VIET NAM NATIONAL UNIVERSITY HO CHI MINH CITY
HO CHI MINH CITY UNIVERSITY OF TECHNOLOGY
FACULTY OF COMPUTER SCIENCE AND ENGINEERING

GRADUATION THESIS

USING MACHINE LEARNING METHODS IN TRANSLATING SIGN LANGUAGE INTO VIETNAMESE

COUNCIL: SOFTWARE ENGINEERING

SUPERVISOR: Assoc. Prof. Quản Thành Thơ
REVIEWER: Trần Hồng Tài, M.Sc
—o0o—
STUDENT 1: Võ Tuấn Khanh (1810220)
STUDENT 2: Nguyễn Trí Nhân (1810390)

HO CHI MINH CITY, 05/2022

Declaration Of Authenticity

We claim that this research is our work, conducted under the supervision and guidance of Associate Professor Quản Thành Thơ. The result of our research is authentic and has never been published in any platforms before. All the materials we utilized in this research were gathered from various sources and are referenced correctly in the References Section.

Furthermore, within this research, we also used the results of several other authors and organizations. All of them have been correctly cited. In any case of plagiarism, we stand by my actions and will be responsible for it. Therefore, Ho Chi Minh City University of Technology is not responsible for any copyright infringements conducted within our research.

Acknowledgment

To complete the thesis outline, I would like to express my deep gratitude to Associate Professor Quản Thành Thơ for his guidance throughout the research process.

We want to express our sincere thanks to the Faculty of Computer Science and Engineering teachers, Ho Chi Minh City University of Technology, for their dedication to imparting knowledge during our years of study at the school. The knowledge accumulated during the study process is the foundation for the research process and the future professional goals.

Finally, we wish you good health and success in your noble career.

Abstract

Presently, more than two million people with deafness and difficulty in hearing in Vietnam cannot talk freely with those who do not have those symptoms. There are a few ways for the deafness and difficulty in hearing to communicate with others, but they are ineffective and inefficient. The most used method is through notes or body language to express their thoughts to others. However, not many people comprehened sign language, making it hard to understand the deaf.

To help those people communicate more efficiently, we have built a system using Artificial Intelligence to translate sign language into Vietnamese. The design, in short, contains two physical modules, which are a camera and a smartphone installed our application. Beneath the application is an artificial-intelligence-based pipeline with six more submodules that translate sign language into text and read the word out loud through the phone's built-in speaker.

Besides, the sign language data is collected from various resources through the site `https://tudienngonngukyhieu.com/` and `https://qipedc.moet.gov.vn/`. Based on the videos instructing vocabulary, we gathered and categorized them for the primary dataset of this system.

Overall, our system currently can translate some words. However, the more time it takes to learn the new terms, the more accurate the system is. The system can translate many more words with a much more extensive sign language library. We suppose our approach not only helps people with disabilities in communicating and enriching their lives, but also increases the volume of the workforce and reduces the economic burden on the national budget since then.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Problem statement

"Each deaf person is like a separate world, and they feel more self-deprecated and alone when they can not interact and share with others. They still have the desire to contribute to society", said Mr. Do Hoang Thai Anh, Vice Chairman of the Hanoi Deaf Association [12].

Language is a universal key that not only connects people but also builds up our society. Any disability that affects the ability to communicate is a significant disadvantage, especially for people with disabilities. They cannot integrate, have fun, learn, and communicate like ordinary people because they cannot express their thoughts, ideas, and desires to develop society as we do. That burden usually makes them fall into poverty, live a isolated life, and be exploited, apart from society. Hence, it is challenging for them to have beautiful lives.

In 2020, Vietnam had more than 2.5 million people who are deaf and mute, yet, only a tiny portion of them took part in education, had the chance to be understood, and integrated with society [5].

According to UNICEF, households with members with disabilities are often poorer, children with disabilities are at risk of having less education than their peers, and employment opportunities for people with disabilities are also lower than those without disabilities. Even though people with disabilities are beneficiaries of the policy, and poverty is not a burden to accessing health facilities, very few people with disabilities (2.3%) have access to functional rehabilitation services when being sick or injured. Besides, inequality still exists in living standards and social participation for people with disabilities. Many organizations have been founded to support, help, and create better living conditions for people with disabilities. However, this work still has many difficulties and inadequacies as there is no formal school or class. In addition, there are insufficient sign language translators while they take an essential role in helping the people with

disabilities connect with society.

A quote from Cavett Robert, "Life is a grindstone, and whether it grinds you down or polishes you up is for you and you alone to decide." However, it is challenging for these people to go to school and get a good education. They have their desires and dreams, but our resources and efforts are not enough to make them a polished grindstone. Furthermore, sign language shares the same property as any other spoken language; each different region and territory has a different way of expressing sign language. These unseen differences make communication, self-expression, and information exchange even more complex and challenging for humanity.

In short, we must admit that understanding and breaking the language barrier is extremely necessary and urgent because the deaf and mute, like many other ordinary people, deserve to be assisted, understood, and acknowledged. Furthermore, we believe our system is the resolution to problems of the deaf and hard of hearing.

1.2 Goals of the thesis

the thesis aims to research, understand, and implement solutions to converting sign language into Vietnamese. In particular, the system must receive a queue of images from the camera mounted on the hat, use the implemented algorithms to process and display text on the phone screen.

We can solve the above problem by breaking it into smaller ones listed below. With each issue, we will give our solution and architect a system that can solve the whole problem of the thesis.

- Search and collect data on sign language, conduct evaluation, classification, and normalization of data
- Find out the approaches that have been implemented
- Design architecture of the model
- Plan to implement, develop a sign language conversion system
- Build an application that users can utilize

1.3 Scopes of the thesis

In this case study, we will build a system including an app and camera module to translate sign language into Vietnamese. Because of the limited time, the scope of the study is also limited as follows:

- The system can only translate Vietnamese words
- The system can only recognize the words trained with before

1.4 Structure of the thesis proposal

This proposal includes four sections and each will convey the related works and output when doing the thesis.

Chapter	Content
1	A brief introduction about plan and objectives of thesis
2	Related works that had been done and how they has helped us in doing the thesis
3	Introduction of theoretical background as foundation knowledge that are applied in the thesis
4	Solution and design approach for problem statement of the thesis
5	Result and evaluation for the thesis
6	Summary of the thesis

Table 1: Structure of the thesis

Chapter 2

Related Work

Nowadays, many scholars worldwide have submitted research projects relating to turning sign language into text, using a variety of methodologies and perspectives.

Two main approaches have been proposed:

- **Glove-based approaches:** This approach requires the deaf and mute to wear a sensor glove. When users have any different actions or gestures, the gloves will record all those movements. After that, data from the sensor will be analyzed by the analyzer component. Finally, that component returns the output to the users.
- **Vision-based approaches:** With this approach, developers will apply image processing algorithms to determine hand positions, gestures, and movements. The user will not have to wear necessary glove-based methods, which is convenient. However, using image processing algorithms, we need to deal with the worst quality output affected by these algorithms.

Specifically, about the vision-based approach, the earlier method used several image processing algorithms to build feature vectors based on a single RGB image of the hand. In this paper, "Real-time sign language recognition using a consumer depth camera" [6], using multi-layered random forest (MLRF) not only allows them to recognize hand signs correctly but also minimizes training time and effort. Alternatively, in this paper, "Sign Language Translation in Urdu/Hindi Through Microsoft Kinect," sign language can be recognized by auxiliary equipment: Microsoft Kinect (see Figure 1), which captures the signs of the deaf person, after that, through the computer system, they can detect what deaf people say.

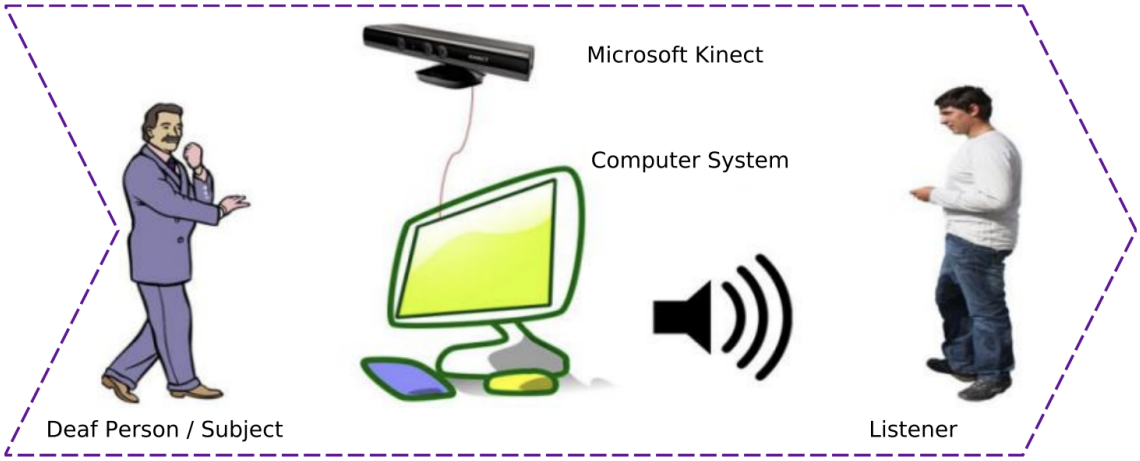


Figure 1: Using Microsoft Kinect to translate sign language

Besides the vision-based approach, glove-based system has a much relevant research. This paper, "The Language of Glove: Wireless gesture decoder with low-power and stretchable hybrid electronics" [9] introduces the way to convert American Sign Language (ASL) alphabet into text and display it on a computer or smartphone (see Figure 2). They can detect which hand gesture is performed with sensor gloves and send the result to the smartphone via Bluetooth. The sign language interprets text and displays it on the digital display screen. This approach is helpful in the real world for the deaf and mute who cannot communicate with ordinary people.

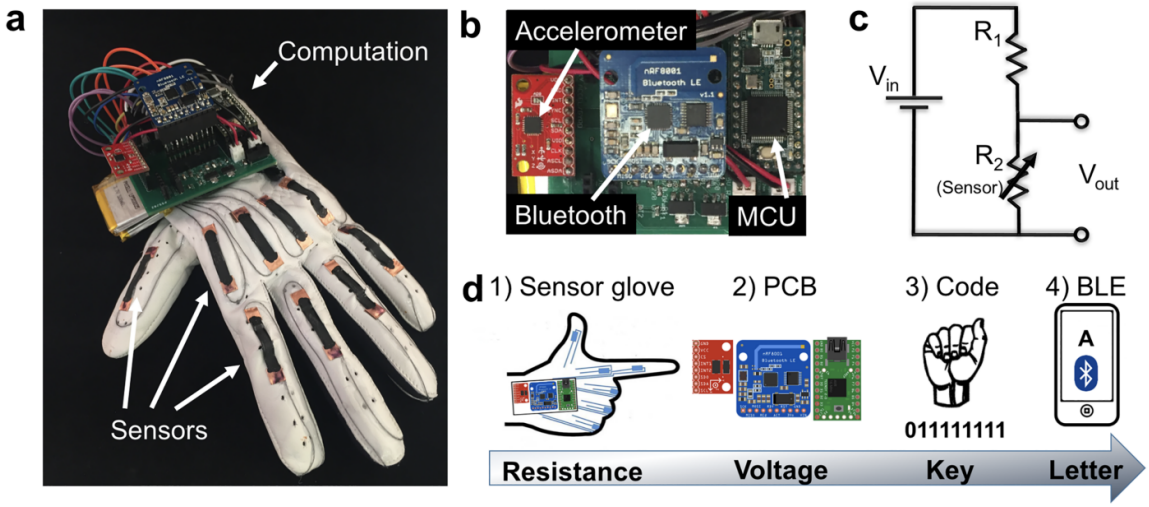


Figure 2: Glove-based approach to translate sign language

Both approaches above have some problems; they can only recognize a minimal number of words, like an alphabet, number, or some word with easy hand shape and no motion. However, it is not easy; many words will use the same hand shape but differ in many characteristics, such as positions and directions. There is currently no model that can handle the conversion of sign

language flexibly and conveniently for the deaf and mute, helping them communicate effectively and naturally. Therefore, thanks to applying appropriate technologies, the authors carry out this graduation thesis to break down the barriers between deaf-and-mute people and ordinary people, helping them become self-sufficient and more confident in daily communication.

Chapter 3

Theoretical Background

3.1 Convolution Neural Network - CNN

Convolution Neural Networks are a particular class of Neural Networks [7]. They are made up of neurons that have learnable weights and biases. Each neuron receives some inputs, performs a dot product, and optionally follows it with a non-linearity. CNN mainly consists of Convolution Layers, Pooling Layers, Activation Layers, and Fully Connected Layers. ConvNet architectures make the explicit assumption that the inputs are images, which allows us to encode specific properties into the architecture. These then make the forward function more efficient to implement and vastly reduce the number of parameters in the network. Some of the primary uses of CNN can be mentioned as image classification, object detection, semantic segmentation, face recognition, ...

