SIGN LANGUAGE RECOGNITION

by

İlke SAVAŞOĞLU

Engineering Project Report

Yeditepe University

Faculty of Engineering

Department of Computer Engineering

2022

SIGN LANGUAGE RECOGNITION

APPROVED BY:

Prof. Dr. Emin Erkan Korkmaz (Supervisor)	. . . .
Assoc. Dr. Dionysis Goularas	. . . .
Asst. Prof. Funda Yıldırım	. . . .

DATE OF APPROVAL: .../.../2022

ACKNOWLEDGEMENTS

First of all I would like to thank my advisor Emin Erkan Korkmaz for his guidance and support throughout my project.

Also I would like to thank my parents for their support and encouragement throughout my education up to the present.

ABSTRACT

SIGN LANGUAGE RECOGNITION

Sign Language is a visual language that people with hearing or speech disabilities create by using hand gestures as a whole to communicate with each other. Since it is very difficult for hearing mute and deaf people to communicate in daily life, it is difficult for people to communicate with them. When communicating with a person who does not know sign language, the fact that person does not understand them poses a significant problem for these persons. Another problem is that they feel insecure to go out alone because they cannot tell their problems to any government agency or doctor. This makes life very difficult for them. My project aims to enable mute and deaf people to communicate with ordinary people who do not know sign language in daily life through the application. In this way, the communication barrier between people will be removed and mutual communication will be possible thanks to the application. From the sign language course, I took last term, I will create a data set of images by making sign language expressions myself and ensure that they are recognized by machine learning algorithm. The application I will make using machine learning will facilitate their communication with people who do not know sign language and increase their self-confidence.

ÖZET

TITLE OF THE PROJECT

İşaret Dili, işitme veya konuşma engelli kişilerin birbirleriyle iletişim kurmak için bir bütün olarak el hareketlerini kullanarak oluşturdukları görsel bir dildir. İşitme engellilerin günlük hayatta iletişim kurmaları çok zor olduğu için insanların onlarla iletişim kurmaları da zordur. İşaret dili bilmeyen bir kişiyle iletişim kurarken, kişinin onu anlamaması bu kişiler için önemli bir sorun teşkil etmektedir. Bir diğer sorun da, sorunlarını herhangi bir devlet kurumuna veya doktora anlatamadıkları için tek başlarına dışarı çıkma konusunda kendilerini güvensiz hissetmeleridir. Bu onlar için hayatı çok zorlaştırır. Projem, işitme engellilerin uygulama üzerinden işaret dili bilmeyen insanlarla günlük hayatta iletişim kurmasını sağlamayı amaçlıyor. Bu sayede kişiler arasındaki iletişim engeli ortadan kalkacak ve uygulama sayesinde karşılıklı iletişim mümkün olacaktır. Geçen dönem aldığım işaret dili kursundan, işaret dili ifadelerini kendim yaparak bir veri seti oluşturacağım ve bunların makine öğrenmesi algoritması tarafından tanınmasını sağlayacağım. Makine öğrenimi kullanarak yapacağım uygulama, işaret dili bilmeyen insanlarla iletişimlerini kolaylaştıracak ve özgüvenlerini artıracaktır.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS/ABBREVIATIONS

TSLI	Turkish Sign Language
ASL	American Sign Language
CNN	Convolutional Neural Network
AI	Artificial Intelligence
RGB	Red, Green, and Blue
IBM	International Business Machines Corporation
GAN	Generative Adversarial Networks
GPU	Graphics Processing Unit

1. INTRODUCTION

Deaf and mute people use hand signs to explain themselves. Ordinary people have difficulty understanding what they mean by using hand signs. For this reason, there is a need for an application that predicts sign language and transfers it in written form to people who do not know sign language. It solves the communication problem between mute and deaf people and ordinary people.

1.1. Sign language

Sign Language is a kind of communication that incorporates hand gestures and other body parts, as well as face expressions and postures. It is mostly used by mute and deaf people. Sign language is not international, contrary to common notion[1]. There are several sign languages, such as Turkish, Bulgarian, and Chinese sign languages[1]. Turkish Sign Language is difficult for Chinese Sign Language users to understand, and vice versa. Turkish Sign Language is a complete natural language with the same linguistic qualities as spoken languages and a different grammar than Turkish. It is the main language of several deaf and mute Turks, as well as many hearing Turks. These, like spoken languages, differ from one region to another. If we need to give an example for Turkey, many gestures are changing in Istanbul and Ankara. A sign language is a language that combines hand forms, orientation and movement of the hands, arms, or torso, facial emotions, and lip patterns to convey meaning instead of using words[2]. A working sign recognition system might allow the deaf and mute people to interact with non-signing people without requiring the need for an interpreter. The project aims to develop a system that can accurately classify recognizing sign language.

Figure 1.1: Hand Gesture

1.1.1. Sign Language and Gesture Recognition

The process of turning the user's signs and motions into text is referred to as sign language recognition. It helps persons who are unable to communicate with the broader public. People who are deaf or mute are frequently cut off from regular social interaction[3]. It has been found that they have difficulty interacting with regular people with their gestures at times, as just a handful of them are recognized by the majority of people. Because persons with hearing loss or who are deaf are unable to communicate verbally, they must rely on visual communication the majority of the time. In the deaf and mute population, sign language is the major mode of communication. It contains syntax and vocabulary much like any other language, but it communicates through visual means. The problem arises when deaf and mute people try to express themselves to other people with the help of these sign language grammar. This is due to the fact that most individuals are not aware of this grammar. As a result, it has been observed that a deaf person's communication is confined to his or her family or the deaf community. The increased public support and funding for a worldwide endeavor emphasize the relevance of sign language. The deaf population is in desperate need of a computer-based system in this technological age. However, academics have been working on the issue for a long time, and the findings are promising. Although interesting technologies for voice recognition are being developed, there is currently no commercial solution for sign recognition on the market. The goal is to produce a user-friendly real-time sign language translator by teaching computers to understand human language. Some steps toward this goal include teaching a computer to recognize speech, facial emotions, and human gestures. Nonverbally communicated information is referred to as gestures. At any one time, a human may make an infinite number of gestures[4]. Computer vision researchers are particularly interested in human gestures since they are interpreted through vision[5]. The project aims to recognize sign language gestures by using a Convolutional neural network. Recognizing these movements using machine language requires a complicated programming algorithm. Therefore, the project focused on a Convolutional neural network for a more efficient output generation.

1.2. Terms

- Deaf individuals utilize *Sign language*, which is a form of communication based on visual motions and signals.
- A *Gesture* is a movement of a bodily part, most often the hand or the head, to communicate an idea or meaning.
- *Fingerspelling* is the use of only the hands to represent the letters of a writing system, as well as occasionally number systems.
- *Deaf-mute* a person who is both deaf and unable to speak.
- The *Alphabet* is a group of letters or symbols in a fixed order that are used to represent a language's core set of spoken sounds, particularly the letters from A to Z.

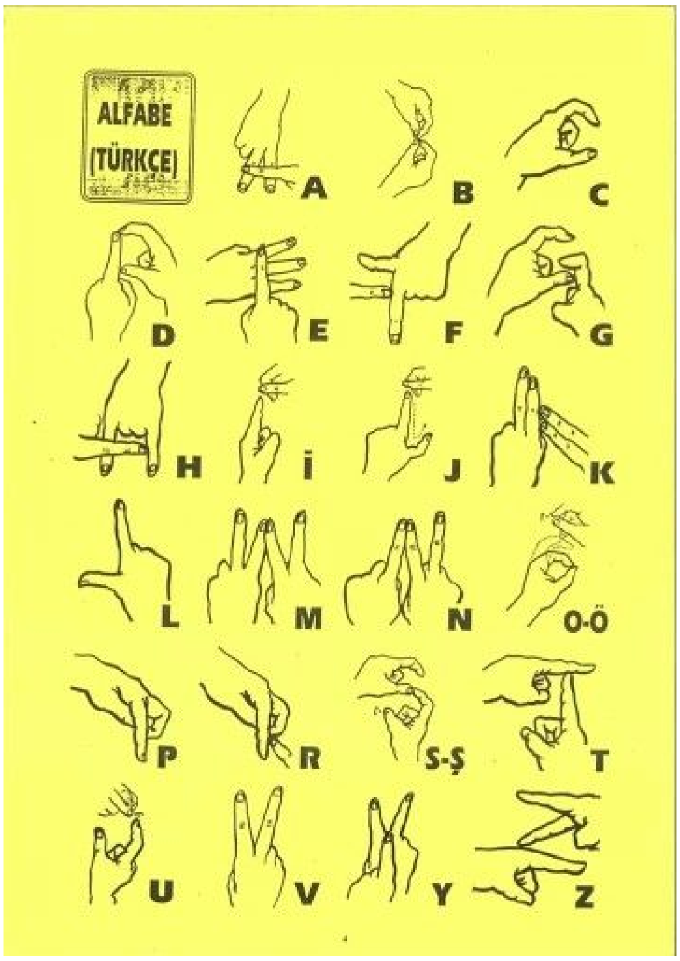


Figure 1.2: Turkish Sign Language Alphabet

1.3. Motivation

According to the data published by TUIK for the disabled in Turkey in 2011, there are 836 thousand deaf and mute, including 406 thousand men and 429 thousand women[6]. These statistics formed the motivation for the project. Deaf and mute people need a proper channel to communicate with ordinary people. Most people cannot understand the sign language of people with disabilities. Therefore, the project aims to transform sign language gestures into readable text for ordinary people.

1.3.1. Accessibility

Individuals or organizations that do not have sufficient sign language knowledge can use this software to communicate with a deaf person.

1.4. Scope and Limitations

- Since the gestures are so similar and their meanings vary from region to region, it is very difficult to generate data on gestures and write a model to distinguish them from one another.
- Adding all the gestures is a problem, as there is not enough data on the internet that includes Turkish Sign Language movements.

1.5. Problem Definition

Hand signs and gestures are used by those who are unable to speak. Ordinary individuals have a hard time deciphering such gestures. As a result, a system must recognize various signs and gestures and communicate the information to regular people. Project acts as a link between deaf persons and the general public.

1.6. Requirements

For the designed model to work correctly, it needs to achieve high accuracy and low loss rate when it finishes learning. After learning, it needs to predict words and phrases in real-time detection correctly.

1.6.1. Necessary softwares and hardwares are:

- Windows
- Python
- Machine learning related libraries
- Database about Turkish Sign Language Gestures

2. BACKGROUND

In this chapter, Artificial Intelligence, Machine Learning, Deep Learning and Convolutional Neural Network subjects, which are the heart of the project, are explained with examples and their histories are mentioned superficially.

