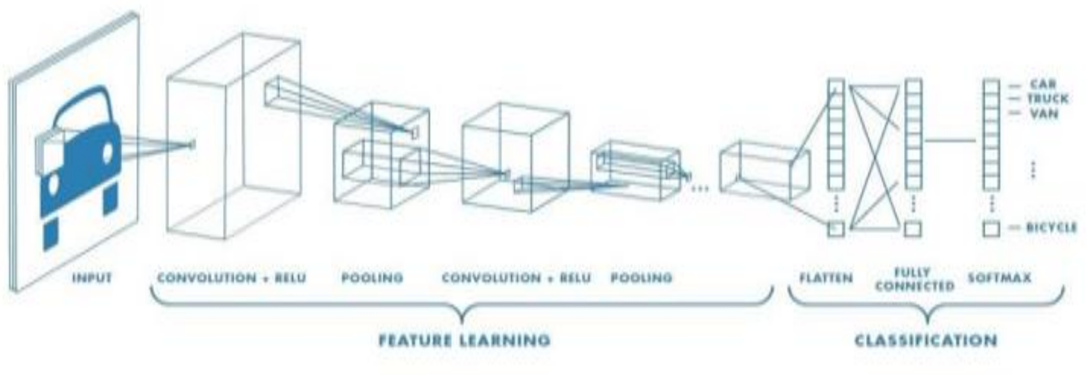


Figure: CNN algorithm

The CNN, or Convolutional Neural Network[18], is a type of deep learning model designed to classify images by extracting key features from them. Compared to other classification algorithms, CNNs require minimal pre-processing of the input image

It is made up of four layers.

a) Convolution Layer
b) ReLu Layer
c) Pooling Layer

d) Fully Connected Layer

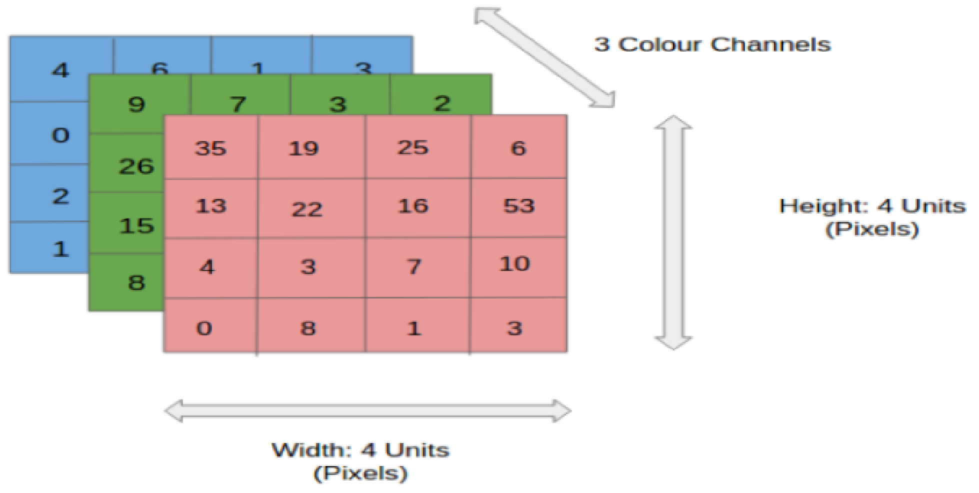


Figure: pixels structure

This image is composed of three-color channels: Red, Green, and Blue. Images can be represented in various color spaces, such as Grayscale, RGB, HSV, CMYK, and more. As image dimensions increase, such as in the case of 8k resolution (7680x4320), the CNN algorithm plays a crucial role in compressing the image into a more manageable format for processing while retaining essential features for accurate predictions.