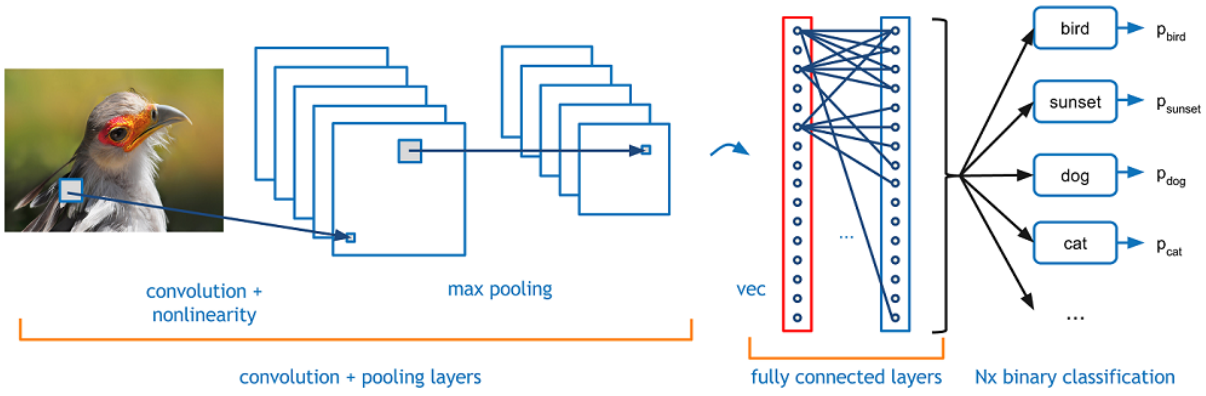


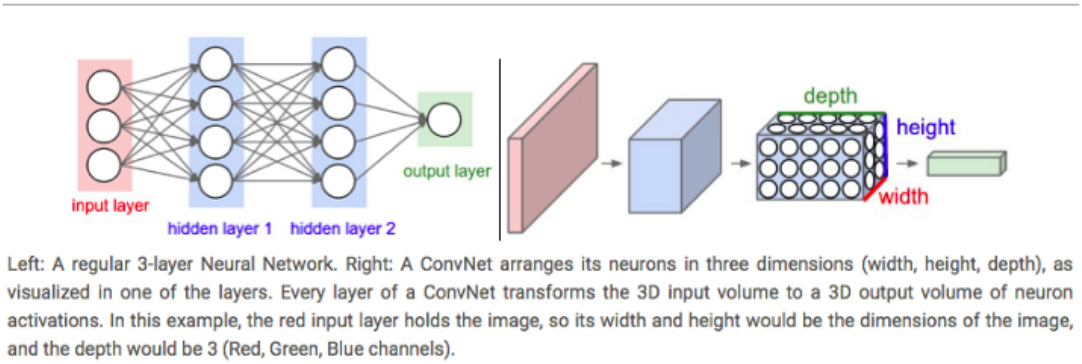
Figure 3: Convolution Neural Network

The Figure 3 above shows an example of a convolution neural network, which is taking an image as input and then extracting features from it through various layers and then finally

predicting the class of the object in the given image.

3.1.1 Architecture

Convolution Neural Networks have a different architecture with regular Neural Networks, and we can see this difference in Figure 4 below. Regular Neural Networks transform an input through a series of hidden layers. Every layer comprises a set of neurons, where each layer is fully connected to all neurons in the previous layers. Finally, a last fully-connected output layer represents the predictions with CNN architecture. First of all, the layers are organized into three dimensions: width, height, and depth. Further, the neurons in one layer do not connect to all neurons in the next layer but only to a small region. Lastly, the system will reduce the final output to a single vector of probability scores, organized along the depth dimension.



Left: A regular 3-layer Neural Network. Right: A ConvNet arranges its neurons in three dimensions (width, height, depth), as visualized in one of the layers. Every layer of a ConvNet transforms the 3D input volume to a 3D output volume of neuron activations. In this example, the red input layer holds the image, so its width and height would be the dimensions of the image, and the depth would be 3 (Red, Green, Blue channels).

Figure 4: Different between Normal Neural Network and Convolution Neural Network

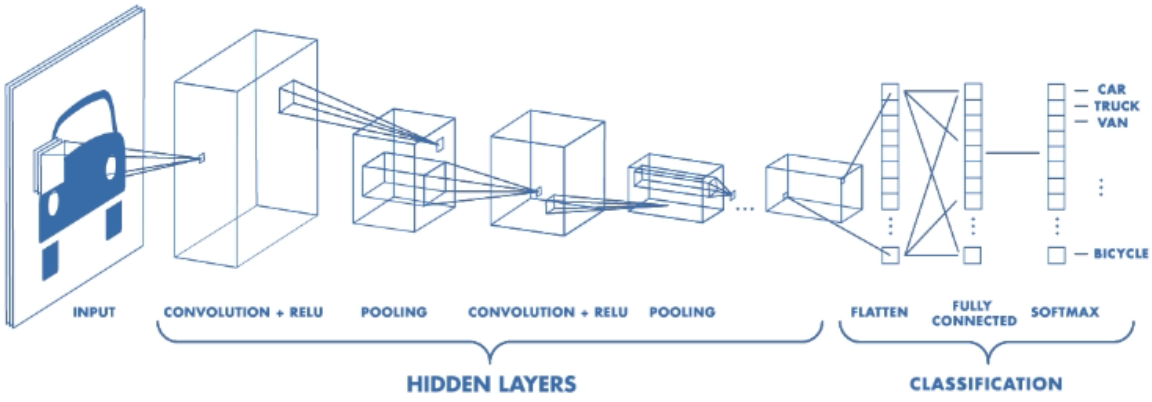


Figure 5: Convolution Neural Network Architecture

As we can see in Figure 5, CNN can be divided into two parts:

1. The hidden layers/ Feature extraction part
   In this part, the network will perform a series of convolutions and pooling operations while the features are detected. Imagine you have a picture of a zebra, these are the parts where the network would recognize: its stripes, two ears, and four legs.
2. The Classification part
   The fully connected layers serve as a classifier on top of their extracted features. By using a provided algorithm, they will assign a probability for the objects on the image.

3.1.2 Feature extraction part

Convolutional Layer

The convolution layer is the core building block of a Convolutional Network that does most of the computational heavy lifting. A convolution is executed by sliding the filter over the input. At every location, matrix multiplication is performed and it sums the result onto the feature map. This extracting features from images happen throughout the CNN's convolutional layers. This process is illustrated in Figure 6

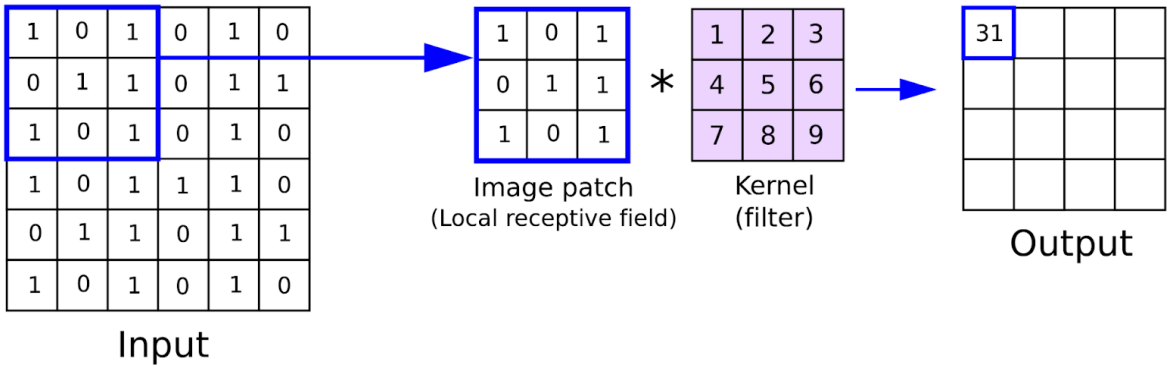


Figure 6: Convolution Neural Network Layer

When the feature map is made, we can pass each value in the feature map through a non-linearity function, such as ReLU, sigmoid before it becomes the input of the next convolution layer.

Because the size of the feature map is always smaller than the input, we have to do something to prevent our feature map from shrinking. This is where we use padding (7). A layer of zero-value pixels is added to surround the input with zeros so that our feature map will not shrink. In addition to keeping the spatial size constant after performing convolution, padding also improves performance, ensures the Kernel and strides size will fit in the input.

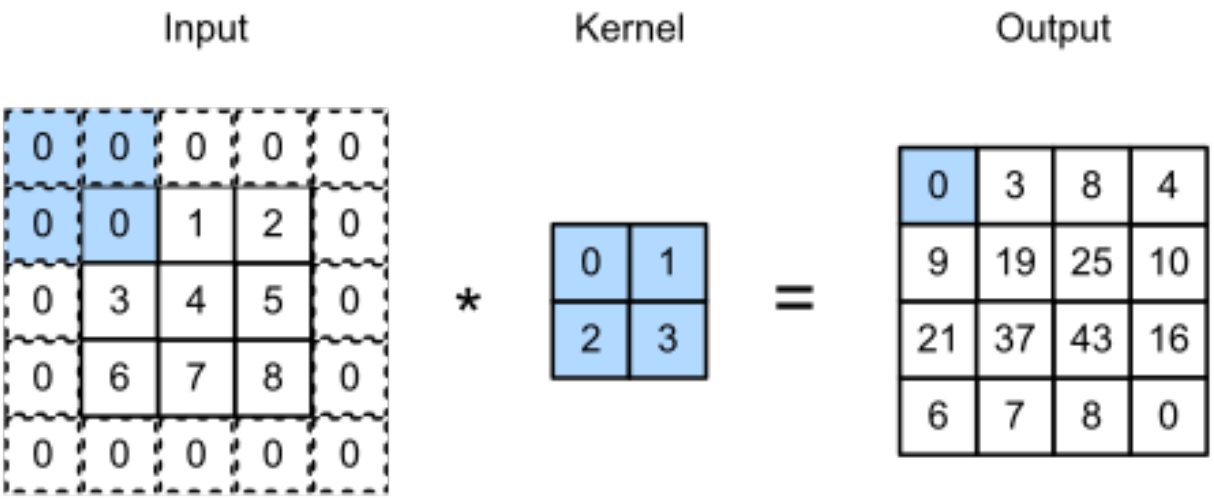


Figure 7: Using padding for strike one in Convolution Layer

Pooling Layers

After a convolution layer, it is common to add a pooling layer in between CNN layers. The function of Pooling is to continuously reduce the dimensionality to reduce the number of parameters and computation in the network. This action shortens the training time and controls overfitting.

There are two main types of Pooling Layers in a CNN: Max Pooling and Average Pooling. The functionality of these two types of layers is demonstrated in Figure 8. Max Pooling restores the maximum value from the picture segment covered by the Kernel. Average Pooling converts the average values from the bit of the picture surrounded by the Kernel.

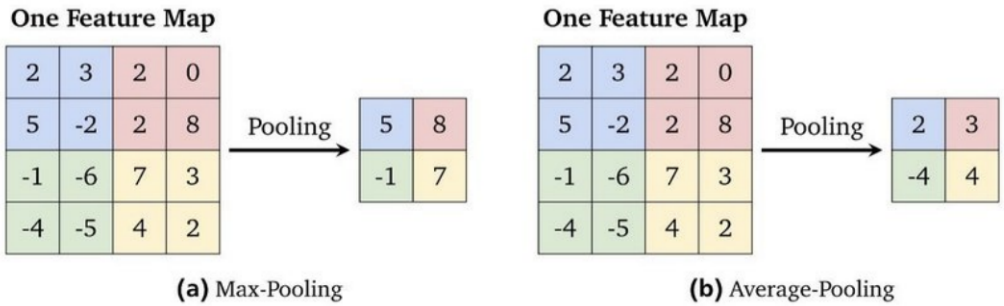


Figure 8: Max Pooling and Average Pooling

Activation Layers

In general, Neural networks and CNNs rely on a non-linear "trigger" function to signal distinct identification of likely features on each hidden layer. CNN may use a variety of specific functions (Figure 9), such as rectified linear units (ReLUs) and continuous trigger (non-linear) functions—to efficiently implement this non-linear triggering.

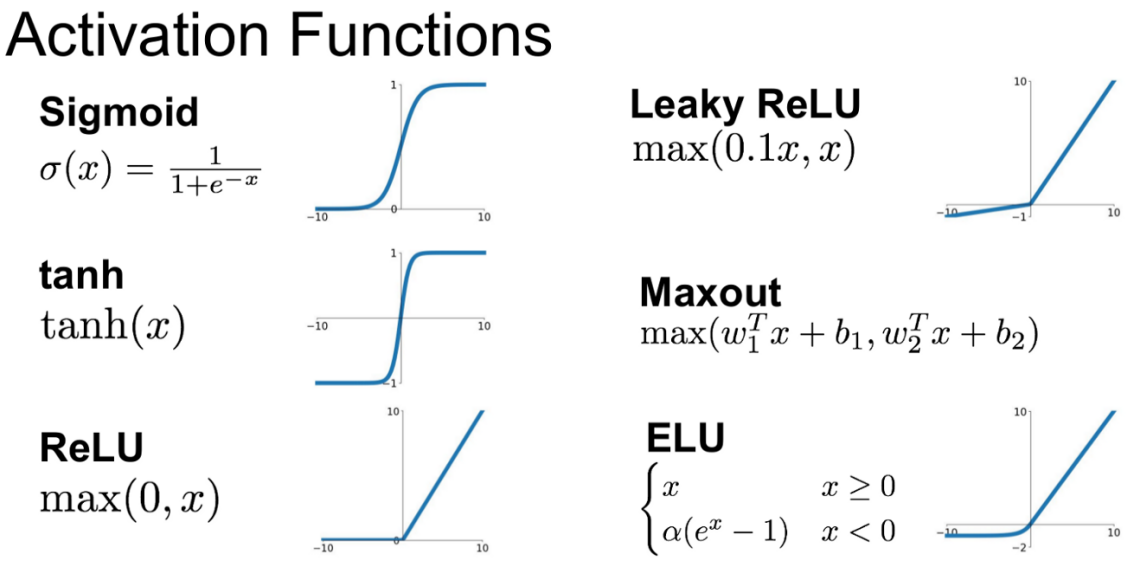


Figure 9: Some Active Function common used in CNN

3.1.3 Classification part

Fully connected layers

The last layers of a CNN are fully connected. Neurons in a fully connected layer have complete connections to all the activations in the previous layer. This part is, in principle, the same as a regular Neural Network.