2.1. What is a Artificial Intelligence?

Artificial intelligence is the ability of computers to accomplish certain activities that need human or animal intellect. This term is commonly attributed to the fathers of the field, Marvin Minsky and John McCarthy, who lived in the 1950s[7]. Artificial intelligence enables machines to comprehend and accomplish specified objectives. Deep learning is a type of machine learning that is included in AI. The former refers to machines that learn from existing data without the need for human intervention. Deep learning enables a machine to process massive volumes of unstructured data such as text, photos, and audio[7].

2.2. Examples of Artificial Intelligence

- Siri, Alexa and other smart assistants
- Self-driving cars
- Robo-advisors
- Conversational bots
- Email spam filters
- Netflix's recommendations

2.3. What is machine learning?

Artificial intelligence (AI) application machine learning offers several astonishing features. Unsupervised machine learning is possible with the help of a machine learning algorithm. Through the input of previous data, the algorithm might appear to get ”smarter” and more precise at forecasting events without being expressly programmed. The mathematical modeling of neural networks provided the inspiration for machine learning. In 1943, logician Walter Pitts and neurologist Warren McCulloch produced a study in which they sought to quantitatively map out human cognition's cognitive processes and decision-making. Alan Turning proposed the Turing Test in 1950, and it became the litmus test for determining whether machines were ”intelligent” or ”unintelligent.” The capacity to persuade a human being that it, the machine, was likewise a human being was required for a machine to be classified as ”intelligent.” Soon after, Dartmouth College's summer research program became the formal genesis of AI. From this time on, ”intelligent” machine learning algorithms and computer programs began to arise, capable of performing everything from arranging salespeople's travel routes to playing games like checkers and tic-tac-toe with humans. Intelligent robots went on to achieve anything from use voice recognition to learn how to pronounce words like a baby to beating a global chess grandmaster in his own field[8].

2.4. Examples of Machine Learning

Machine learning is relevant to many fields and industries and has the ability to grow over time[9]. Here are three real-world instances of machine learning in action.

2.4.1. Classification

Classification aids in the analysis of an object's measurements in order to determine the category to which it belongs. Analysts utilize data to develop a productive relationships. Before deciding to disburse loans, a bank, for example, examines the customers' ability to repay them. We can accomplish that by taking into account aspects such as the customer's wages, savings, and financial history. This information is derived from the loan's historical data[10].

2.4.2. Image Recognition

One of the most popular applications of machine learning is image identification. You can classify an item as a digital picture in a variety of circumstances. The intensity of each pixel, for example, is one of the metrics in a black and white picture. Each pixel in a colorful picture produces three intensity readings in three separate colors: red, green, and blue (RGB). Face detection in images may also be accomplished via machine learning. Each individual in a database of numerous persons has their own category. Character recognition, which distinguishes handwritten and printed letters, is also done with machine learning. A piece of text may be divided into smaller pictures, each of which contains a single word.[10].

2.4.3. Medical Diagnosis

Machine learning may be applied to approaches and systems that aid in an illness diagnosis. It is used to analyze clinical data and their combinations for prognosis, such as illness progression prediction, as well as to retrieve medical knowledge for outcome research, therapeutic management, and monitoring of patients. All of those are examples of successful machine learning implementations. It can aid in the integration of computer-based healthcare systems[10].

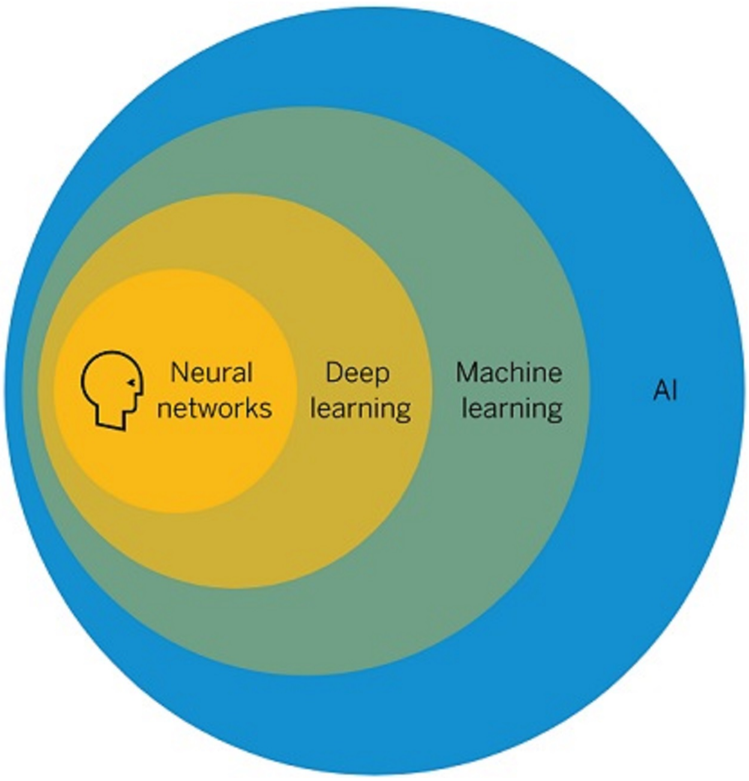


Figure 2.1: Artificial Intelligence

2.5. What is Deep Learning?

Deep learning is a type of machine learning methodology[11] that, like the majority of machine learning approaches, iteratively modifies itself when it achieves a predefined cutoff point[12]. Deep learning models were born out of artificial neural networks; [12] a complex [11] interconnection of very simple processing nodes [11] organized into successive layers [11]; an input layer, an output layer, and hidden layers[11]–[13]. It was revealed that by adding additional hidden layers to a neural network, hierarchical patterns might be learned from the input, significantly improving the prediction capacity of the neural network[14].

2.5.1. History of Deep Learning

Warren McCulloch, a neurophysiologist, and Walter Pitts, a mathematician, published a study in the 1940s on how biological neurons could operate[15]. McCulloch and Pitts put their idea to the test, and in 1943, they succeeded[15], Electrical circuits were used to create the first artificial neural network[15], [16]. Computers were smart enough to model a neural network in the 1950s, about the same time that Alan Turing released his AI paper on the Turing Test[15]. In 1957, Frank Rosenblatt proposed the Perceptron, an enhanced artificial neuron architecture[16], It has garnered a lot of traction due to its simple and effective learning algorithm[16].ADALINE (Adaptive Linear Elements), another artificial neuron built by Widrow and Hoff in the 1960s, employed a different learning mechanism[15]. They built the first neural network that was utilized to address a real-world problem when they joined Multiple ADALINE neurons (MADALINE) together, and it is still being used commercially to this day[15].

The 1970s would be recognized as the Artificial Intelligence era. Winter is known for its unexpected disillusionment and loss of interest in AI[17], This resulted in a reduction in funding for AI research, resulting in a brief halt in AI progress[17]. Previous triumphs have led to an overestimation of AI's capabilities, and promises were ultimately broken[15]. The AI boom has also sparked certain ethical and philosophical problems in people's minds, leading to an increase in dread and skepticism of AI[15]. Nonetheless, at the time, neural network research was not the dominant AI paradigm. Heuristic programming advances have mostly overtaken neural network advances[15] They will become less well-known as a result of this. Those who had heard of them favored more traditional, linear von neumann computational structures, which required less memory and were less computationally costly than complicated, distributed, parallel computational designs such as neural networks[15]. Marvin Minsky and Seymour Papert's book ”Perceptron” proved that neural networks couldn't handle non-linearly separable data and were restricted in the issues they could answer. This gave AI researchers the impression that neural networks were a theoretical dead end[16]. Despite the fact that the world was not ready for neural networks, a tiny number of AI researchers known as the ”Parallel Distributed Processing” research group continued to work on them despite the lack of funding[18]. They quickly overcome the constraints described in ”The Perceptron” by expanding single-layered neural networks to multi-layered neural networks, with the first multi-layered neural network being implemented in 1975[15]. ”Neural networks began competing with Support Vector Machines,” claiming that they could provide better results with the same quantity of data. Rumelhard, McClelland, and the PDP Research Group's book ”Parallel Distributed Processing” demonstrated that multi-layered neural networks were significantly more resilient and could be used to learn a wide range of functions.

There was still one big stumbling block to neural net research: the learning methods that worked so well for shallow neural networks didn't scale effectively for deeper networks[15]. Backpropagation was first introduced to multilayered neural networks in the late 1970s by Rumelhart, Williams, and Hinton as an efficient and scalable learning technique[15]. It was the discovery that allowed deep learning to take off by making it efficient and rapid, and it has since become the most well-known gradient descent method[18]. Interest in AI has resurfaced, and AI research has resurfaced, with deep learning leading the way[18]. When Japan launched a fifth-generation effort in neural networks, the US became concerned that they would be left behind and began pouring money into the area[15]. Deep learning has ”increasingly taken over AI activities, ranging from language understanding, audio and picture recognition, machine translation, planning, and even game playing and autonomous driving” since that time. Yan LeCunn proved the potential of the backpropagation method at Bell Labs in the early 1900s by training a deep neural network to read handwritten digits, and IBM's DeepBlue defeated chess champion Gary Kasparov. Backpropagation momentarily went out of favor in the 2000s when GPUs were introduced for faster processing but were prohibitively costly.

2.5.2. Examples of Deep Learning

By 2010, GPUs had become faster and more inexpensive, and IBM Watson had won the 1 million dollar first-place award on Jeopardy. DeepMind taught computers to play classic Atari games using deep reinforcement learning in 2015. ImageNet's automatic labeling error rate fell from 28 to less than 3 in 2016, making it more accurate than humans. DeepMind's AlphaGo defeated the world Go champion 4–1, then AlphaGo Zero defeated the original AlphaGo 100–0 not long after. In 2017, an AI system was able to classify skin cancer on par with dermatologists. For a restricted domain, IBM and Microsoft both achieved human-level voice recognition. Microsoft reached human-level machine translation quality for Chinese-to-English translation in 2018[14].

2.6. What is Convolutional Neural Network?

A convolutional neural network is a deep learning network design that learns directly from input, removing the requirement for feature extraction by hand. CNNs are particularly effective for recognizing objects, people, and sceneries by looking for patterns in pictures. They're also useful for categorizing non-image data including audio, time series, and signal data. CNNs are frequently used in applications that need object identification and computer vision, including as self-driving vehicles and facial recognition[19].