Convolution Layer

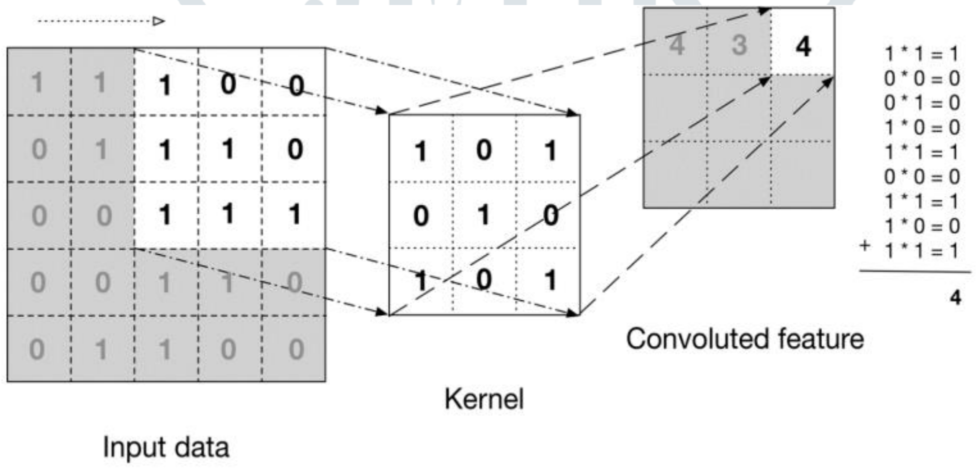


Figure: convolution layer

Grayscale images[19] can be used in convolutional neural networks (CNNs) to extract features. To do this, a filter or kernel is applied to the input image, resulting in a convolved feature. Filters typically come in sizes of 3x3 or 5x5. For example, when using a 3x3 filter, the filter matrix is multiplied with the 3x3 input array matrix, and the resulting value is stored. This is done for each 3x3 matrix in the input array, and the results are saved to create the output of the filter, which is a 3x3 matrix. Therefore, when using a 3x3 filter with a 5x5 input array matrix, the output will be a 3x3 matrix.

ReLu Layer
The Rectified Linear Unit (ReLU) is a function that scales the parameters to non-negative values. In this layer, any negative pixel values are turned into 0's. By using the ReLU function, we can make our images more non-linear, which is important because images themselves are non-linear. The ReLU function helps to break up the linearity that may be introduced during the convolution process. It removes all the dark parts from an image, leaving only the positive values (the grey and white colors). The main difference between the non-rectified and rectified versions of an image is that the latter has all negative pixel values set to 0.

Pooling Layer
The pooling (POOL) layer serves to decrease the height and width of the input, leading to reduced computation and improved invariance of feature detectors to input position. This procedure contributes to the convolutional neural network's ability to handle spatial variance. Additionally, pooling helps to reduce image size and parameter count, preventing the problem of overfitting. When a pooling layer[20] is applied, the result is down sampled or pooled feature maps that provide a summarized version of the discovered features in the input.

Fully Connected Layer
The main goal of a convolutional neural network is to take the input and extract features that can help identify images more accurately. The network is composed of interconnected neurons that activate when they recognize specific patterns and transmit signals to the output layer, which assigns output classes based on weight values. The loss function is used to determine the accuracy of the network, which

can be improved through optimization. In the fully connected layer, each neuron detects a specific feature, such as a nose, and preserves its value, which is then communicated to the output classes for trained images.

IMPLEMENTATION

To complete this project, we divided it into 3 parts

1)Dividing Dataset

2)Creating a CNN model

3)Predicting Sign

Dividing Dataset

To develop our model, we started with a dataset of handwritten numbers from 0-9, consisting of 1500 images per number. We split the dataset into a ratio of 80:20, resulting in 1200 images for training and 300 images for testing. To organise the dataset, we utilised the OS library to create directories and folders. We created a folder named "ISL Dataset," within which we created a "Train-Test" folder. Within the "Train-Test" folder, we created two subfolders named "Train" and "Test" (as depicted in Fig.1). In each of these subfolders, we created ten folders representing the numbers 0-9. This organisation allowed us to easily feed the data into our model for training and testing.