Figure 10 illustrates the way of input value stream into the fully connected layer. Because these fully connected layers can only accept one-dimensional data, we need to convert our 3D data to 1D data. After passing through some FC, we will get the data classification result.

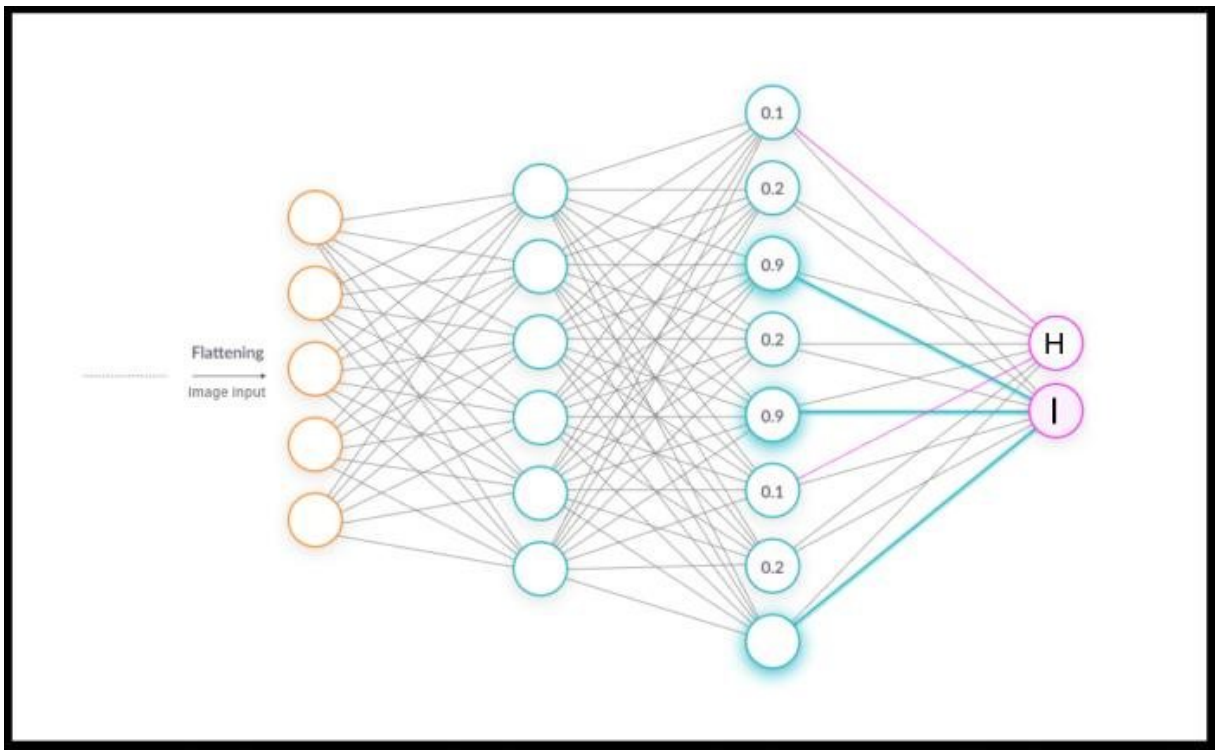


Figure 10: Fully connected layer

3.2 Media Pipe

3.2.1 Introduction to Media Pipe Hands

MediaPipe Hands (11) is a high-resolution tracking system for hands and fingers [15]. It uses machine learning to infer 21 3D hand landmarks from a single frame. This solution delivers real-time performance on a cell phone and even scales to many hands, whereas current state-of-the-art systems rely primarily on powerful desktop environments for inference.

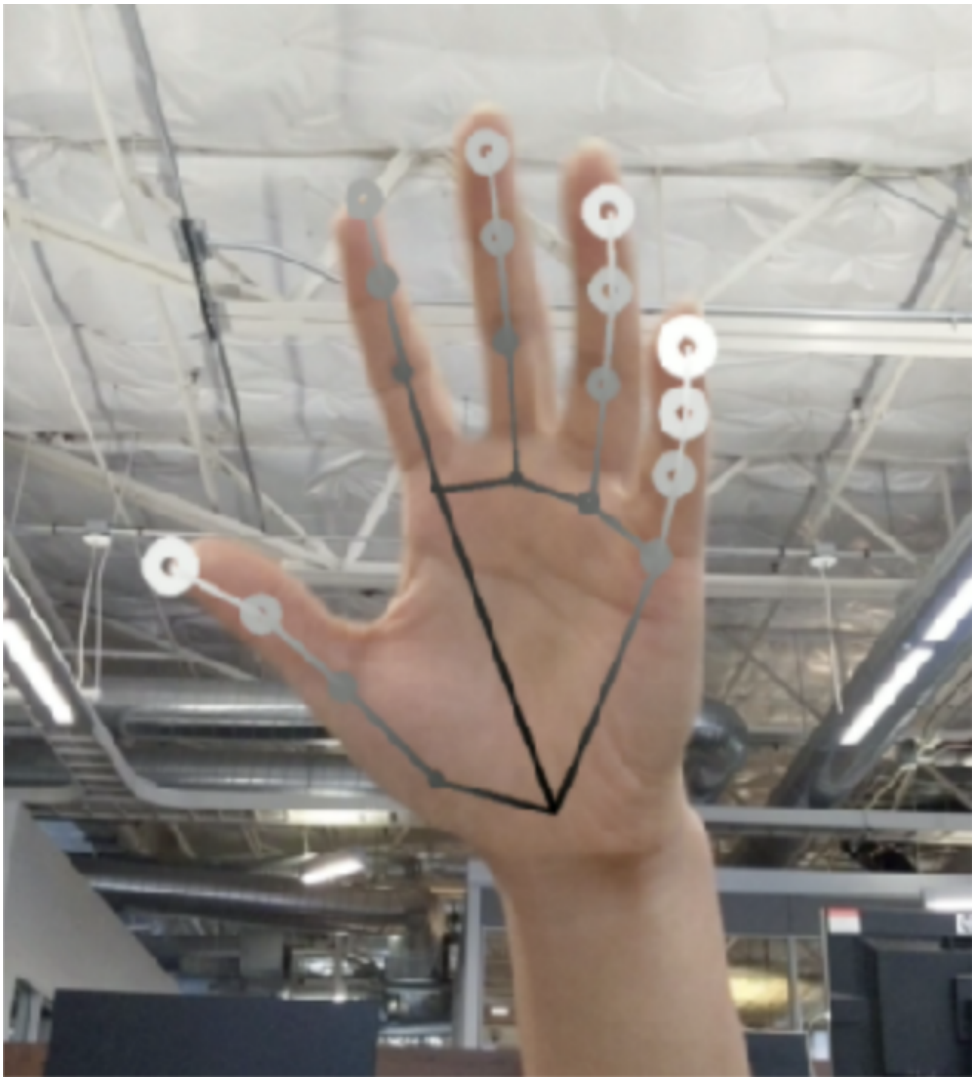


Figure 11: Media Pipe real time tracking 3D hand landmarks

MediaPipe Hands makes use of a machine learning pipeline that consists of several models that work together: A palm detection model, which acts on the entire image, will return an orientated hand bounding box. A hand landmark model that returns high-fidelity 3D hand key points from the cropped image region determined by the palm detector.

However, providing the hand landmark model with a correctly cropped hand image minimizes the requirement for data augmentation drastically (such as rotations, translations, and scaling) and instead, allows the network to focus on coordinate prediction accuracy. Furthermore, in this ML pipeline, crops can be created based on the hand landmarks recognized in the previous frame, and palm detection is only used to localize the hand when the landmark model can no longer detect its presence.

3.2.2 Palm detection model

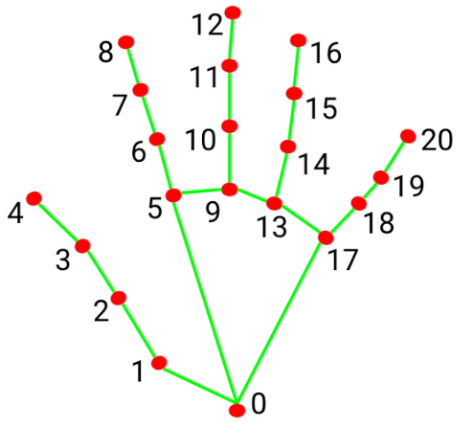
The Media Pipe team provides the palm detection model to detect initial hand locations and distinguish whether the hand recognized is left or right, which is very useful as each sign goes along with a different side will result in different meanings. They created a single-shot detector model, comparable to the face detection model in MediaPipe Face Mesh [8], tailored for mobile real-time applications. Hand detection is difficult: our model must detect occluded and self-occluded hands and work across many hand sizes with a significant scale span relative to the image frame.

According to their statements, the methods they used to address the above challenges vary in many strategies. First, instead of training a hand detector, they trained a palm detector because estimating bounding boxes of inflexible objects like palms and fists was much easier than recognizing hands with articulated fingers. Furthermore, the non-maximum suppression method performs effectively even in two-hand self-occlusion situations such as handshakes because palms are small objects. Furthermore, palms can be simulated using square bounding boxes (anchors in ML language) that ignore other aspect ratios, reducing 3-5 anchors. Second, even for tiny objects, an encoder-decoder feature extractor is used for more extensive picture context awareness (similar to the Retina Net approach). Finally, the significant scale variance limits focus loss during training to support many anchors.

Using the strategies described above gives an average precision of 95.7 percent in palm detection. With no decoder and a regular cross-entropy loss, the baseline is just 86.22 percent.

3.2.3 Hand landmark model

Following palm detection over the entire image, our next hand landmark model uses regression to accomplish exact key point localization of 21 3D hand-knuckle coordinates (see Figure 12) within the detected hand regions, i.e., direct, coordinate prediction. Even with partially visible hands and self-occlusions, the model develops a consistent internal hand posture representation.



0. WRIST
1. THUMB_CMC
2. THUMB_MCP
3. THUMB_IP
4. THUMB_TIP
5. INDEX_FINGER_MCP
6. INDEX_FINGER_PIP
7. INDEX_FINGER_DIP
8. INDEX_FINGER_TIP
9. MIDDLE_FINGER_MCP
10. MIDDLE_FINGER_PIP
11. MIDDLE_FINGER_DIP
12. MIDDLE_FINGER_TIP
13. RING_FINGER_MCP
14. RING_FINGER_PIP
15. RING_FINGER_DIP
16. RING_FINGER_TIP
17. PINKY_MCP
18. PINKY_PIP
19. PINKY_DIP
20. PINKY_TIP

Figure 12: 21 hand landmarks

3.3 Distance Matrix

A distance matrix [1] is a table that shows the distance between pairs of objects. For example, in the Figure 13., we can see the length between A and B is 16, B and C is 37, and so on. The diagonal of the table is the distance to the object from itself, so the value, as we can see, is 0. Distance matrices are sometimes called dissimilarity matrices.

Distance Matrix

	A	B	C	D	E	F
A	0	16	47	72	77	79
B	16	0	37	57	65	66
C	47	37	0	40	30	35
D	72	57	40	0	31	23
E	77	65	30	31	0	10
F	79	66	35	23	10	0

Figure 13: Distance matrix

3.3.1 Create Distance Matrix

A distance matrix is computed from a raw data table. In the example below (Figure 14), we can use high school math (Pythagoras) to work out the distance between A and B.

$$\sqrt{(24-9)^2+(54-49)^2} = 15.81 \approx 16$$

Figure 14: Calculating distance between A and B

We can use the same formula with more than two variables, known as the Euclidean distance. As a result, we have the distance matrix represented like Figure 15.

Raw Data

	X	Y
A	9	49
B	24	54
C	51	28
D	81	54
E	81	23
F	86	32

Distance Matrix

	A	B	C	D	E	F
A	0	16	47	72	77	79
B	16	0	37	57	65	66
C	47	37	0	40	30	35
D	72	57	40	0	31	23
E	77	65	30	31	0	10
F	79	66	35	23	10	0

Figure 15: The distance matrix is constructed from raw data

3.4 Beam search and Connectionist Temporal Classification

3.4.1 Connectionist Temporal Classification

Connectionist Temporal Classification (CTC) [4] is a type of Neural Network output helpful in tackling sequence problems like handwriting (Figure 16) and speech recognition where the timing varies. Using CTC ensures that one does not need an aligned dataset, which makes the training process more straightforward.