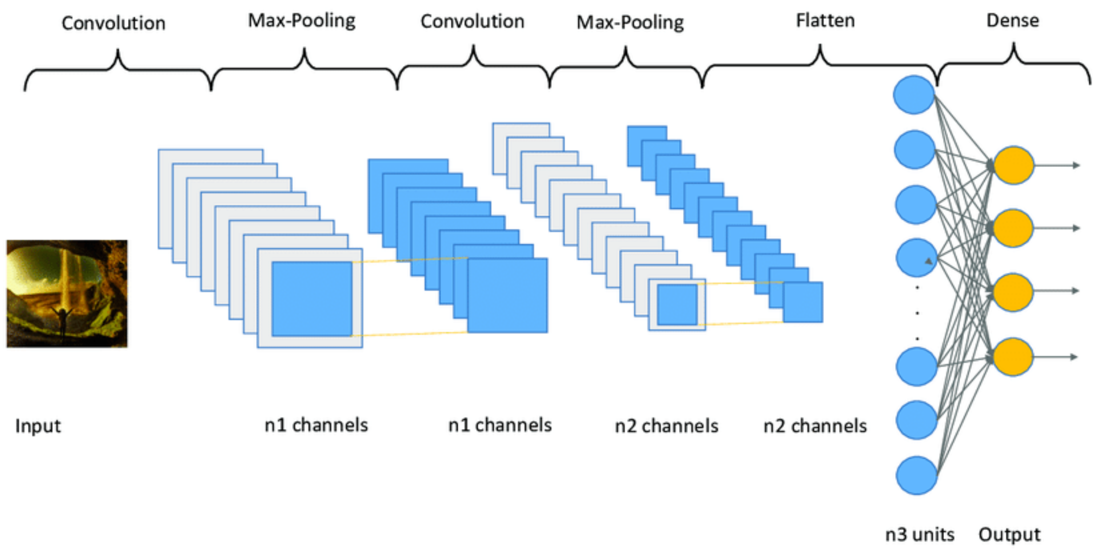


Figure 2.2: Convolutional Neural Network

2.6.1. What Makes CNNs So Useful?

The use of CNNs for deep learning is popular for four reasons:

- CNNs do away with the requirement for feature extraction by hand. The CNN learns the characteristics immediately[20].
- The outputs of CNNs are quite accurate in terms of recognition[20].
- CNNs may be retrained to do new jobs, allowing you to expand on current networks[20].
- CNNs are the most effective architecture for detecting and learning significant characteristics in image and time series data[20].

CNNs are a critical component of applications such as:

- Medical Imaging: CNNs can visually detect the presence or absence of cancer cells in pictures by examining thousands of pathology reports[20].
- Audio Processing: Term detection may be used in any device with a microphone to recognize when a certain word or phrase ('Hey Siri!') is spoken. Regardless of the

surroundings, CNNs can reliably learn and recognize the keyword while disregarding all other sentences[20].

- Stop Sign Detection: CNNs are used in automated driving to identify the existence of a sign or other item and make judgments based on the results[20]. Synthetic Data Generation: New pictures may be created using Generative Adversarial Networks (GANs) for use in deep learning applications such as facial recognition and automated driving[20].

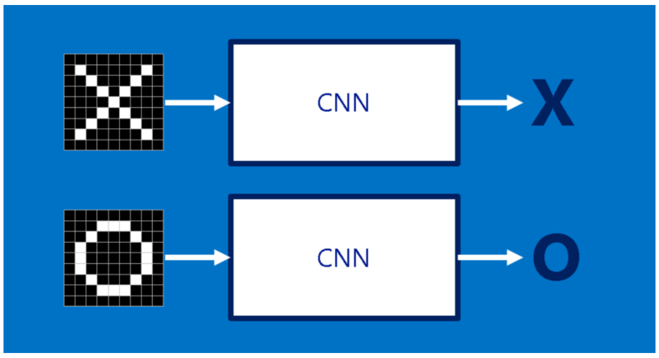


Figure 2.3: Convolutional Neural Network Example

3. ANALYSIS AND DESIGN

This chapter describes how the interface, dataset, model, image processing, and gesture determination are implemented and provides information on what they include. The complete process can be seen on Figure 3.1

3.1. Interface

The interface is designed so that deaf users can use the camera to make gestures in a specific area of the application in real-time. A person who does not know sign language can translate the signs into sentences or words. People will make this choice through the buttons on the interface, and the meaning of the translation will be written on the screen according to their choice.

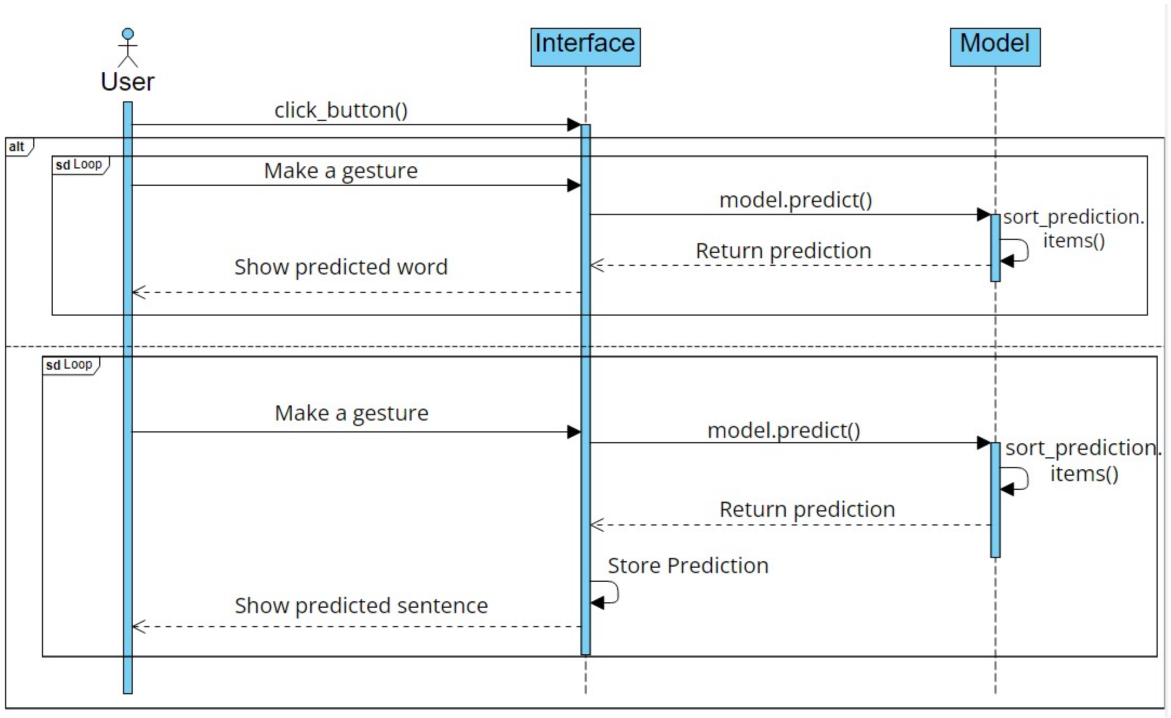


Figure 3.1: User Interface Interaction

3.2. Dataset

A new dataset is created for this project for the training and testing processes. A dataset of approximately 500 images was created for each category. Two different people have created these images. A program was written to create the dataset more easily. While creating this data set, it was tried to be captured from every angle so that the gesture could be better understood in the model and not cause problems with other gestures. This dataset is ready to add more data and more categories with the written code.

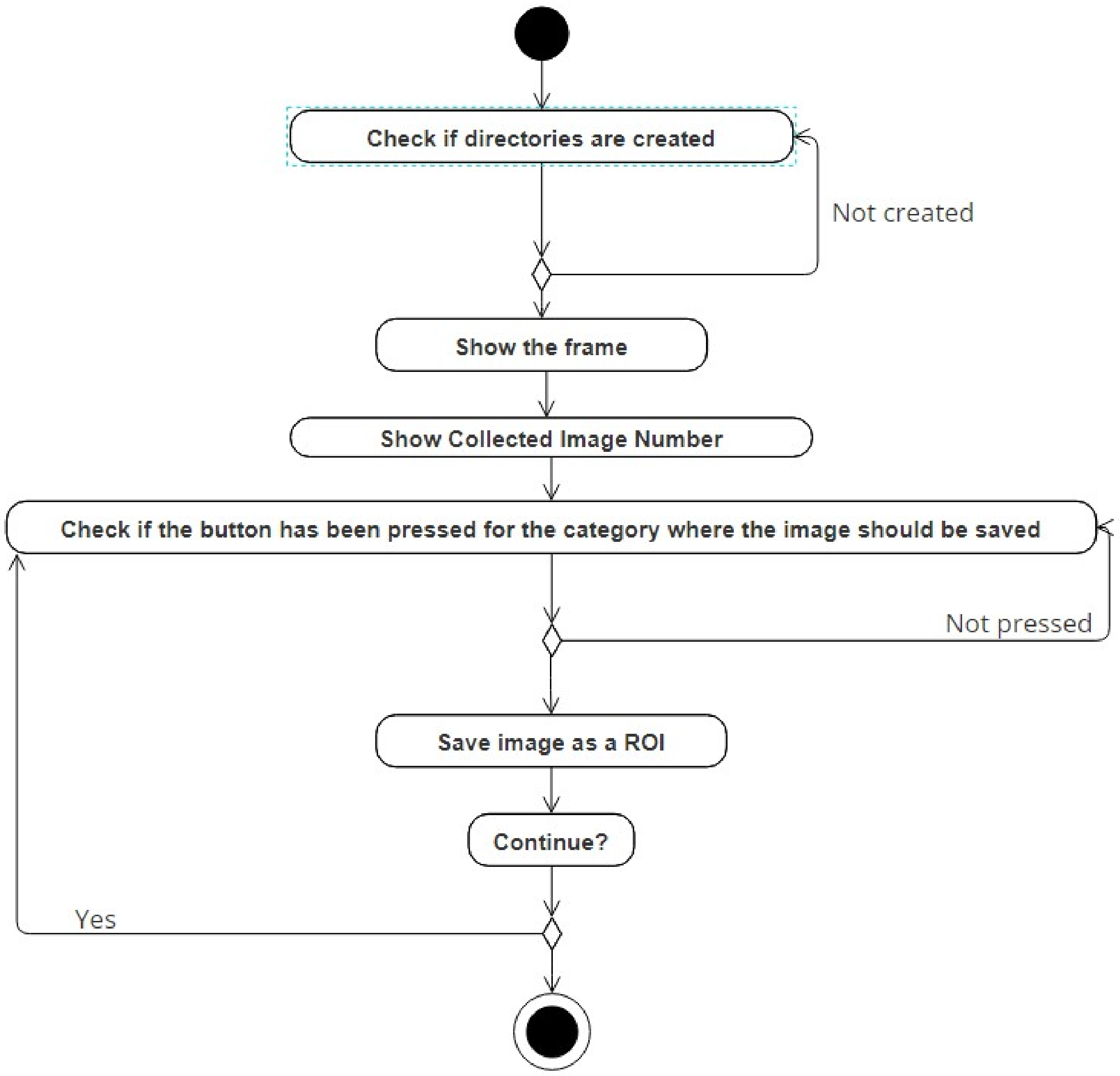

Check if directories are created
Not created
Show the frame
Show Collected Image Number
Check if the button has been pressed for the category where the image should be saved
Not pressed
Save image as a ROI
Continue?
Yes

Figure 3.2: Dataset Application Activity Diagram

3.3. Model

In this project, a model is needed to solve the classification problem. After doing some research, it was decided what these models could be. Convolutional Neural Network was focused on, then previous projects were examined, and it was understood that this was the most suitable model for the project. This model is an extremely good picture recognition method. CNN solves the classification issue with a normal Neural Network, but it also includes extra layers to represent information and identify particular traits.. This helped classify the movements and solve the problem.