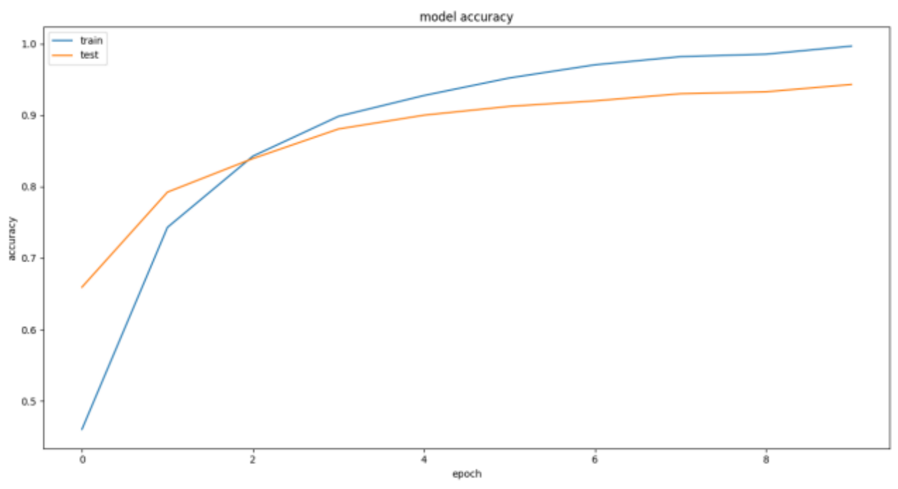
Creating a CNN model

We used Kera's' ImageDataGenerator to load the training and testing data sets, utilizing the flow from directory function, which lets us use class names as folder names for the imported images. We created a function using Matplotlib to display images from the data sets in batches for testing purposes. Using a trial-and-error approach, we constructed a CNN model with a sequential structure to achieve a reasonable accuracy level. We then applied the trained model to predict signs in the last phase and saved the model. During training call-backs, the LR is decreased on a plateau and early stopping is employed, both of which are based on the validation dataset loss. The validation accuracy and loss are calculated after each epoch of training. If the validation loss does not decrease, the model's LR is lowered using the Reduce LR method to prevent overshooting the loss minima. We also use the early stopping algorithm, so if the validation accuracy continues to decrease for several epochs, the training is stopped. After constructing the model, we trained it for ten epochs of training data. We then performed visualization and a quick test to verify that the model is working as intended when detecting live video feeds. The word dict is a dictionary containing the label names for all upcoming labels. Finally, we utilized Matplotlib to plot and summarize the model's accuracy and loss.

Predicting *Sign*

In this section, we begin by defining a bounding box to identify the region of interest (ROI) and calculating the accumulated average to detect any foreground item. Next, we search for the maximum contour, and if one is found, it indicates that a hand has been detected, and we use the ROI threshold as a test image. We load the saved model using Keras models and feed the hand's ROI threshold image as input into our predictive model. To accomplish this, we wrote a function that computes the accumulated weighted average of the background. We then wrote another function to segment the hand, which involves obtaining the maximum contours and the thresholder

Model Accuracy after successful training



Figures: Representing Model Accuracy

Model Loss after successful training

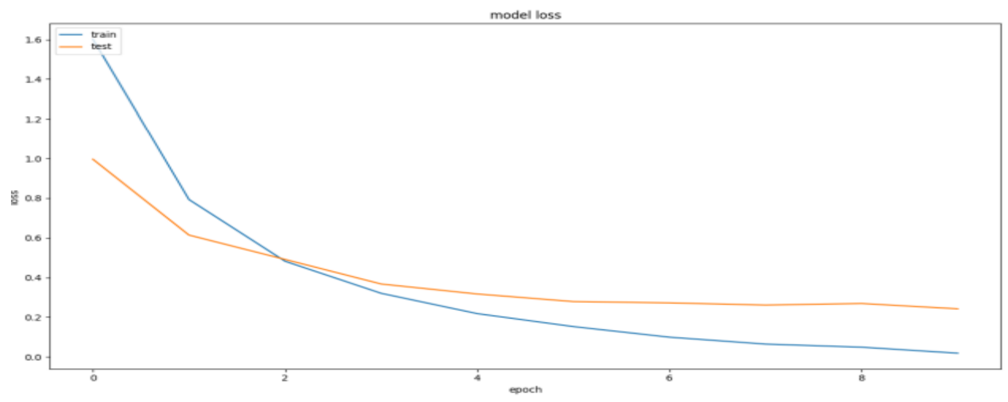


Figure: *representing Model Loss*

CONCLUSION

The aim of our project is to bridge the gap for visually impaired individuals by introducing a low-cost computer into the communication chain to automatically collect and recognize sign language. With the need for various types of images as data sources for explanation and analysis in today's applications, several features must be retrieved to carry out various tasks. The convolutional neural network's goal is to achieve the most accurate categorization possible. A sign language recognition system is a powerful tool when it comes to developing expert knowledge, edge detection, and merging erroneous information from numerous sources. Additionally, the suggested sign language recognition system can expand to recognize gestures and facial expressions alongside sign language. Presenting translations as more suitable language sentences rather than labels is more acceptable to users and improves text readability. Increasing the scope of different sign languages is possible by feeding more training data to the system to enhance detection accuracy. This project has the potential to develop to include sign-to-voice conversion