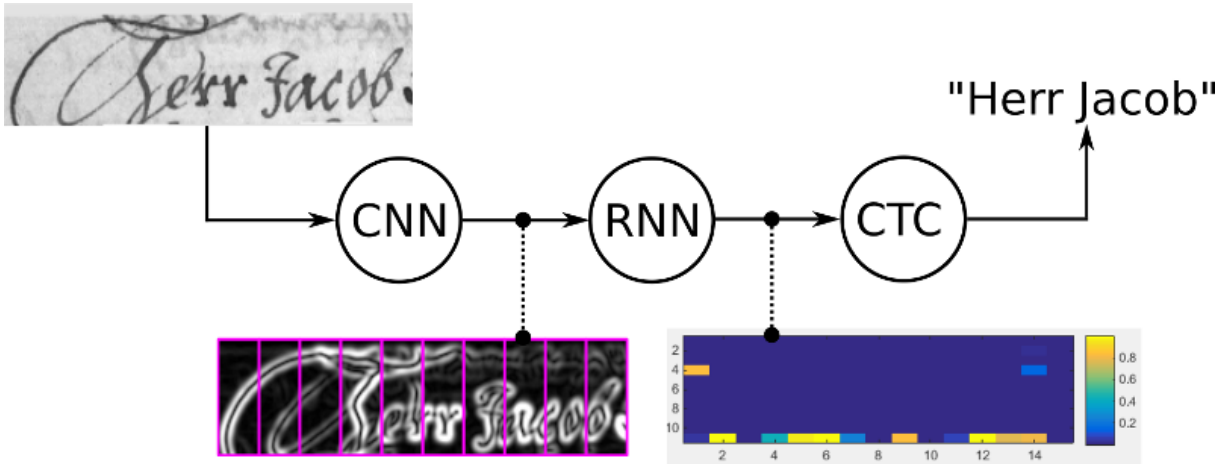


Figure 16: Overview of a Neural Network for handwriting recognition

3.4.2 Why we want to use CTC

In the context of handwritten recognition, we could create a dataset with images of text-lines, and then specify for each horizontal position of the image the corresponding character as shown in Figure 17 Then, we could train a model to output a character-score for each horizontal position. However, there are two problems with this solution.

- It takes much time, and annotating the dataset at the character level is tiresome.
- What if the character takes up more than one time-step ?. We could get "tooo" because the "o" is a wide-character as shown in Figure 17. We must remove all duplicate characters like "t" and "o".

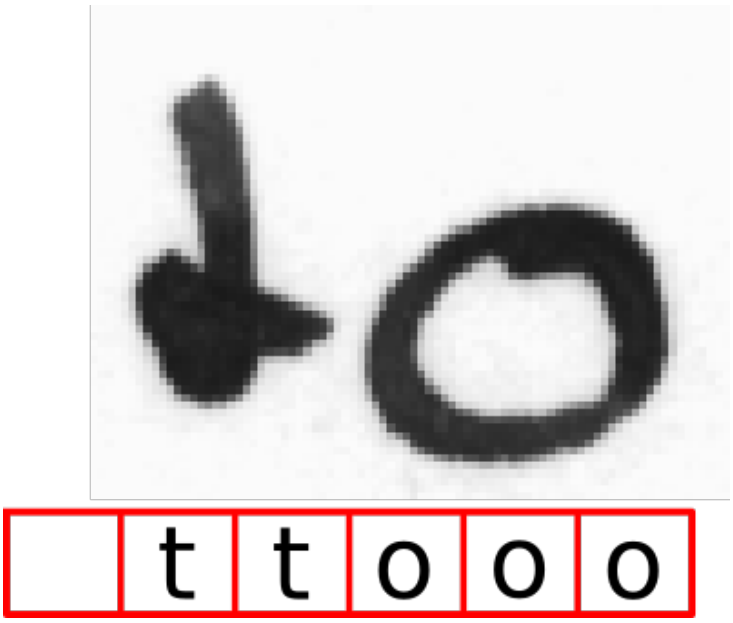


Figure 17: Annotation for each horizontal position of the image

CTC can solve both problems for us:

- We can ignore both the position and width of the character in the image and only requires the text that occurs in the picture.
- Using decode techniques, we can directly get the result of the network, and no further post-processing of the recognized text is needed.

3.4.3 Beam Search with CTC decoder

CTC has more than the Decoding phase, it can have the Encoding, Loss calculation, but we don't need it in the thesis scope anymore. So, here, we only mention to CTC decoder, but in the way, it combines with Beam Search [10]. Because CTC in decoding context can connect with another algorithm like best-path decoding, ...

Beam search

In computer science, beam search [2] is a heuristic search algorithm that explores a graph by expanding the most promising node in a limited set. Beam search is an optimization of best-first search the reduces its memory requirements. Best-first search is a graph search that orders all partial solutions (states) according to some heuristic. But in beam search, only a predetermined number of the best partial solutions are kept as candidates. Pseudocode for the basic version of beam-search is shown in Figure 18

Data: NN output matrix mat, BW
Result: decoded text

```
1  beams = {∅};
2  scores(∅, 0) = 1;
3  for t = 1...T do
4  |   bestBeams = bestBeams(beams, BW);
5  |   beams={};
6  |   for b ∈ bestBeams do
7  |   |   beams = beams ∪ b;
8  |   |   scores(b, t) = calcScore(mat, b, t);
9  |   |   for c ∈ alphabet do
10 |   |   |   b' = b + c;
11 |   |   |   scores(b', t) = calcScore(mat, b', t);
12 |   |   |   beams = beams ∪ b';
13 |   |   end
14 |   end
15 end
16 return bestBeams(beams, 1);
```

Figure 18: Basic version of Beam Search

The beam search algorithm will be implemented through the following steps, with two parameters will be included: output matrix and beam width (BW), which specifies the number of beams to keep. First, the beam list and corresponding score are initialized (lines 1 and 2). After that, from 3-15, the algorithm will loop over all time-steps of the matrix output. At this point, only the best scoring beams (equal BW) from the previous time-step are kept (line 4). For each beam, we calculate the score and get a result (line 8); we will cover this step in more details later. Further, each beam is extended by all possible characters from the alphabet (line 10), and again, a score is calculated (line 11). After the last time-step, the best beams are returned (line 16).

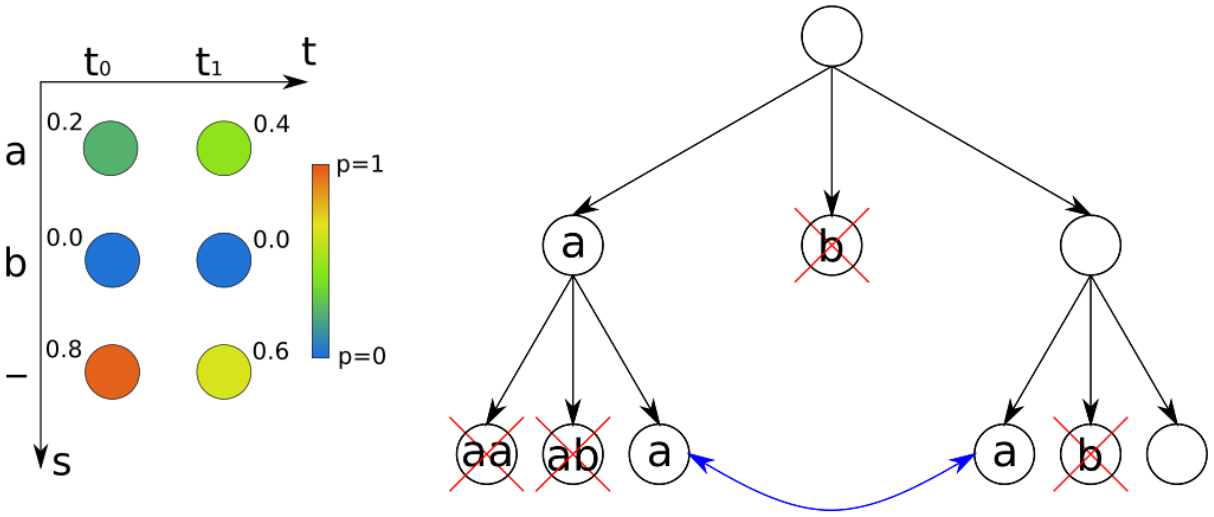


Figure 19: NN output and tree of beams with alphabet = "a", "b" and BW = 2

As we can see, in Figure 19, the output matrices are decoded, and the tree of beams is shown. Beam search algorithm extended as possible and keep exactly BW candidates. Finally, we finished the last iteration, and the final step of the algorithm is to return the beam with the highest score, which is "a" in this example.

Calculating the score

As discussed above, in this part, we will talk about how to score the beam. We will split the beam-score into the score of paths ending with a blank(e.g. 'aa-') and paths ending with non-blank (e.g. 'aaa').

- We denote the probability of all paths ending with a blank and corresponding to a beam b at time-step t by $P_b(b,t)$ and by $P_{nb}(b,t)$ for the non-blank case.
- The probability $P_{tot}(b,t)$ of a beam b at time-step t is simply the sum of P_b and P_{nb}, for example: $P_{tot}(b,t) = P_b(b,t) + P_{nb}(b,t)$

blank: 'aa-' +
- '-' = 'aa--' → "a" (copy)
- 'a' = 'aa-a' → "aa" (extend)
- 'b' = 'aa-b' → "ab" (extend)

non-blank: 'aaa' +
- '-' = 'aaa-' → "a" (copy)
- 'a' = 'aaaa' → "a" (copy)
- 'b' = 'aaab' → "ab" (extend)

Figure 20: The effect of appending a character to paths ending with blank and non-blank

In Figure 20, we will see what happens when we extend a path. Three main cases we can mention is:

- Extended by blank ('a' + '-' = 'a-')
- Extended by repeating last character ('aa' + 'a' = 'aaa' or 'aa-' + 'a' = 'aa-a')
- Extended by some other character ('aa' + 'b' = 'aab')

And when we collapse the extended paths, two results we will get and some cases we needed to handle:

- The unchanged (copied) beam ('a' $\rightarrow$ 'a'):
  - To copy a beam, we can extend corresponding paths by a blank and get paths ending with a blank: $P_b(n,t) += P_{tot}(b,t-1) * mat(blank,t)$
  - Besides, with the non-blank ending paths case, if we extend it by the last character (the beam is not empty): $P_{nb}(b,t) += P_{nb}(b,t-1) * mat(b[-1],t)$ with -1 indexes the last character in the beam
- An extended beam ('a' $\rightarrow$ 'aa' or 'ab'):
  - To extend a beam. With the last character is different from the character we need to extend, then there is no need for separating blanks ('-') in the paths: $P_{nb}(b+c,t) += P_{tot}(b,t-1) * mat(c,t)$
  - Or the last character of beam is repeated, we must ensure that the paths end with a blank: $P_{nb}(b+c,t) += P_b(b,t-1) * mat(c,t)$
  - We don't need to care about $P_b(b+c,t)$ because we added a non-blank character

Putting it all together

The CTC beam search algorithm is depicted in the Figure 21. It is similar to the basic version that was previously shown. It does, however, contain the code for scoring the beams: copied beams (lines 7-10) and extended beams (lines 15-19). Finally, in order to find the best scoring beams, the program rates them using the Ptot (line 4) and then selects the best beams (BW).

Data: NN output matrix mat, BW and LM
Result: decoded text

$beams = \{\varnothing\}$;
$P_b(\varnothing, 0) = 1$;
for $t = 1...T$ **do**
  $bestBeams = bestBeams(beams, BW)$;
  $beams$={};
  for $b \in bestBeams$ **do**
    if b != $\varnothing$ **then**
      $P_{nb}(b,t)$ += $P_{nb}(b, t-1) \cdot mat(b(-1), t)$;
    end
    $P_b(b,t)$ += $P_{tot}(b, t-1) \cdot mat(blank, t)$;
    $beams = beams \cup b$;
    for $c \in alphabet$ **do**
      $b' = b + c$;
      $P_{txt}(b') = applyLM(LM, b, c)$;
      if $b(t) == c$ **then**
        $P_{nb}(b',t)$ += $P_b(b, t-1) \cdot mat(c, t)$;
      else
        $P_{nb}(b',t)$ += $P_{tot}(b, t-1) \cdot mat(c, t)$;
      end
      $beams = beams \cup b'$;
    end
  end
end
return $bestBeams(beams, 1)$;

Figure 21: CTC beam search

3.5 Technology

Overall, the product app has been developed with technologies such as Java for Android apps and Firebase system authentication. Furthermore, to keep the application as light as possible, we use only native components and no external UI libraries.

3.5.1 Java

Figure 22: Java logo

James Gosling developed Java at Sun Microsystems, Inc. in 1995, later acquired by Oracle Corporation. It's a straightforward programming language. Java makes programming easy to write, compile, and debug. It aids in the development of reusable code and modular programs.

Java is an object-oriented programming language based on classes that are designed to have as few implementation dependencies as feasible. Furthermore, Java is a general-purpose programming language that allows developers to write code once and run it on any device that supports Java. Java programs are compiled into byte code that may be run on any Java Virtual Machine. Besides, Java has a syntax that is similar to C and C++.

3.5.2 Firebase

Figure 23: FireBase logo

Firebase is a web and mobile app development platform with simple and powerful APIs that don't require a server or backend. Firebase is a cloud-based platform. Also present is Google's server system. Its main goal is to make database operations simpler for users so they can program apps more easily. The real-time database service allows users to store and synchronize data. This is a completely cloud-based service. If the device is offline, it will use up its memory before automatically syncing with the server once it is online.