3.3.1. CNN Layers

3.3.1.1. Convolution Layer . This is the initial layer that extracts the different characteristics from the input photos. The convolution mathematical process is done between the input picture and a filter of a certain size MxM in this layer. The dot product between the filter and the sections of the input picture with regard to the size of the filter is taken by sliding the filter across the input image.(MxM).

The Feature map is the result, and it contains information about the picture, such as the corners and edges. This feature map is then supplied to further layers, which learn a variety of different input picture characteristics[21].

3.3.1.2. Pooling Layer. A Pooling Layer is usually applied after a Convolutional Layer. The major goal of this layer is to reduce computational expenses by reducing the size of the convolved feature map. This is accomplished by reducing the connections between layers and working on each feature map individually. There are various sorts of Pooling procedures based on the approach utilized[21].

The feature map is used as the most important aspect in Max Pooling. The average of the components in a predetermined sized Image segment is calculated using Average Pooling. Sum Pooling calculates the total sum of the components in the designated section. Between the Convolutional Layer and the FC Layer, the Pooling Layer normally acts as a link.

3.3.1.3. Fully Connected Layer. The weights and biases, as well as the neurons, make up the Fully Connected (FC) layer, which is used to link the neurons between two layers. The last several layers of a CNN Architecture are generally positioned before the output layer.

The preceding layers' input picture is flattened and supplied to the FC layer. After that, the flattened vector proceeds through a few additional FC levels, which is where the mathematical function operations are normally performed. At this point, the categorization process gets underway[21].

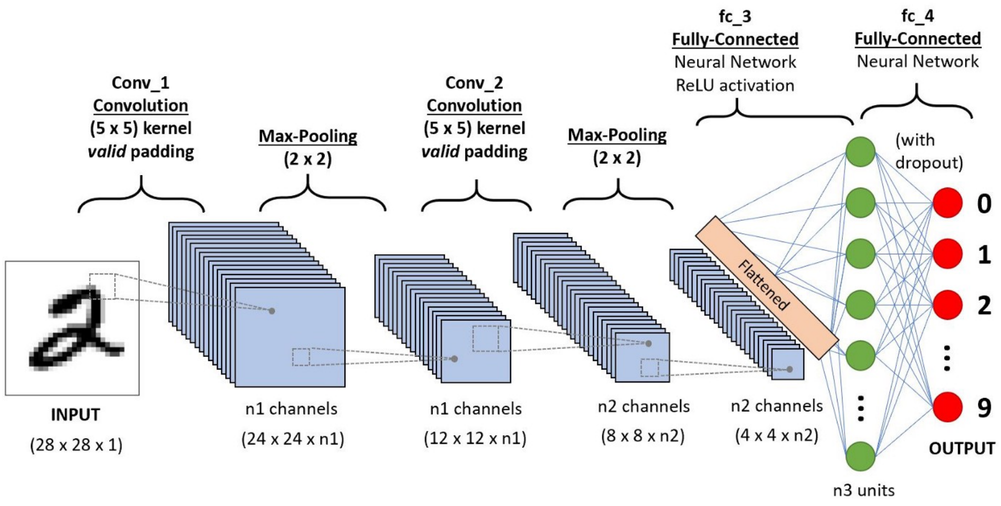
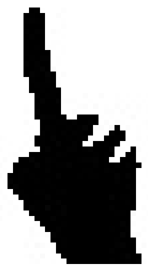


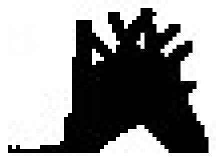
Figure 3.3: CNN Model Example

<u>3.3.1.4. Dropout.</u> When all of the characteristics are linked to the FC layer, the training dataset is prone to overfitting. Overfitting happens when a model performs so well on training data that it has a detrimental influence on its performance when applied to fresh data. A dropout layer is used to solve this problem, in which a few neurons are removed from the neural network during the training process, resulting in a smaller model. After passing a dropout of 0.3, 30% of the nodes in the neural network are thrown out at randomly[21].

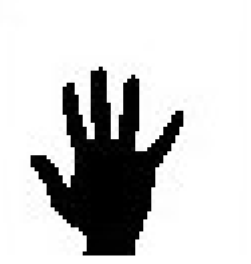
<u>3.3.1.5. Activation Functions.</u> Finally, the activation function is one of the most crucial elements in the CNN model. They're utilized to learn and approximate any form of network variable-to-variable association that's both continuous and complicated. In basic terms, it determines which model information should fire in the forward direction and which should not at the network's end. It gives the network non-linearity. The ReLU, Softmax, tanH, and Sigmoid functions are some of the most often utilized activation functions. Each of these functions has a distinct use. Sigmoid and softmax functions are chosen for binary classification CNN models, while softmax is commonly employed for multi-class classification[21].



(a) First Example



(b) Second Example



(c) Third Example

(d) Fourth Example

Figure 3.4: Examples From Dataset

3.3.1.6. Dense Layer. The dense layer is a deep-connected neural network layer, meaning that each neuron in the dense layer gets input from all neurons in the preceding layer. In the models, the dense layer is revealed to be the most usually used layer. The dense layer conducts matrix-vector multiplication in the background. The values in the matrix are parameters that can be taught and modified using backpropagation techniques. The dense layer produces an'm' dimensional vector as its output. As a result, the dense layer is mostly utilized to alter the dimensions of the vector. Dense layers also perform operations on the vector, such as rotation, scaling, and translation[22].

3.4. Determination of the Model

Five models were tested in this project. The first two models contain a convolutional layer with a 3x3 kernel size, a pooling layer with a pool size with 2x2, a flatten layer and a dense layer. In the first model, the Convolutional Layer's filter was applied as 16. In the second model, this number was increased to 32, but negative results were obtained from both models. In the remaining three models, the number of convolutional layers with 3x3 kernel size has been increased to two and the number of pooling with 2x2 pool size has been increased to two. In the third model, the filter parameters of the Convolutional layers were determined as 16, in the fourth model 32 and in the fifth model 64.

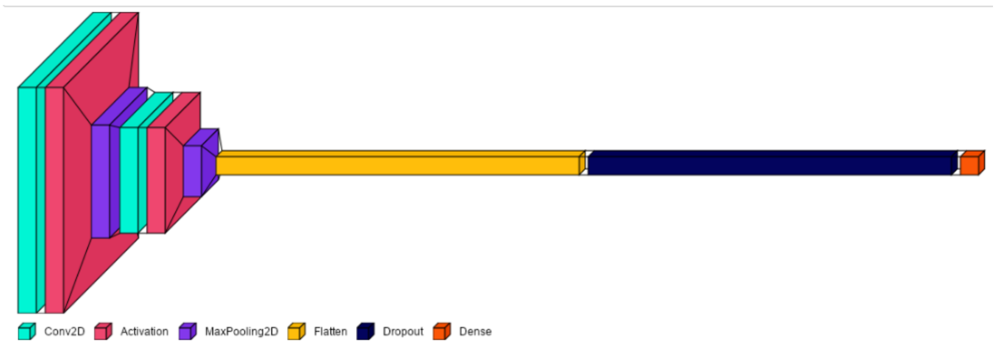


Figure 3.5: Model Visualization for Third, Fourth and Fifth Model

3.5. Determination of Gestures

Determining the gestures used in the project first started with understanding how the model works. The model to be applied determined that black and white data would be used while creating the data set. For this reason, it was decided to use simpler gestures that the model could achieve with a higher accuracy rate instead of the gestures in which the hands were intertwined.

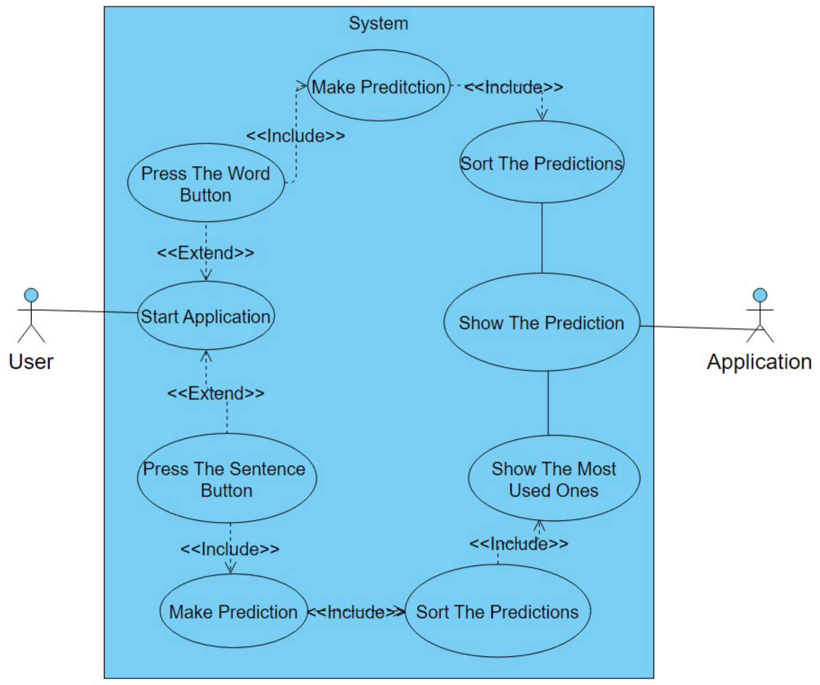


Figure 3.6: Use Case Diagram for Application

3.6. Image Processing

Preprocessing is required to clear image data for model input. Therefore, image preprocessing was performed before entering data into the Convolutional neural network model. Image preprocessing can also reduce model training time and increase model extraction speed.

```
Require Image, Categories, DataPath
$imgSize \Leftarrow 64$
$data \Leftarrow [], target \Leftarrow []$

for Category in Categories do
    $FolderPath \Leftarrow join(DataPath, Category)$
    $imgNames = listDirectory(FolderPath)$
    for Image in imgNames do
        $ImgPath \Leftarrow join(FolderPath, Image)$
        $Img = read(ImgPath)$
        $Gray = ColorSpaceConversions(Img)$
        $Resized = Resize(Gray, (imgSize, imgSize)$
        $Data \Leftarrow append(Resized)$
        $target \Leftarrow append(Category)$
    end for
end for
```

Figure 3.7: Image Processing Algorithm.

4. IMPLEMENTATION

In the implementation chapter, detailed information is given about the language chosen to be used in the project, which libraries are used, how the interface is created, what the dataset contains and how the model is designed.