Future Scope:
There are two potential avenues for further development in this project. Firstly, we can create a model that is capable of recognising Indian Sign Language (ISL) at both the word and sentence level by incorporating temporal analysis to detect changes over time. Secondly, we can expand this project to create a comprehensive product that addresses the needs of the speech and hearing-impaired community, thus helping to bridge the communication gap that exists for these individuals.

REFERENCES

[1]AN Article Reference of grapevine

https://www.sciencedirect.com/science/article/pii/S2212977416300205

[2] AN Article Reference of the methodology

https://www.sciencedirect.com/science/article/abs/pii/S0951524096000018

[3] AN Article Reference of hearing impairments

https://www.tandfonline.com/doi/abs/10.1080/14992020500060370

[4] AN Article Reference of speech-impaired

https://www.sciencedirect.com/science/article/abs/pii/S0165587603003033

[5] AN Article Reference of deaf individuals

https://psycnet.apa.org/record/1994-26294-001

[6] AN Web Reference of flex sensor

https://onlinelibrary.wiley.com/doi/abs/10.1002/smll.201602790

[7] AN Web Reference of alphabet signs

https://www.mdpi.com/1424-8220/21/17/5856

[8] AN Web Reference of impairments

https://journals.sagepub.com/doi/abs/10.1177/0018720817708397?journalCode=hfsa

[9] AN Web Reference of prototype

https://www.sciencedirect.com/science/article/abs/pii/S0956713508003307

[10] AN Web Reference of regardless

https://onlinelibrary.wiley.com/doi/abs/10.1002/iroh.201601851

[11] A Book Reference of Image scaling

https://dl.acm.org/doi/abs/10.5555/984432

[12] A Book Reference of digital image

https://dl.acm.org/doi/abs/10.5555/4901

[13] A Book Reference of upscaling

https://onlinelibrary.wiley.com/doi/abs/10.1002/fld.267

[14] A Book Reference of resolution enhancement

https://www.spiedigitallibrary.org/conference-proceedings-of-spie/5377/0000/Resolution-enhancement-technology--the-past-the-present-and-extensions/10.1117/12.548923.short?SSO=1

[15] A Book Reference of magnification

https://ieeexplore.ieee.org/abstract/document/607494/

[16] AN Article Reference of Image segmentation

https://peerj.com/articles/453/?report=reader&utm_source=TrendMD&utm_campaign=PeerJ_TrendMD_1&utm_medium=TrendMD

[17] AN Article Reference of pixel segments

https://www.hindawi.com/journals/jhe/2018/3640705/

[18] AN Web Reference of Convolutional Neural Network

https://dl.acm.org/doi/abs/10.1145/2567948.2577348

[19] AN Web Reference of Grayscale images

https://dl.acm.org/doi/abs/10.1145/1597990.1598049

[20] A Book Reference of a pooling layer

https://link.springer.com/chapter/10.1007/978-3-030-21005-2_23

BIBLOGRAPHY

Gudiwaka vijayalakshmi working as a Assistant professor in Master of computer application sanketika vidya parishad engineering college, visakhapatanam Andhra pradesh. with 2 years of experience in Computer science and engineering (CSE) , accredited by NAAC. With her area of interests in java, python,.Net, HTML.

Balivada Sai Sree is studying his 2nd year, Master of Computer Applications in Sanketika Vidya Parishad Engineering College, affiliated to Andhra University, accredited by NAAC. With her interest in python, web development and as a part of academic project, Python-based Action Recognition for Detecting Sign Language used web-based Python speech to text converter. As a result of a desire to comprehend. In completion of his MCA.