This feature largely comprises of backend services that help developers construct and manage their apps more effectively. The following services are included in this feature:

- **Realtime database:** The Firebase Realtime Database is a cloud-based NoSQL database that processes data at millisecond speed. In the simplest sense, it can be thought of as a large JSON file.
- **Cloud firestore:** The Cloud Firestore is a NoSQL document database that allows users to store, sync, and query data from anywhere in the world using the app. It stores information in the form of documents, which are objects. It stores any data type, including text, binary data, and even JSON trees, using a key-value pair.
- **Authentication:** Using UI libraries and SDKs, the Firebase Authentication service makes it simple to authenticate users in the app. It reduces the time and effort required to develop and maintain the user authentication service. It even handles processes like account mergers, which can take a long time if done manually.
- **Remote configuration:** The remote configuration service speeds up the distribution of updates to users. Changes could range from updating UI components to altering application functionality. These are frequently used to deliver limited-time offers and content to a mobile application.
- **Hosting:** Firebase provides fast and secure application hosting. It can host both static and dynamic websites, as well as microservices. With a single command, it can host an application.

Chapter 4

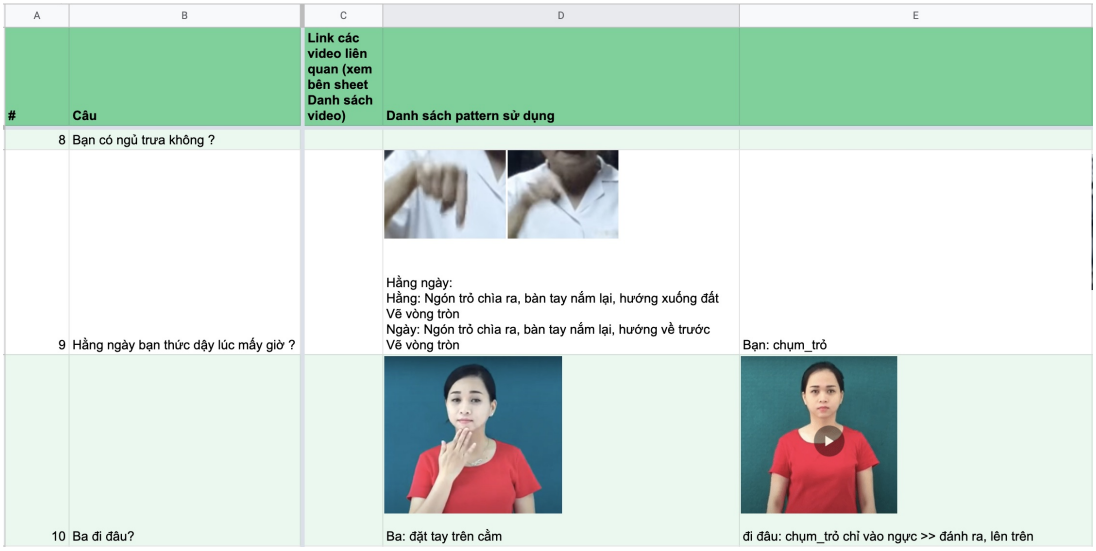
Design and Solution

4.1 Gathering Data

Before we can design a system that can translate sign language, we must first understand what a sign language word is and where to collect the data. After observing the sign language on the websites `https://tudienngonngukyhieu.com/` and `https://qipedc.moet.gov.vn/`, we noticed that some of the words have the same pattern. Furthermore, the sign's meaning is determined by its orientation and location. Furthermore, some of the terms require hand or finger movements to represent the meaning. As a result, using those four factors, we can convert a word from sign language to Vietnamese.

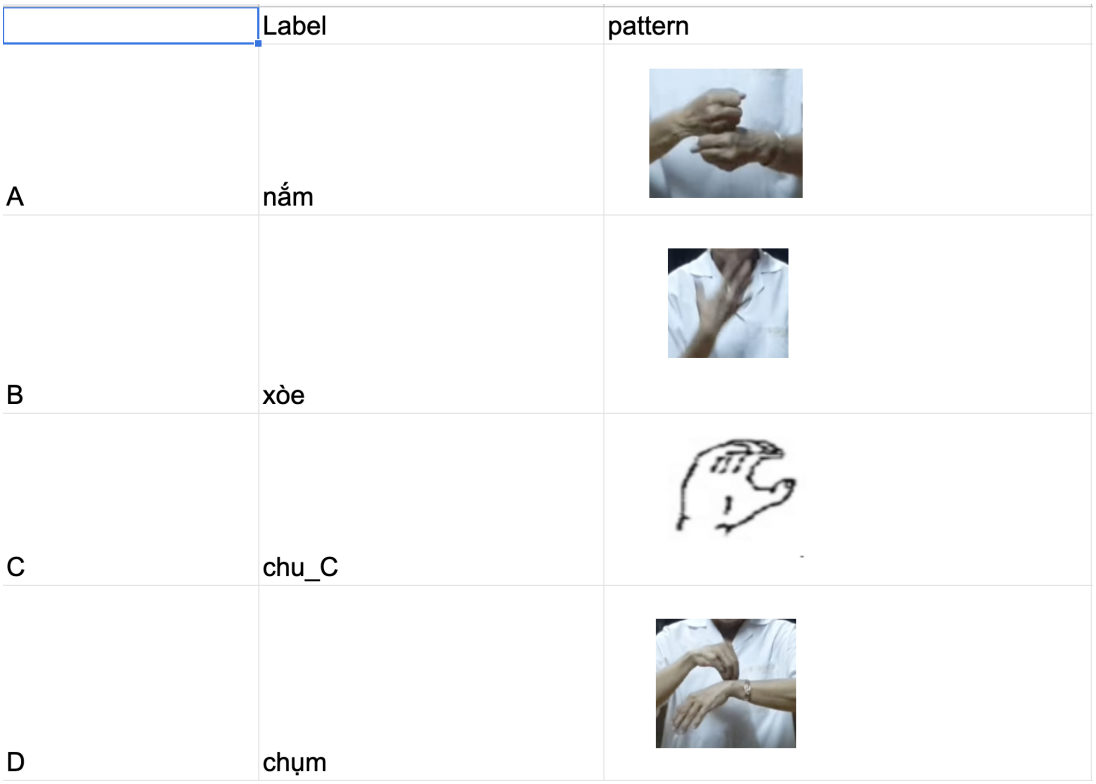
However, we must first collect sign language data for the thesis's model training phase. Fortunately, those two websites contain a sufficient number of sign language words. We also learned a few words from YouTube videos taught by Mrs. Le Thi Thu Xuong [14] and channel CDS, Central Deaf Services in Dang Nang, Vietnam [11]. As a result, we can independently train and test our system.

We gathered a large number of words from those two websites and YouTube channel to prepare the data. Then we label and divide it into many elements, which we will discuss later in the Section 4.3.4. For the data we collected, we created a Google Sheet. We have prepared many labeled words in this file (see Figure 24). In addition, we have a sheet for many different hand shapes (see Figure 25), which helps us classify hand patterns.



#	Câu	Link các video liên quan (xem bên sheet Danh sách video)	Danh sách pattern sử dụng	
8	Bạn có ngủ trưa không ?			
9	Hằng ngày bạn thức dậy lúc mấy giờ ?		Hằng ngày: Hằng: Ngón trỏ chìa ra, bàn tay nắm lại, hướng xuống đất Vẽ vòng tròn Ngày: Ngón trỏ chìa ra, bàn tay nắm lại, hướng về trước Vẽ vòng tròn	Bạn: chụm_trỏ
10	Ba đi đâu?		Ba: đặt tay trên cằm	đi đâu: chụm_trỏ chỉ vào ngực >> đánh ra, lên trên

Figure 24: Google sheet about words labeled



	Label	pattern
A	nắm	
B	xòe	
C	chu_C	
D	chụm	

Figure 25: Google sheet about hand shapes can be recognized

4.2 System Structure

Overall, the system is composed of three hardware modules: a camera module, the user's smartphone, and the server. The server, which will be the focus of this thesis, is one of those

modules that handles the most complex work.

Our artificial intelligence system for sign language translation consists of six major modules: hand pattern recognition, direction determination, location detection, action detection, word decoder, and text to speech (Figure 26). To begin, the system continuously captures the motion of the hand, processes it with the hand landmark model, and then stores it in those modules. Each of them plays a distinct role. The word decoder module will take the output data and produce the corresponding outcome after combining the results of the first four modules (hand pattern, direction, location, and action detection). The result will then appear on the main screen, while the phone will speak out that word.

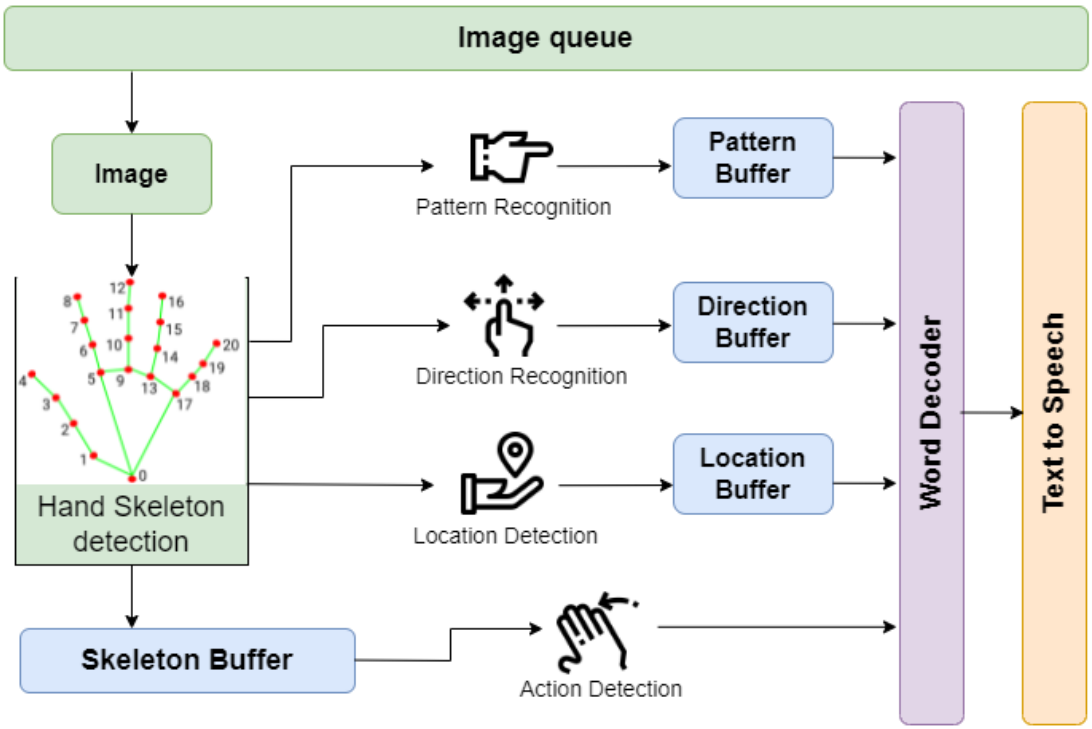


Figure 26: Overview of the old system structure

Those six major modules mentioned above are the ones we had planned for at the start of the thesis. Despite going through most of the modules, we found it difficult to build the action detection module during the implementation period. It is problematic because of the demands it places on the smartphone and server.

There are some sign language words that contain a lot of moving patterns. The first method is to send the entire video captured by the camera to the server for processing. This method, on the other hand, necessitates a strong connection between the camera module, smartphone, and server and places strain on the physical devices (the camera and smartphone); as a result, those devices will become hot and may be damaged in some way. Another option is to increase the frame rate to get the action, but this can put the devices under strain. Furthermore, we must have an algorithm that is constantly processing and detecting movements, which we admit is difficult

to achieve. To solve the problem, we had to deprecate that module and change our method for obtaining the correct Vietnamese word. Instead of combining the four modules, including the action detection module, there are now only three remaining: pattern, direction, and location. Furthermore, in the word decoder module, we use the beam search heuristic search algorithm, which uses the results of the three modules to look up the word in the database and return it to us. Section 4.3 will go over each module's role and how it works.

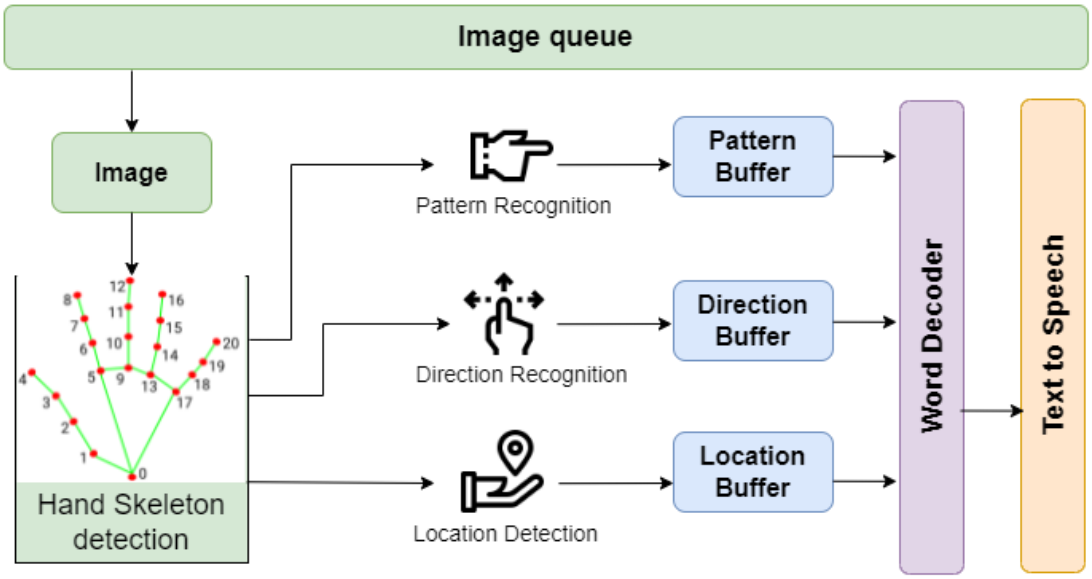


Figure 27: Overview of the new system structure

4.3 Detail Implementation

4.3.1 Hand pattern recognition

The first and most fundamental module of this system is hand pattern recognition. When a person with a disability performs sign language, their hands move in a variety of ways, with their hands spread out, clenched, or their fingers pointing out at something. As a result, the role of this module is to recognize the hand pattern. The system can then provide the final result by combining the outcome with other modules.