4.1. Language

It was chosen to use the python language when implementing this project. Python offers us beneficial libraries in machine learning and interface building. It is also a plus that it does not have as complex syntax as other languages.

4.1.1. Libraries

- *cv2* is a good program for image processing and computer vision problems. It's an accessible package for tasks including face identification, objection tracking, landmark detection, and more. Python, Java, and C++ are among the languages supported.
- *os* In Python, the OS module has methods for dealing with the operating system. Python's basic utility modules include OS. This module allows you to use operating system-dependent functions on the go. Many functions to interface with the file system are included in the *os* and *os.path* modules.
- *tensorflow* is a machine learning and artificial intelligence software library that is free and open-source. It may be used for a variety of applications, but it focuses on deep neural network training and inference.
- *keras* Google's keras is a high-level deep learning API for creating neural networks. It is built in Python and is used to make neural network construction simple. It also allows for the calculation of numerous neural networks in the backend.
- *numpy* Python's numpy library is the foundation for scientific computing. It's a Python library that includes a multidimensional array object, derived objects, and a variety of routines for performing fast array operations, such as mathematical, sorting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation, and more.
- *matplotlib.pyplot* is a set of algorithms that allow matplotlib to behave similarly to MATLAB. Each pyplot function modifies a figure in some way, such as creating a

figure, a plotting area in a figure, charting certain lines in a plotting area, decorating the plot with names, and so on. Matplotlib is used.

- *sklearn.modelselection* is a technique for creating a blueprint for analyzing data and then applying it to fresh data. When creating a forecast, choosing the right model assists you to get accurate results. To do so, you'll need to use a specific dataset to train your model. The model is then put to the test against a new dataset.

- *sklearn.metrics* For specialized uses, the sklearn.metrics module includes routines for measuring prediction error.

- *seaborn.heatmap* A depiction of rectangular data as a color-encoded matrix is known as seaborn.heatmap. It accepts a 2D dataset as a parameter. An ndarray may be created from that dataset. This is a fantastic approach to display data since it allows you to see the relationship between variables such as time.

- *pandas* is a data manipulation and analysis software package for the Python programming language. It includes data structures and methods for manipulating numerical tables and time series, in particular.

- *operator* is one of Python's built-in modules, and it includes functions like add(x, y), floordiv(x, y), and others that may be used to perform a variety of mathematical, relational, logical, and bitwise operations on two input integers.

- *visualkeras* is a python package with the facility of visualization of deep learning models architecture build using the Keras API also we can visualize the networks built using Keras included in TensorFlow.

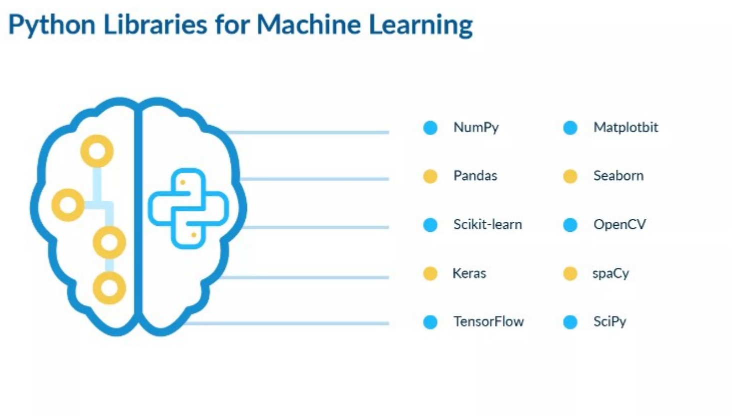
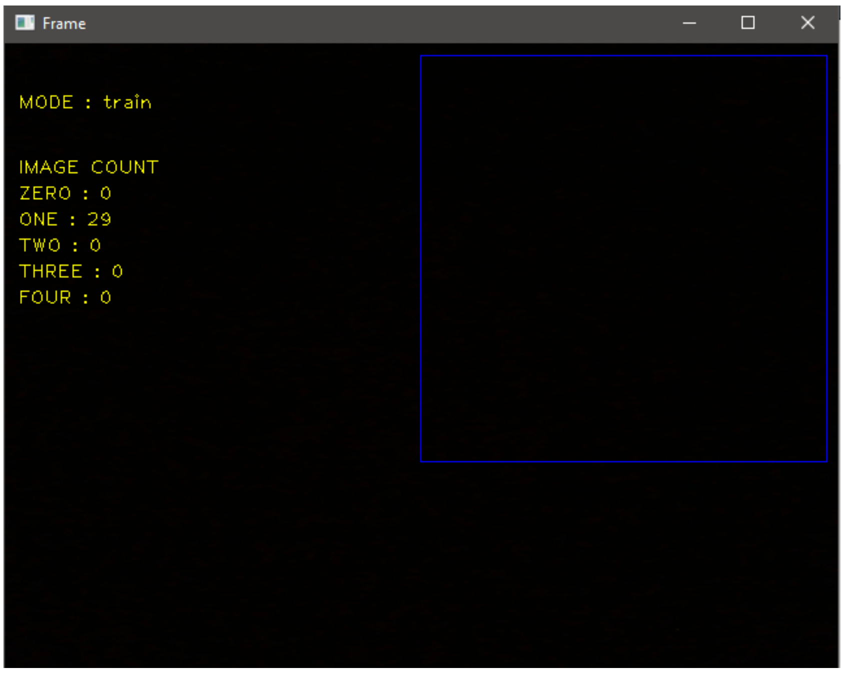


Figure 4.1: Python libraries examples for Machine learning

4.2. Interface to Collect Data

OpenCV was used as a library in the program written to create the data set. OpenCV is an open-source, free library and has many methods for building real-time applications. While creating the interface first, the camera is captured using the OpenCV library. Then the files in the directory and the number of images in the files are printed on the left side of the screen as text. After these, the area that the hand should be taken while making the movement is indicated in the form of a rectangle in the application. When the application is started, the image of the ROI on a small screen is also opened by capturing the camera itself to create a data set efficiently. Thanks to this interface, it is possible to create data sets of up to 5 categories using the keys on the keyboard.



```
MODE : train

IMAGE COUNT
ZERO : 0
ONE : 29
TWO : 0
THREE : 0
FOUR : 0
```

Figure 4.2: Collect Data Interface

4.3. Interface to Predict Gestures

While designing the interface on which the model is loaded, the operator, NumPy, Keras and OpenCV libraries are used to predict the sign. Two buttons have been added to allow users to perceive the signs as words or sentences according to their wishes. When the user wants to translate the displayed movement as a single word, the word predicted by the model is written under the buttons as a word. When the user selects the Sentence button, after correctly guessing the words using OpenCV, the model puts the predicted words on the bottom of the interface in order. If the model cannot detect any movement for a long time after the sentence is finished, the sentence written under the interface is removed from the interface.

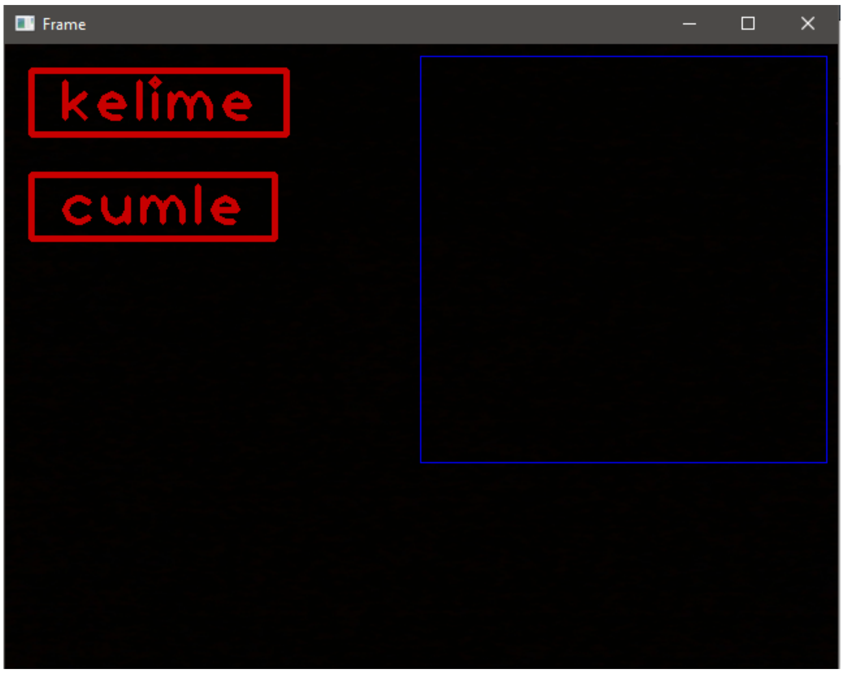


Figure 4.3: Prediction Interface

4.4. Creating a Dataset

Using the program whose interface was created in section 4.2, 29 gestures, 28 of which were words, were added. A category was created from blank data for the word that will be displayed when a gesture is not detected. About 500 data were collected for each movement. In total, 9268 data were collected for training and 2317 for testing.

Table 4.1: Dataset Categories with Size

	Category	Data Size		Category	Data Size
1	Aksam	419	16	Lise	468
2	Alti	275	17	Nasilsin	344
3	Bir	311	18	Nerede	336
4	Yedi	280	19	Hayir	355
5	Dokuz	227	20	On	398
6	Dort	480	21	Sekiz	351
7	Merhaba	419	22	Sen	679
8	Icinde	434	23	Sifir	417
9	Iki	568	24	Sinav	347
10	Seni Seviyorum	491	25	Dur/Bes	434
11	Isim	319	26	Tanismak	306
12	Istanbul	390	27	Turkiye	418
13	Iyi	479	28	Uc	488
14	Kisi	305	29	Hareket Algilanamadi	200
15	Yes	647			

4.5. Designing the CNN-Model

In this section, information is given about what stages the image goes through before the image is entered into the model, which layers, parameters and functions are used while designing the model, and how the data set is separated as a percentage.

4.5.1. Image Pre-Processing

Image preprocessing is both the first and most important step to perform. In this step, all images were reshaped to the required 64x64 size and divided into 255 as a normalization step. Since all categories are in separate files, all data was processed using loops to navigate through these files.

4.5.2. Model Building

After the data set consisting of 28 movements was created, the model was started to be created. A Convolutional Neural Network consists of several convolution and pooling layers. Added two layers of Conv2D and MaxPooling to the model. The first parameter of the Conv2D layer is where we have to play hard to get the best possible model. After adding the convolution layers, Max pooling layers were added where the input layer was added using the Flatten function. There are a variety of alternative activation functions, but RelU is the most often utilized for many types of neural networks as it is easier to train due to its linear behaviour and generally provides better performance. Therefore, the relu function is added to the Convolution layer as activation. Dropout layers are essential in training CNNs because they avoid overloading the training data. In order to prevent this in the training of the model, the Dropout layer has been added after the flatten function. Finally, added the output layer, which gives us the output at the end. The Dense() layer is used in the same way. It took parameter 29 because there are 29 categories. Also, the activation function used here is the softmax function since it is a multi-class problem. Above was only model architecture. Now it needs to be compiled before training. The optimizer used is the common adam optimizer. The categorical cross-entropy loss function was chosen because the labels of the evaluated dataset were categorical and not single-hot encoded. An early stop was used to prevent overfitting. This was implemented in the compilation part, as we did not know how many epochs our model should be trained for.

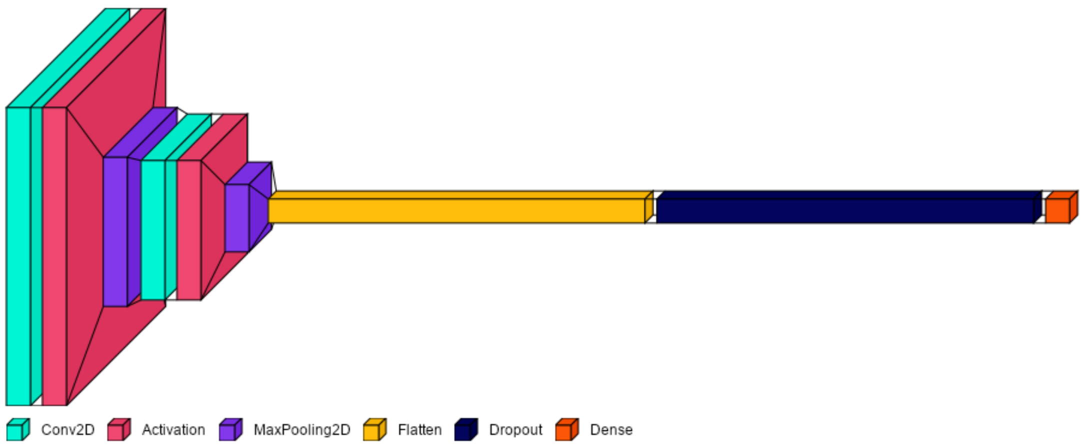


Figure 4.4: Visualized Convolutional Neural Network Model

4.5.3. Train-Test-Val Split

In this step, the dataset is split into a training set, test set, and validation set to use the validation set approach to train the model to classify across categories. A total of 80% of the data was utilized for training and 20% for testing. For validation, 20% of the training set was used.

5. TEST AND RESULTS

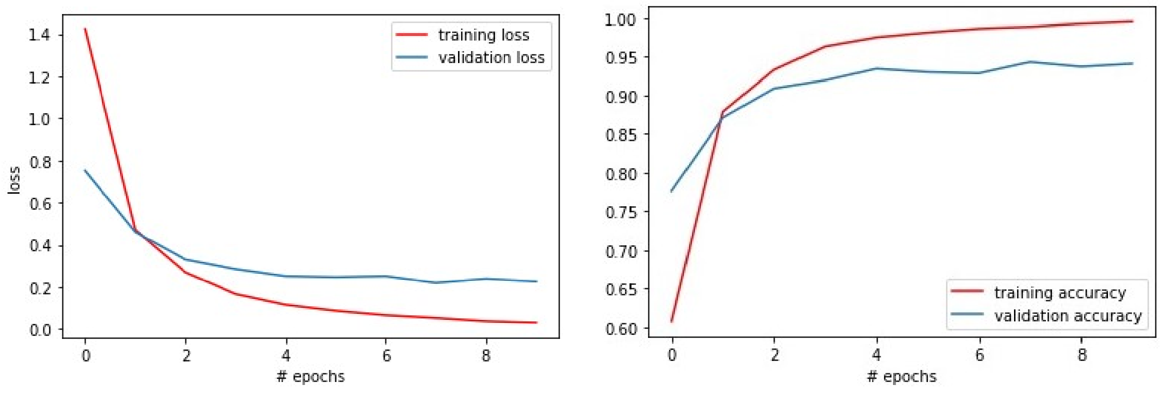
The test and results chapter contains information about the tests performed to create the best model and the performance of the models, the interpretation of the F1 Score tables and the Confusion Matrices of the models, how to choose the best model based on tables, matrices and real-time test performance of the models.

5.1. Designing the Most Accurate Model

This section provides detailed information about the tests performed to design the model with the highest performance, the layers added, the parameters changed and the consequences of changing them. The results of the tests can be observed by using graphs and tables.

5.1.1. First Model

While creating the model was started by adding a Convolutional layer with 16 filters and a 'relu' activation function as the first layer. After that, the pooling layer with two parameters was added. After adding a flatten layer to flatten the data in matrix form, a dropout layer has been added to prevent overfitting. Finally, since there are 29 categories as the output layer, the dense layer activation function, whose parameter is 29, has been added as 'softmax'. However, it was decided that this model was not good enough and did not perform correctly. It gave us 21% as loss rate and 94% as accuracy. The category with the least f1 score among the categories gave the value of 0.30. That is a pretty low score.

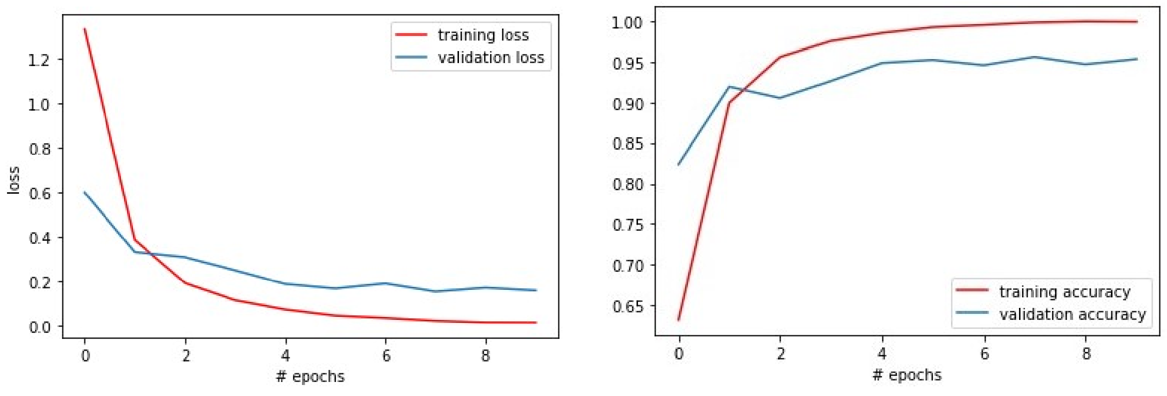


(a) Loss (b) Accuracy

Figure 5.1: First model loss and accuracy

5.1.2. Second Model

In the second model test, changing the filters was tried. It was thought that increasing the filter value in the first parameter in the Convolutional Layer could increase the performance. On the model made in the previous model, only the parameters were changed, and the parameter value from 16 was changed to 32. Changing the parameters increased the accuracy to 95% and reduced the loss rate to 18%. The least f1 score between categories increased up to 0.42. However, that still did not prevent underfitting.

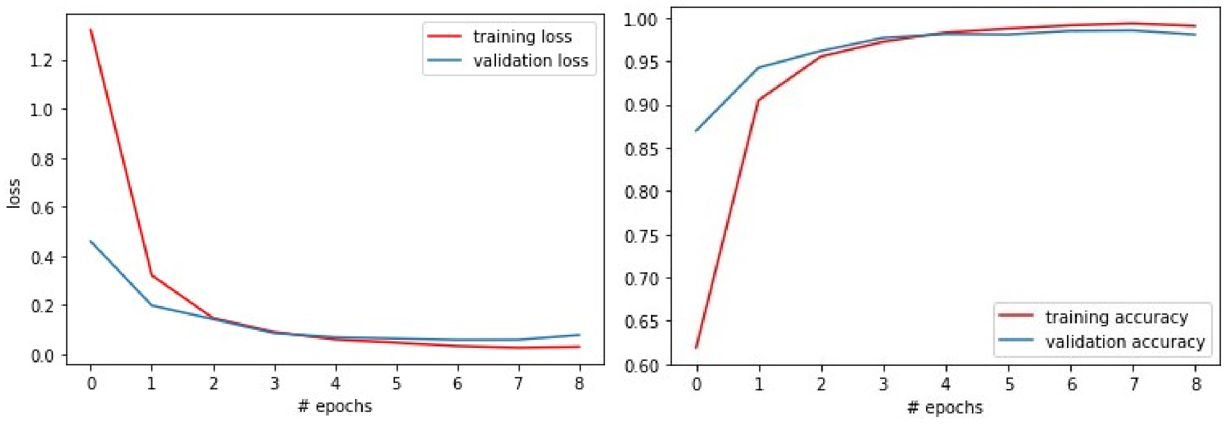


(a) Loss (b) Accuracy

Figure 5.2: Second model loss and accuracy

5.1.3. Third Model

While testing the third model, it was found to be inadequate, and another convolution layer with the 'relu' function was added, and then another Pooling layer with parameter two was added. With these, the filter parameter of both Convolutional Layers was set to 16, and the third model was completed. This model outperformed previous models in terms of a loss ratio of 5% and accuracy of 98%, but still could be improved a bit, as seen from the confusion matrix.(Figure 5.8(c))

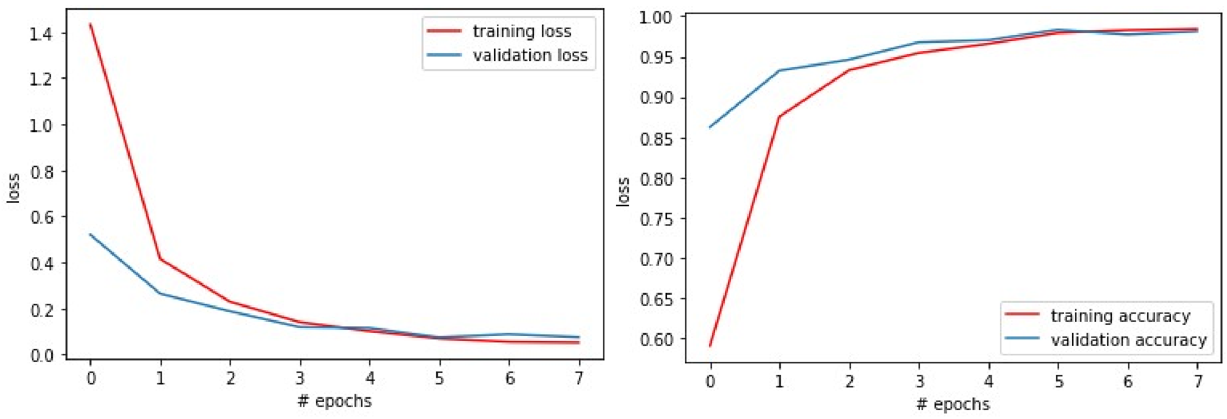


(a) Loss (b) Accuracy

Figure 5.3: Third Model loss and accuracy

5.1.4. Fourth Model

The number of filters in the Convolutional layers, which was only 16 without changing the layers in the third model, was increased to 32. As a result of this test, it was observed that the accuracy rate remained constant at 98%, and the loss increased by 1 point to 6%. Nevertheless, this test turned out to give better results in the confusion matrix.