This module makes use of the hand landmark model's output, which has a matrix size of 21. We get a new matrix representing the distance between those 21 coordinates after calculating all of the values in that matrix. As seen in Figure 27, using the distance matrix as the input of CNN with the designed structure (see Figure 28) will tell us the pattern of the hand at the time it is captured.

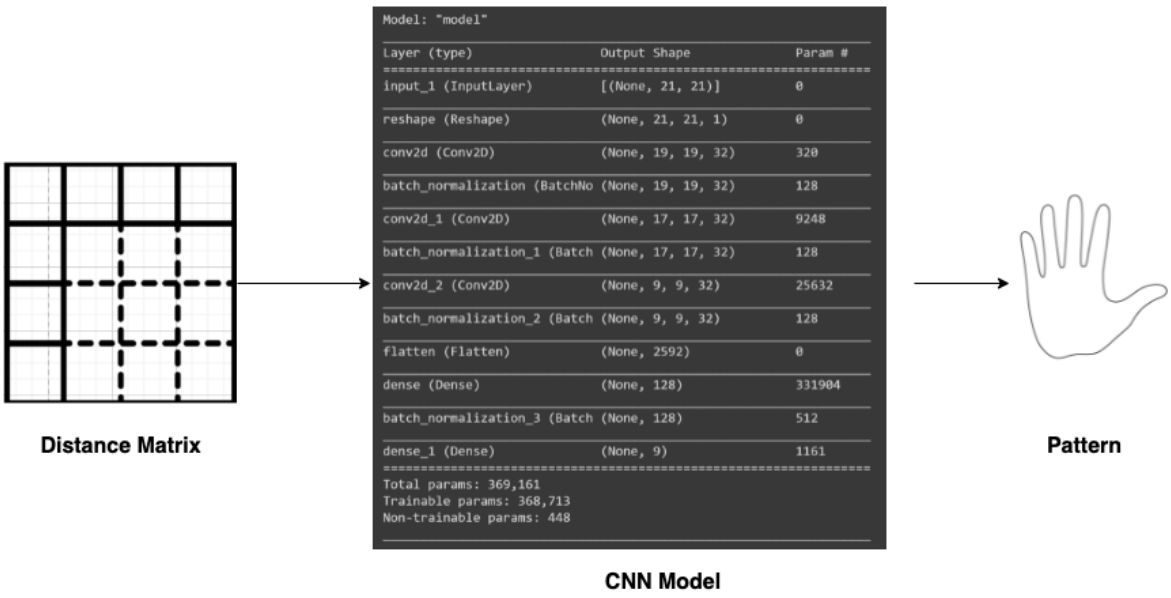


Figure 28: Hand pattern recognition pipe line

4.3.2 Direction determination

The hand has four directions, which are right, left, up, down, front, and back. The pattern of each hand, when combined with different directions, yields a different meaning. For example, the pattern that points at someone means the word "you," whereas the pattern that points at ourselves means the word "I." (see Figure 29 and Figure 30).

Figure 29: Word "You" (bạn) in sign language

Figure 30: Word "I" (tôi) in sign language

We use the hand landmark model provided by MediaPipe to determine the direction of the hand (see Section 3.2). The idea here is to calculate the distance between the tip of the index finger and the wrist, which is known as a **vector(0, 8)**, and then project it to the axes Ox, Oy, and Oz, in that order. Following that, we compare those coordinates to the others. Finally, the one of immense value will tell us which axis the hand is on; additionally, we will know which direction the hand is by projecting the direction from the wrist to the tip of the index finger on that corresponding axis.

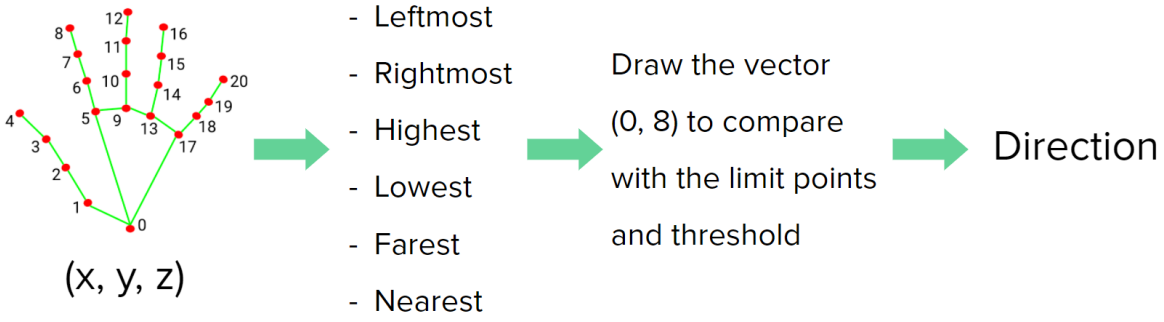


Figure 31: Steps to detect the direction of the hand

For instance, a hand is pointing in the left direction. When projected on the axis Ox, the distance value will be the largest of the three projected values. Then, compute the vector drawn from the wrist to the tip of the index finger; we'll know the hand's direction.

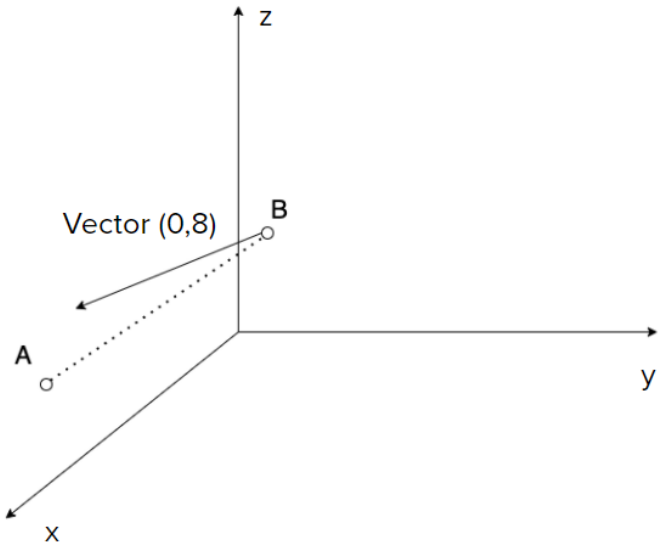


Figure 32: Vector(0, 8) represent the hand pointing toward the left

4.3.3 Location detection

Hand placement differs; is the hand on the brow, mouth, or chest level, and so on. Each hand pattern associated with each location will yield a different set of words. Nonetheless, with only one camera and a view from above, it is difficult for the AI to determine the coordinates of the hand (see Figure 33). However, we devised some solutions to this problem.

Figure 33: View from the camera module

The first solution to detect the hand's location is using an ultrasonic sensor. In short, this sensor is an instrument that measures the distance to an object using ultrasonic sound waves (see

Figure 34). It works by emitting a sound wave with a frequency above the human hearing range. The sensor's transducer functions as a microphone, receiving and transmitting ultrasonic sound. The sensor measures the time between sending and receiving an ultrasonic pulse to determine the distance to a target.

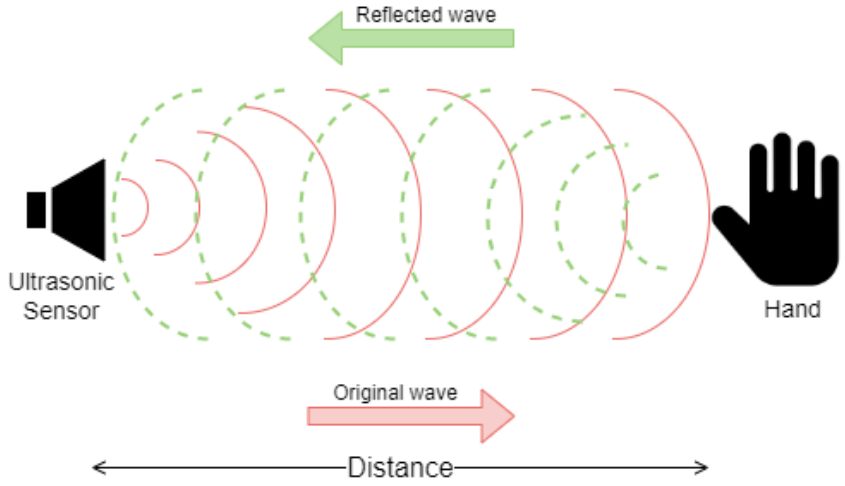


Figure 34: Illustration of how the ultrasonic sensor works

In the thesis, we use the ultrasonic sensor HY-SRF05 (Figure 35), which is relatively inexpensive and meets our requirement for measuring the distance between the camera and the hand. According to the retailer, this sensor has a scanning range of up to 15 degrees. Furthermore, its scanning range is between 2 cm and 450 cm, with a relative error of around 0.3 cm. Furthermore, the most accurate measurement distance is less than 100 cm, which is more than enough to measure from the user's forehead to their waist.

Figure 35: The ultrasonic sensor HY-SRF05

However, the sensor method has yet to produce the desired result. The sensor can only detect one hand at a time, but what if sign language uses both hands at the same time? In this case, we cannot achieve the best possible results with only one sensor.

From there, we propose the following solution for measuring this distance. Replace the distance measurement method based on object size with one that uses a location sensor. It is similar to the zooming method, but it is superior in that it does not depend on the size of the hand, making it ideal for determining the distance for the problem we are attempting to solve.

This method is as follows. We will use the following formula:

$$distance\_predict = A * distance\_input + B * distance\_input + C$$

In which *distance_input* will be the distance between two points (5,17) on the hand model taken from the MediaPipe model, and *distance_predict* will be the distance from the camera to the hand.

The trio of coefficients A, B, and C will be determined by interpolating the above polynomial with a pre-prepared data set. After having the above three coefficients, combined with calculating the distance between 2 points 5 and 17, we will quickly deduce the hand's distance.

Figure 36 shows the result we get

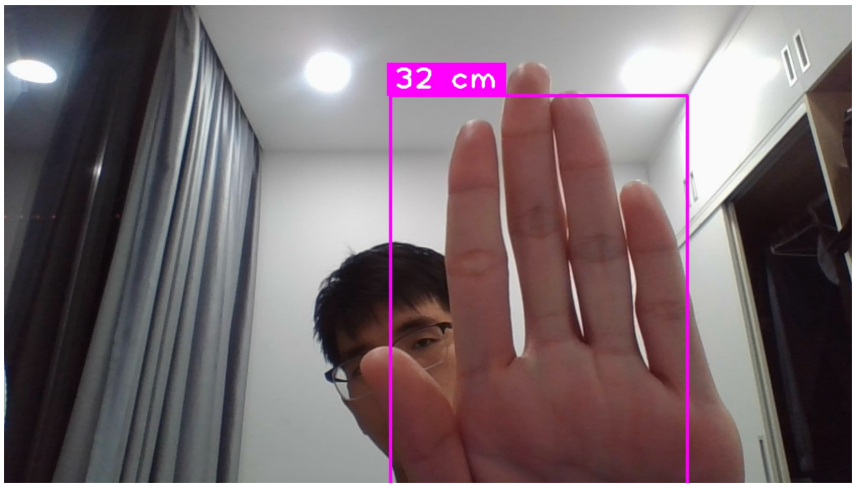


Figure 36: Result of location module

As a result, after receiving the result from this location module, we placed it in the hand state. Similarly, in the Section that follows, we will discuss how that hand state will aid us in translating sign language into Vietnamese.

4.3.4 Word decoder

As previously discussed, there are significant technical challenges in implementing the action detection module. We conducted research and proposed a new model to address these issues. As a result, this change has an impact on the word decoder module, which requires some tweaking.

The previous model breaks down a word into four components: pattern, location, direction, and action. It will search the database for the corresponding word after receiving the outputs

from the four modules. Figure 37 demonstrates how a four-factor input is mapped to the correct word in the database. We have the system use a basic searching algorithm to find the most appropriate word. If none are found, it will replace or deprecate some parts of the input before attempting to find another word. The application will display the appropriate word on the screen after decoding it.