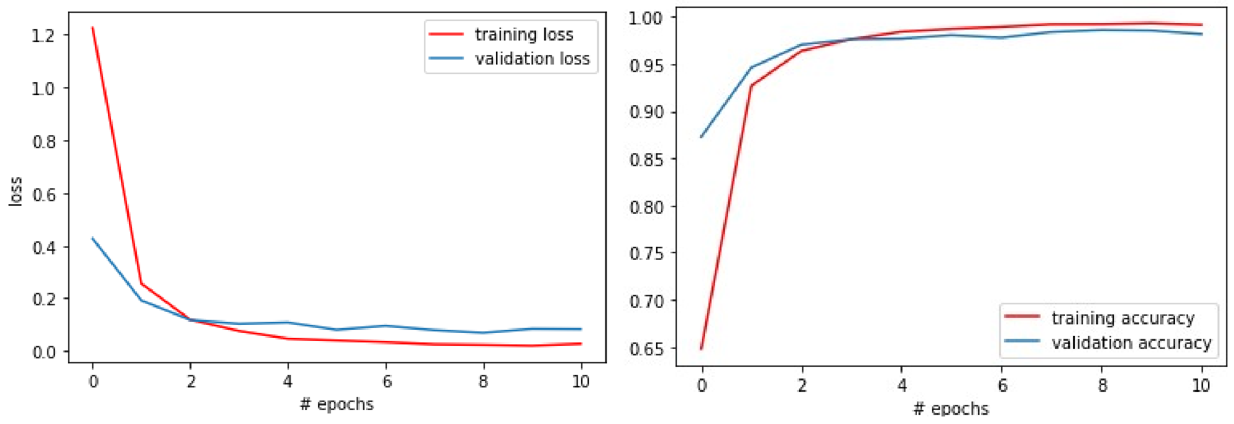


(a) Loss (b) Accuracy

Figure 5.4: Fourth Model loss and accuracy

5.1.5. Fifth Model

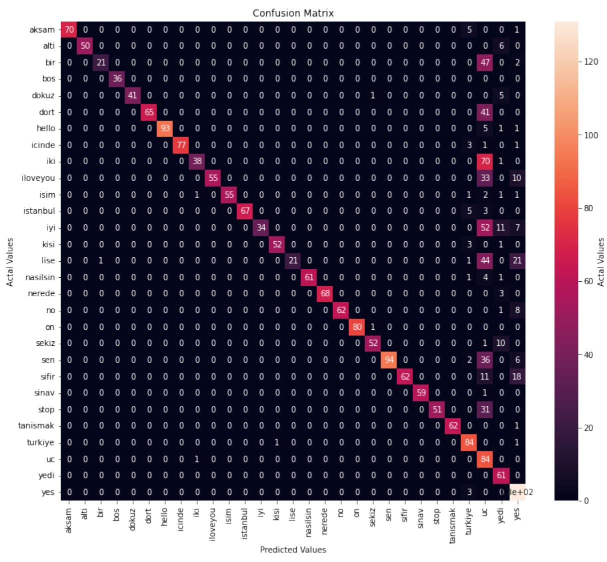
In the fourth model, the performance increase with the increasing of the number of filters was examined. Then, the number of filters in the Convolutional layers was increased to 64 without changing the Third and Fourth model layers. As a result of this test, it was observed that the accuracy rate remained constant at 98%, and the loss was almost the same as the fourth test model but increased by 2 point to 7% from the third model using 16 filters.



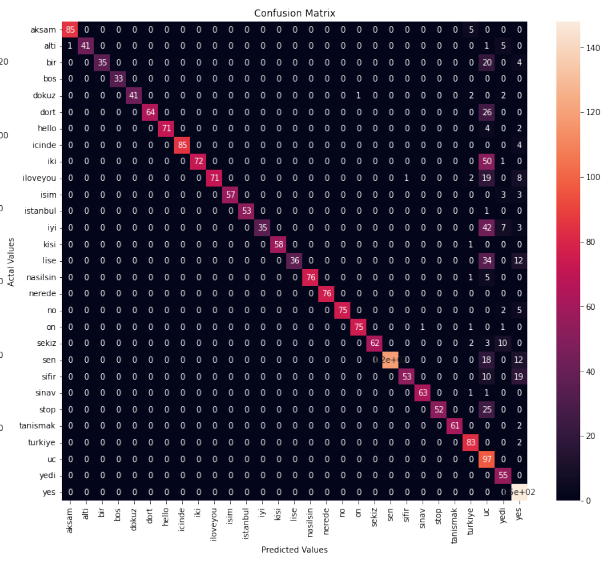
(a) Loss (b) Accuracy

Figure 5.5: Fifth Model loss and accuracy

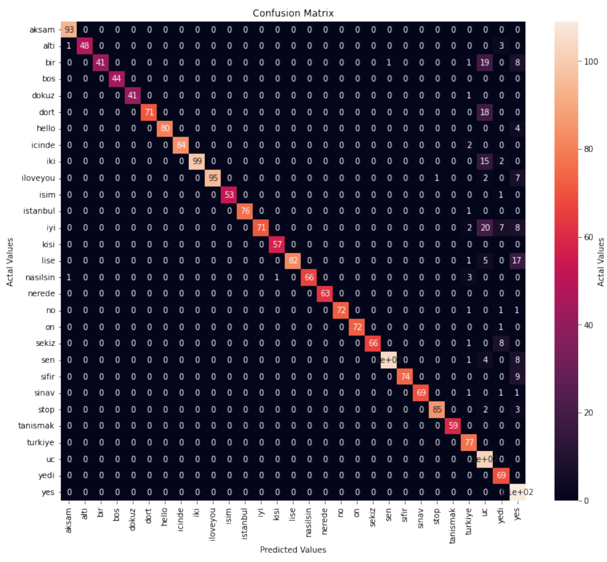
When we examine the confusion matrix, it is observed that the best results are obtained when 64 is entered as the filter parameter. Because when we examined the confusion matrix of other models, it was observed that a word was confused with other categories too much, but this was minimized in our last model. As a result of these tests, it was decided to apply the fifth model.



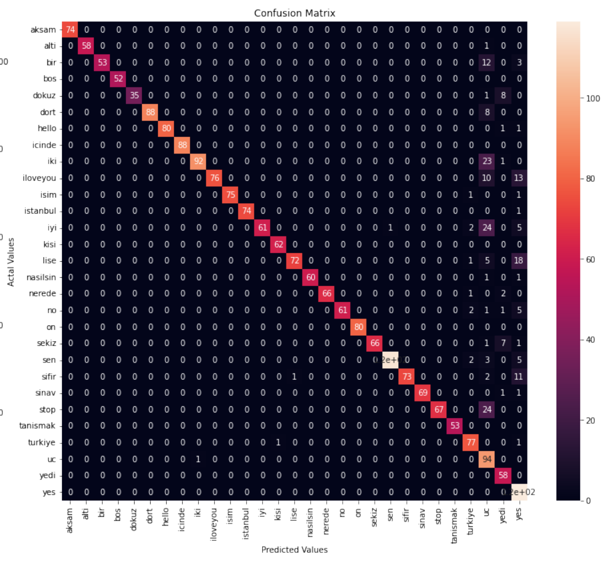
(a) Confusion Matrix for First Model



(b) Confusion Matrix for Second Model



(c) Confusion Matrix for Third Model



(d) Confusion Matrix for Forth Model

Figure 5.6: Confusion Matrix for Tested Models

5.2. Choosing and Testing the Best Model

When the table below (Table 5.1) is examined, it is seen that the loss rates for the first two models are quite high. We see that the remaining models with 64, 32 and 16 filters and two convolutional layers all give very close results. Which model to use was decided after examining the confusion matrixes and the f1 score table.

Table 5.1: Accuracy and Loss Table for Models

Model Number	Loss	Accuracy
1	0.2124	0.9491
2	0.1862	0.9521
3	0.0579	0.9866
4	0.0636	0.9814
5	0.0715	0.9845

When we look at the table containing the F1 Scores (Table 5.2), it is possible to say that the first two models gave very unsuccessful scores. When we look at the other models, we can observe that the fourth model, which takes 32 as a parameter, is quite unsuccessful compared to the fifth and third models. Here we had to choose between the third and fifth models. When the table was examined, it was seen that the f1 score of the numbers that the model predicted incorrectly during the real-time test was higher in the fifth model. Therefore, the fifth model, the model that takes 64 as a parameter, was preferred.

Table 5.2: F1 Scores for Each Model

	Category	First Model	Second Model	Third	Fourth	Fifth
1	Aksam	0.94	0.97	0.99	0.98	0.99
2	Alti	0.96	0.90	0.99	0.99	0.99
3	Bir	0.52	0.76	0.93	0.85	0.91
4	Hareket Algilanamadi	1	1	1	1	1
5	Dokuz	0.84	0.96	0.99	0.92	0.98
6	Dort	0.73	0.85	0.79	0.64	0.98
7	Merhaba	0.92	0.96	0.97	0.94	0.99
8	Icinde	0.97	0.95	0.99	0.93	0.98
9	Iki	0.62	0.69	0.91	0.76	0.94
10	Seni Seviyorum	0.85	0.81	0.93	0.91	0.95
11	Isim	0.94	0.96	0.97	0.98	0.97
12	Istanbul	0.90	0.93	0.96	0.95	0.98
13	Iyi	0.73	0.86	0.93	0.75	0.94
14	Kisi	0.99	0.99	0.99	0.96	0.99
15	Lise	0.54	0.71	0.94	0.84	0.91
16	Nasilsin	0.96	0.94	0.96	0.98	0.99
17	Nerede	0.98	0.99	0.99	0.99	0.99
18	Hayir	0.94	0.89	0.97	0.87	0.98
19	On	0.99	0.99	0.96	0.98	0.97
20	Sekiz	0.83	0.93	0.95	0.94	0.89
21	Sen	0.72	0.75	0.83	0.86	0.97
22	Sifir	0.60	0.71	0.96	0.71	0.92
23	Sinav	0.96	0.98	0.99	0.93	0.98
24	Dur/Bes	0.82	0.85	0.99	0.80	0.94
25	Tanismak	0.95	0.99	0.98	0.99	0.96
26	Turkiye	0.80	0.88	0.87	0.90	0.90
27	Uc	0.30	0.42	0.72	0.53	0.81
28	Yedi	0.69	0.82	0.89	0.79	0.90
29	Yes	0.58	0.62	0.83	0.67	0.84

The Confusion Matrix Shapes we saw above (Figure 5.6) are the confusion matrix of the first four tests. The confusion matrix shown below (Figure 5.7) is the confusion matrix of the model with the highest performance. It belongs to our last model, where we used 64 as the filter parameter.

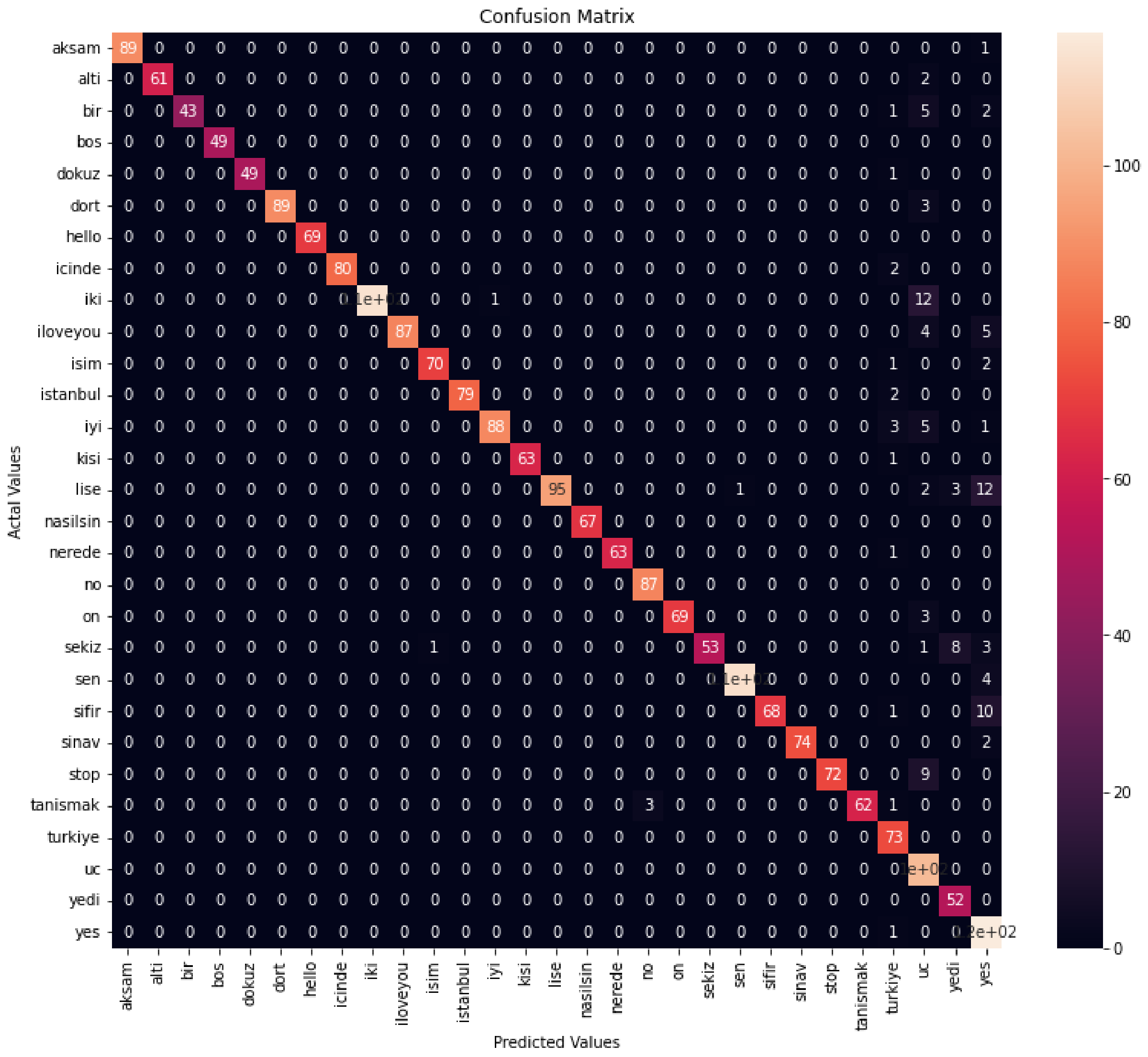


Figure 5.7: Confusion Matrix for Implemented Model

In the real-time testing phase, the model with two convolution layers and 64 as a parameter was observed to work well, except that it predicted incorrect number categories when predicting some number gestures.

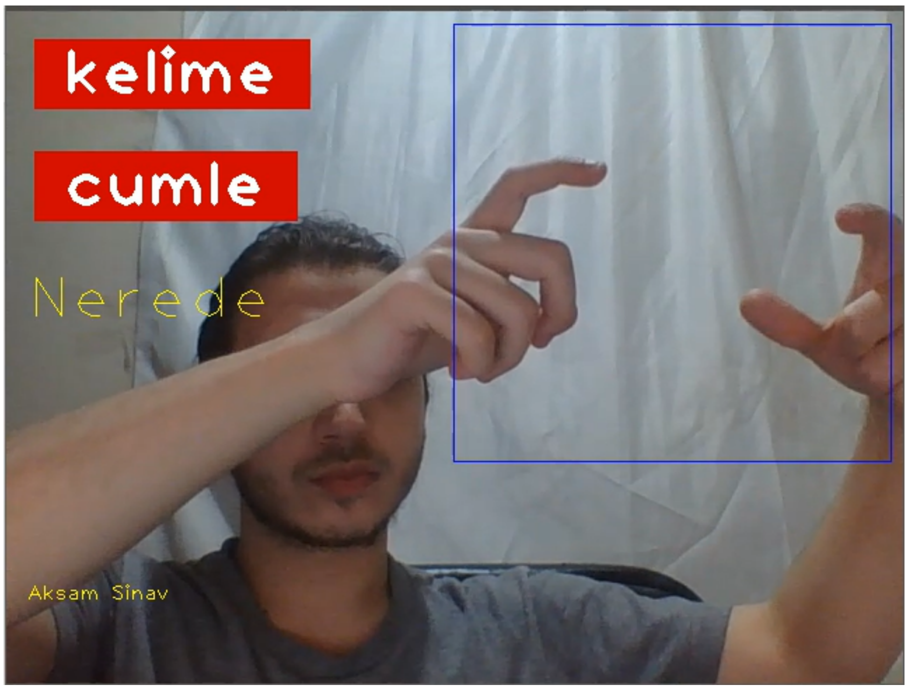


Figure 5.8: Real Time Testing the Model

6. CONCLUSION

The machine learning model applied in the project has been subjected to real-time tests, and it has been observed that it gives very accurate results. It is never possible to completely solve the communication problem of a deaf or hard of hearing person with a person who does not know sign language. However, due to this project, part of the communication problem has been solved.

6.1. Future Work

The most important future need for improving the machine learning model is to add gestures data that has been made by more people and add more words. If these needs are met, it will be easier to solve the communication problem between people. If the model has difficulty making predictions as a result of adding this data, it may be necessary to add a few more layers to the model.

Bibliography

[1] S. Smith. “Not one for all: Understanding different sign languages.” (2021), [Online]. Available: `https://dayinterpreting.com/blog/not-one-for-all-understanding-different-sign-languages/`.

[2] Wikipedia. “Sign language.” (2021), [Online]. Available: `https://en.wikibooks.org/wiki/Sign_language`.

[3] J. Shang and J. Wu, “A robust sign language recognition system with multiple wi-fi devices,” in *Proceedings of the Workshop on Mobility in the Evolving Internet Architecture*, 2017, pp. 19–24.

[4] H. Renuka and B. Goutam, “Hand gesture recognition system to control soft front panels,” *International Journal of Engineering Research & Technology (IJERT)*, vol. 3, no. 12, pp. 5–10, 2014.

[5] S. R.Karthik kumar I.J.Augustin jacob, “Hand gesture recognition system for better human computer interaction,” 2013.

[6] A. G. S. B. Yrd. Doç. Dr. Zafer YILDIZ Dr. Savaş YILDIZ, “İşitme engelli turizmi,” *Suleyman Demirel University Visionary Journal*, vol. 9, no. 20, pp. 103–117, 2018.

[7] M. Raza. “Machine learning introduction.” (2021), [Online]. Available: `https://medium.com/geekculture/machine-learning-introduction-c7c0b40ad6d1`.

[8] W. Chai. “A timeline of machine learning history.” (2020), [Online]. Available: `https://www.techtarget.com/whatis/A-Timeline-of-Machine-Learning-History`.

[9] L. GUERROUJ. “Machine learning: 6 real-world examples.” (2021), [Online]. Available: `https://www.salesforce.com/eu/blog/2020/06/real-world-examples-of-machine-learning.html`.

[10] M. Kaur. “Top 10 real-life examples of machine learning.” (2019), [Online]. Available: `https://bigdata-madesimple.com/top-10-real-life-examples-of-machine-learning/`.

[11] D. K. Judith Hurwitz, *Machine Learning For Dummies®, IBM Limited Edition*. IBM, 2018.

[12] A. Burkov, *The Hundred-Page Machine Learning Book*. Andriy Burkov, 2020.

[13] W. K. Clifford. “Connectionist ai.” (2018), [Online]. Available: `http://www.mysearch.org.uk/website1/html/106.Connectionist.html`.

[14] M. Terry-Jack. “Deep learning: Background research.” (2019), [Online]. Available: https://medium.com/@b.terryjack/deep-learning-background-research-64578f0d551d.

[15] S. University. “History: The 1940’s to the 1970’s.” (), [Online]. Available: https://cs.stanford.edu/people/eroberts/courses/soco/projects/neural-networks/History/history1.html.

[16] J.-C. B. Loiseau. “Rosenblatt’s perceptron, the first modern neural network.” (2019), [Online]. Available: https://towardsdatascience.com/rosenblatts-perceptron-the-very-first-neural-network-37a3ec09038a.

[17] K. Some. “The history, evolution and growth of deep learning.” (2018), [Online]. Available: https://www.analyticsinsight.net/the-history-evolution-and-growth-of-deep-learning/.

[18] Wikipedia. “Connectionism.” (2014), [Online]. Available: https://en.wikipedia.org/wiki/Connectionism.

[19] S. Akbani. “Brain tumor detection using deep learning.” (2022), [Online]. Available: https://www.ijraset.com/research-paper/brain-tumor-detection-by-deep-learning.

[20] Mathworks. “What makes cnns so useful?” (2022), [Online]. Available: https://www.mathworks.com/discovery/convolutional-neural-network-matlab.html.

[21] M. Gurucharan. “Basic cnn architecture: Explaining 5 layers of convolutional neural network.” (2020), [Online]. Available: https://www.upgrad.com/blog/basic-cnn-architecture/#:~:text=Convolutional%20Layer,of%20a%20particular%20size%20MxM..

[22] Y. Verma. “A complete understanding of dense layers in neural networks.” (2021), [Online]. Available: https://analyticsindiamag.com/a-complete-understanding-of-dense-layers-in-neural-networks.