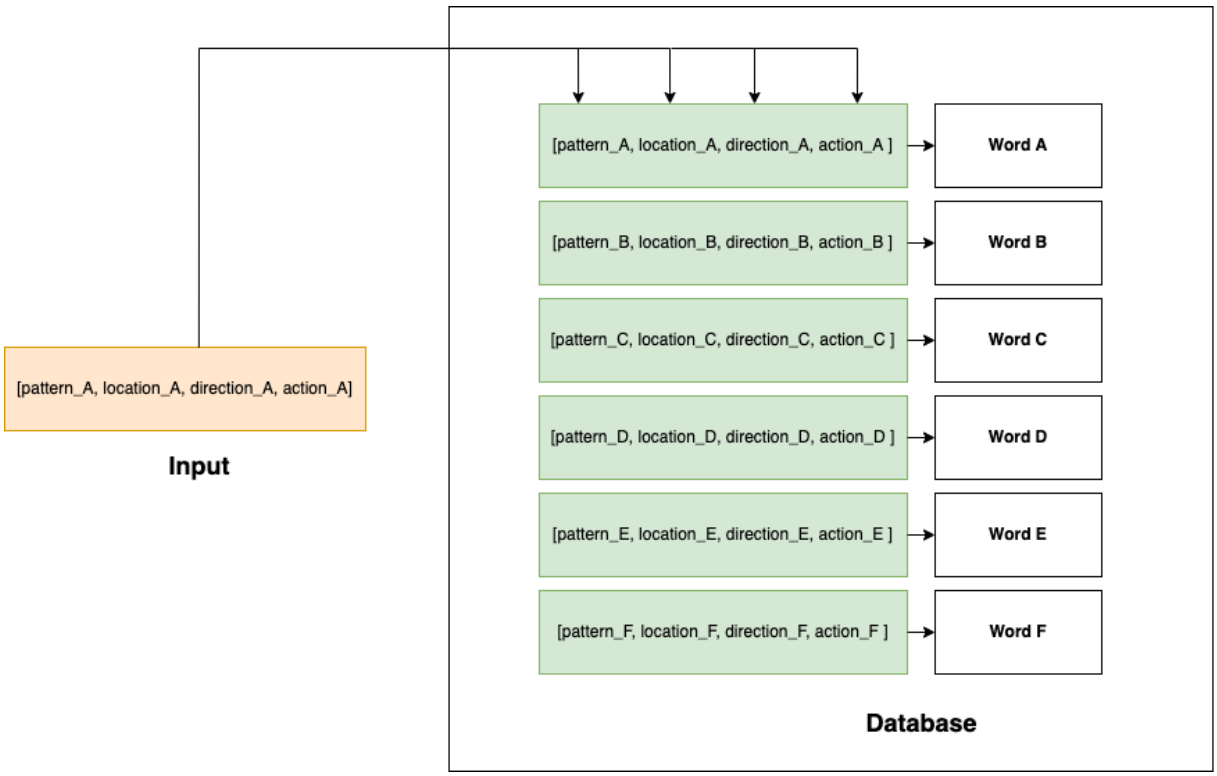


Figure 37: Map one to one data from four component with word in database and get result

Introduction to handstate

The question that arises immediately following the deprecation of the action detection module is how we can find the correct word without that module. As a result, unlike the previous model, we propose a different model for a word that is not encoded into four factors. It only has three remaining elements: pattern, direction, and location. As a result, each combination of those three elements is referred to as a hand state (Figure 38), and a word can be decoded into a variety of hand states.

This concept of hand state stems from natural language processing research, in which a word is made up of many characters. As a result, a word is concatenated from several hand states in the thesis. Then, after going through the processing steps discussed later in this proposal, we will get the desired word.

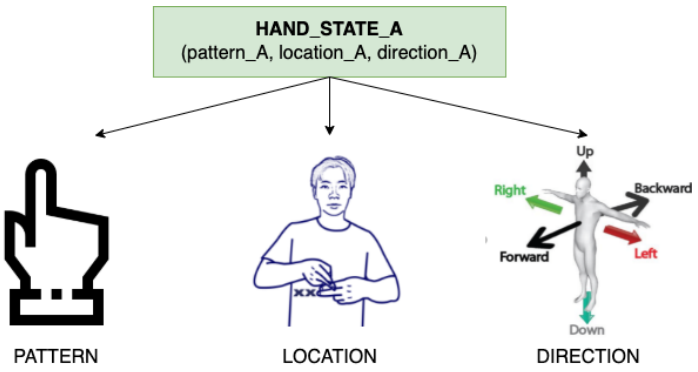


Figure 38: Hand state which construct from pattern, location and direction

Using beam search and CTC decode to map word

After we have grasped the concept of hand state, we will proceed to the most important part of the model: converting the received hand states into words.

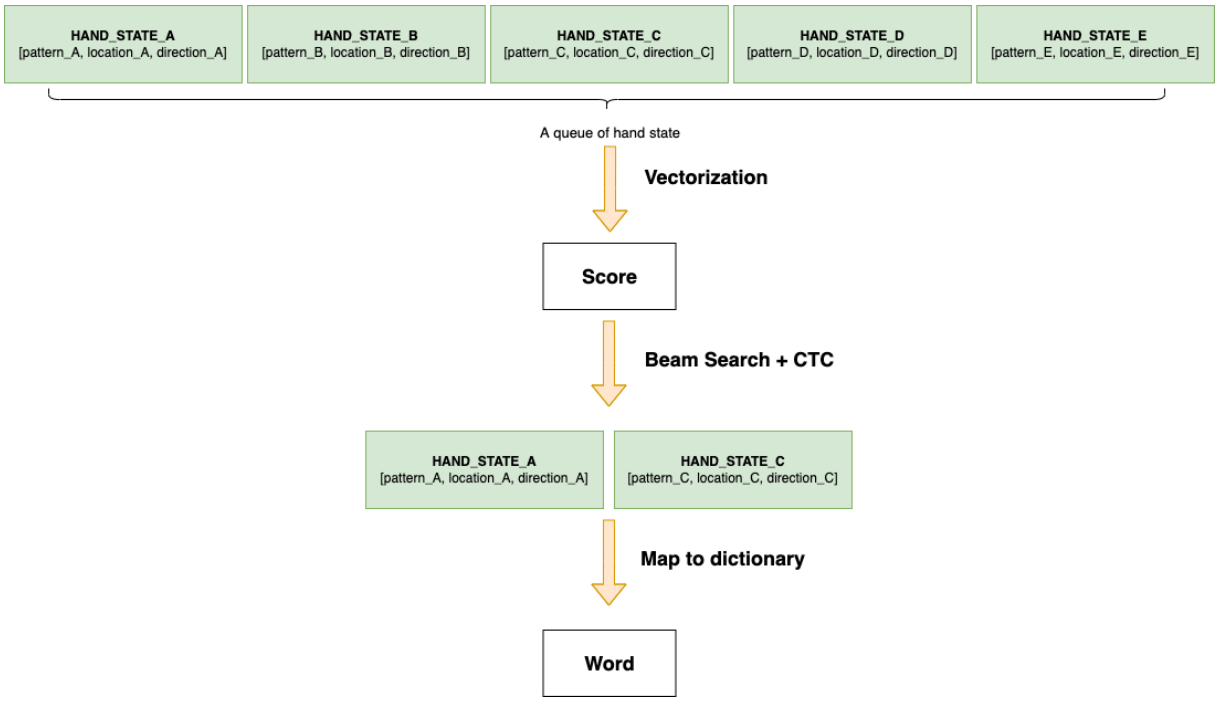


Figure 39: Architecture

The authors' model for this Section is shown in Figure 39. A queue of hand states from the previous three components will be used as input. The queue length is set to 5 in our conventions, but this is not the final number because we need more calculations and experimentation to find the optimal queue length. This model is comprised of three steps:

1. **Vectorization:** This step converts a queue of many hand states (Figure 40) into a matrix for

beam search input.

2. **Beam search:** In this step, we will use a beam search algorithm to determine which hand states are appropriate for the database input. Furthermore, we propose using the CTC decode model to eliminate incorrect or previously duplicated hand states, increasing the model's efficiency.
3. **Map to the dictionary:** Finally, after going through the preceding two steps, we will obtain the most likely hand state from the initial queue. Our job is to match these hand states to the correct word in the database.

Vectorization

We get a list of hand states when we get to this step. Because we need a matrix representing the correlation between the component outputs and the data in the database before entering the beam search module.

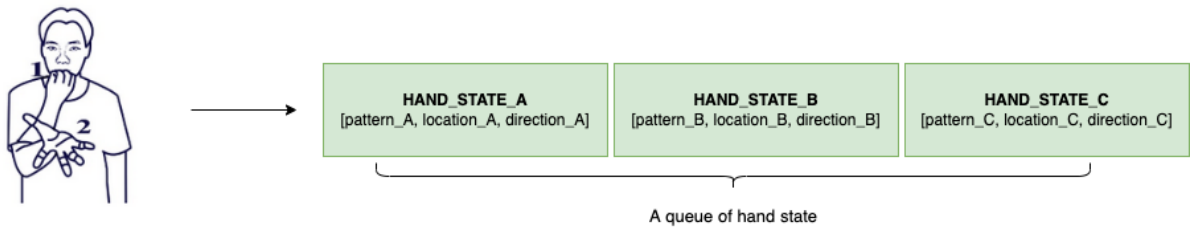


Figure 40: A queue of hand states to sign "Thank you"

From this queue, we will cycle through each hand state, compare it to the available hand state database, and calculate the score based on the principles shown in Figure 41:

1. If the word in the hand matches the word in the database, the score will be increased.
2. Otherwise, if that hand state does not match any, the score will be reduced.

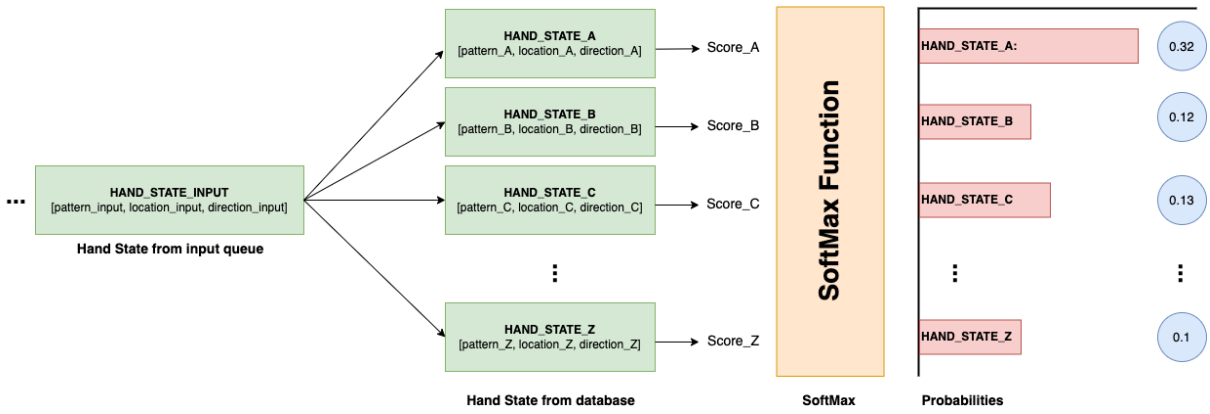


Figure 41: Vectorization

The first principle states that the higher the score, the more similar the hand state retrieved from the queue is to that in the database. For example, in the database, we have the following hand state:

$\begin{bmatrix} pattern\_A & location\_A & direction\_A \end{bmatrix}$, and the hand state we get from the input is $\begin{bmatrix} pattern\_A & locati$ then this hand state will be rated higher than the hand state $\begin{bmatrix} pattern\_A & location\_A & direction\_B \end{bmatrix}$. And so forth. In turn, we will score the hand states taken from the queue.

The minus point is evaluated on the second point based on its matching pattern with the hand states in the database. When the system recognizes patterns from the hand pattern recognition module (via the vision approach), they are likely to be incorrectly detected. We devised a rule to address this issue and improve the accuracy of the results. The minus point will be lower for patterns that are difficult to detect than for simple ones. In short, the smaller the minus point, the more complex the pattern to be recognized.

Following the completion of the preceding evaluation and scoring steps, we will use a function to normalize the data (here, the authors use the softmax function [13]) and return a set of probabilities for the hand states in the row. Wait. This set of probabilities will be used as input for the beam search step.

Using beamsearch with CTC decode

The hand states in our queue have been converted to a MxN matrix after passing the vectorization step, where M is the length of the hand state's database and N is the length of the queue.

We will retrieve the most likely k-hand state from the database by using beam search (Figure 42). We can imagine what happened later during the beam search based on the image below.

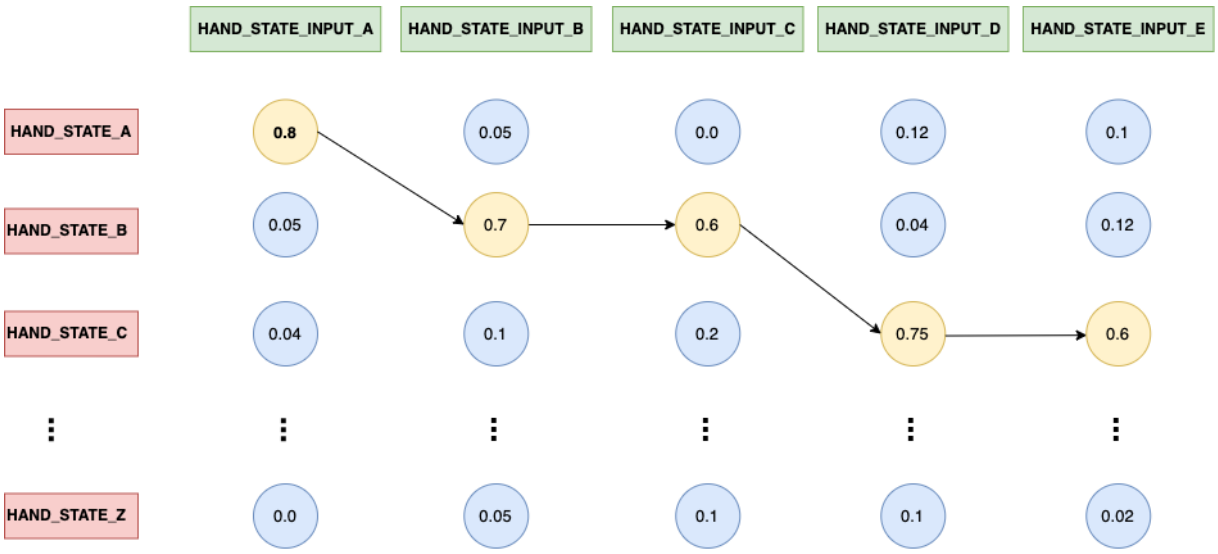


Figure 42: Beam Search

However, as we can see, after the beam search step, we will get a sequence of hand states with a length equal to the length of the input queue. These hand states may contain duplicate hand states or incorrect hand states. As a result, we must use the CTC algorithm to remove the hand states from infection. We will set a threshold in the vectorization step to discard these hand states and treat them as a blank character for the incorrect hand states. Furthermore, we will get the desired result after beam search CTC decode.

Map to dictionary

Figure 43: Map to dictionary and get word

All that remains is for us to map to the database, as we did in the previous Section, but instead of mapping with a 4-component set, we will map with input retrieved from this previous step. Finally, without using the action detect module, we will obtain the word (Figure 43).

4.3.5 Text-to-speech

Furthermore, because people do not always read the result from the phone's screen, the application can speak it out loud to make it easier for them to know the answer after the translation process. One method is to create a database of many sound files that have been tagged with the

corresponding Vietnamese word. This approach, however, is inefficient because it necessitates a significant amount of effort to create the database. Every single word must be recorded and then mapped together.

Instead, we apply a well-known speech service from Google known as Text-to-Speech[3]. Text-to-Speech, according to Google, converts text input into natural human speech audio data. Furthermore, this service supports a wide range of languages, including the one we require, Vietnamese. Using the API provided by that speech service, our application can speak up the result without requiring a large sound database on the user's phone.

This API is available for free. The cost of text-to-speech is determined by the number of characters sent to the service each month to be synthesized into audio. According to Google, "the first 1 million characters for WaveNet voices are free each month." Each month, the first 4 million characters for standard (non-WaveNet) voices are free. After the free tier, Text-to-Speech is charged per 1 million text characters processed. The thesis only requires the standard (non-WaveNet) plan, which includes 4 million characters for free each month and costs USD 4.00 for each additional 1 million characters. The number of characters we will use in the upcoming phases of building the application is negligible in comparison to the 4 million free characters. As a result, we decided to use the Google service to implement this Text-to-Speech module.

4.3.6 The camera module

After discussing the solutions and implementations of the system's soft modules, we must move on to the main one, which is considered the thesis's eyes, which is the camera module. This Section will go over the components of a camera module and show some images of a real one that we built.

Parts for the camera modules are easily available from any retailer that sells electrical components, robots, and Arduino kits. Furthermore, in this day and age of e-commerce, it is much easier to find and compare the components that we require online. The components needed to construct a camera module are listed below.

First, we require a camera component, which the ESP32-CAM is ideal for. It is inexpensive and simple to use, which makes it ideal for our thesis, which requires complex functions such as image tracking and recognition. Furthermore, it incorporates Wi-Fi and traditional Bluetooth, allowing us to send images to the user's smartphone for the next steps in sign language translation.

Second, we will require a converter adapter to assist us in sideloading the program into the camera module. The ultrasonic sensor mentioned in the Section 4.3.3 is the third component we require. It aids in location detection by informing the system of the distance between the hands

and the camera module. Last but not least, this camera module requires a battery to power the entire module, and we believe that a volume of around 100 mAh is adequate.

Furthermore, all of the above components must be stored in a box. We designed that package on Tinkercad, an online 3D modeling program that runs in a web browser, using current 3D printing technology. After gathering all of the necessary components, we attempted to put them together and obtained the result shown below.

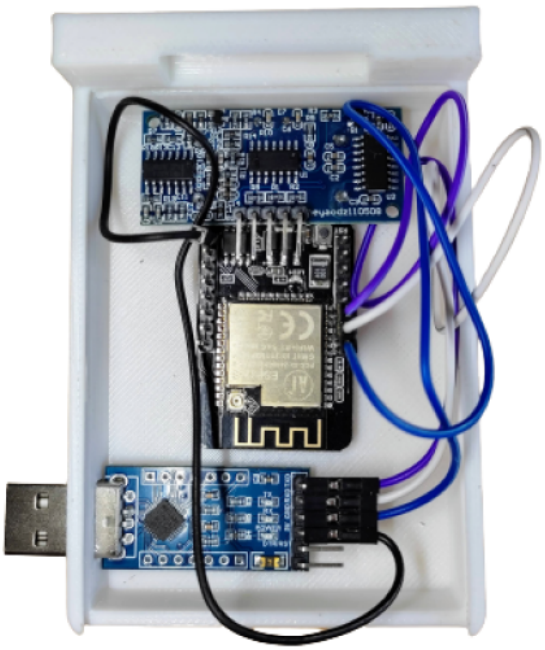


Figure 44: The components inside the camera module prototype

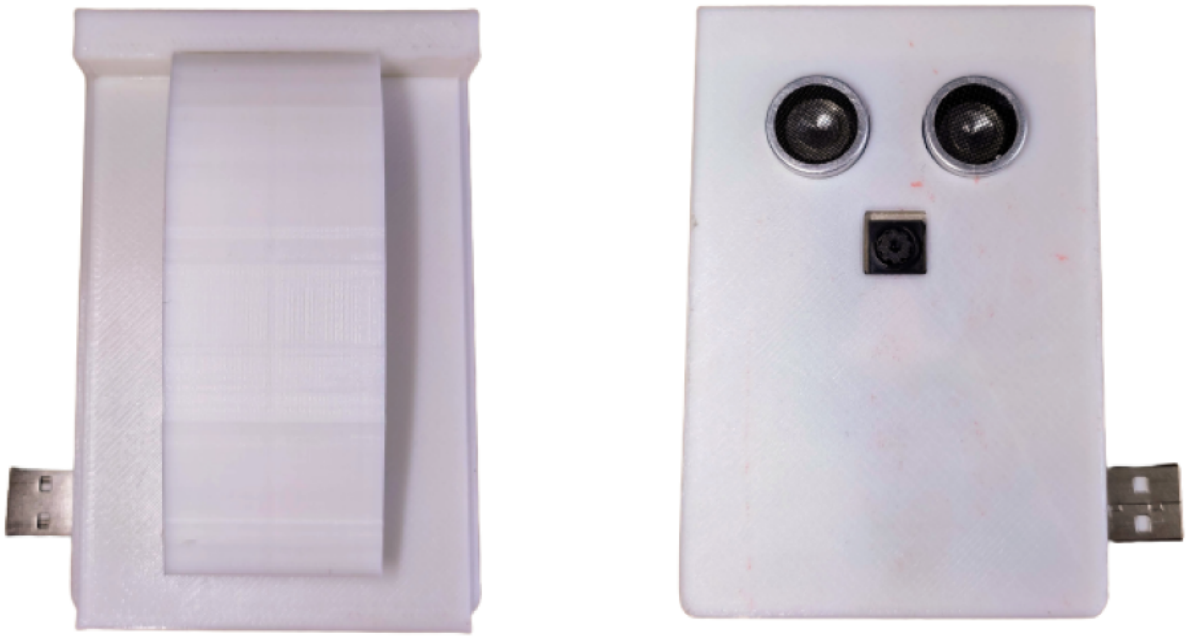


Figure 45: Views of the camera module prototype from the above and under

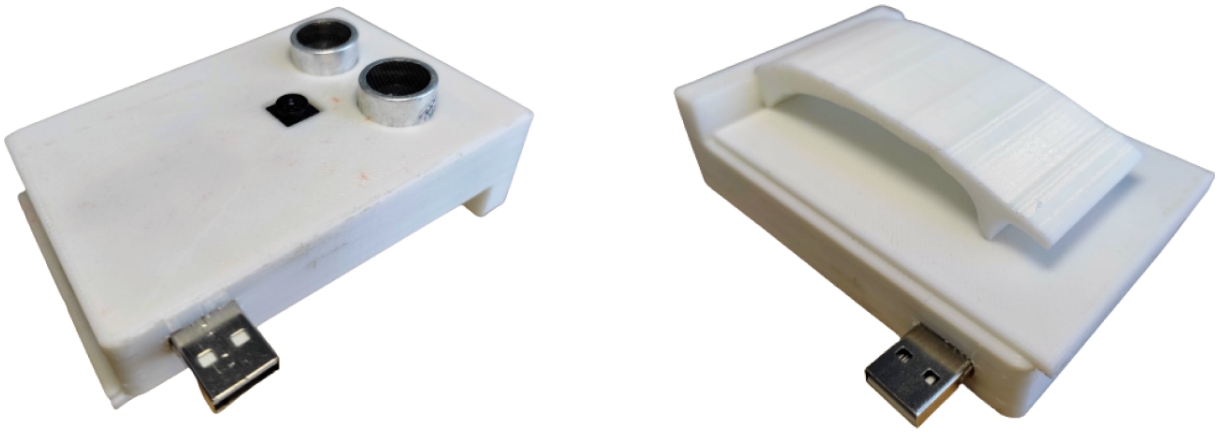


Figure 46: Views of the camera module prototype from the sides

Nonetheless, the box's cover is difficult to insert due to the smallest number that a 3D printer can print. The hanger that assisted in hanging the box on the hat is not as flexible as we had hoped, so it needs to be redesigned.

4.3.7 App design

The application must meet user experience and user interface requirements. And, as part of the thesis research, we created a prototype for the application, which included one additional

feature in addition to the main one. A sign language dictionary is one of the extra features. Table 7 demonstrates this feature.

Before delving into the design of this application, it is important to note that they do not yet cover all of the screens required for the application. And they do not represent the final design that we have. However, when designing this prototype, we followed some conventions, such as rounding the corners and keeping the colors pale and not too bright, in order to make the users feel calm and at ease while using the application.

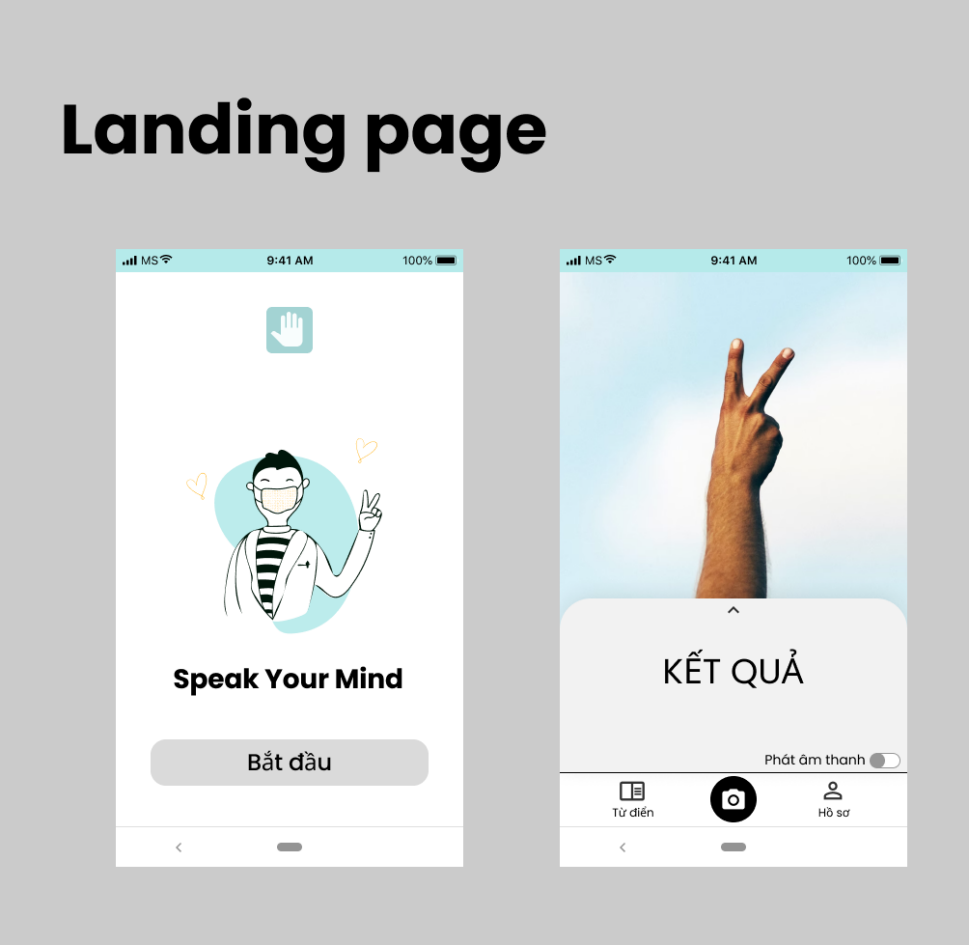


Figure 47: The landing page of the application

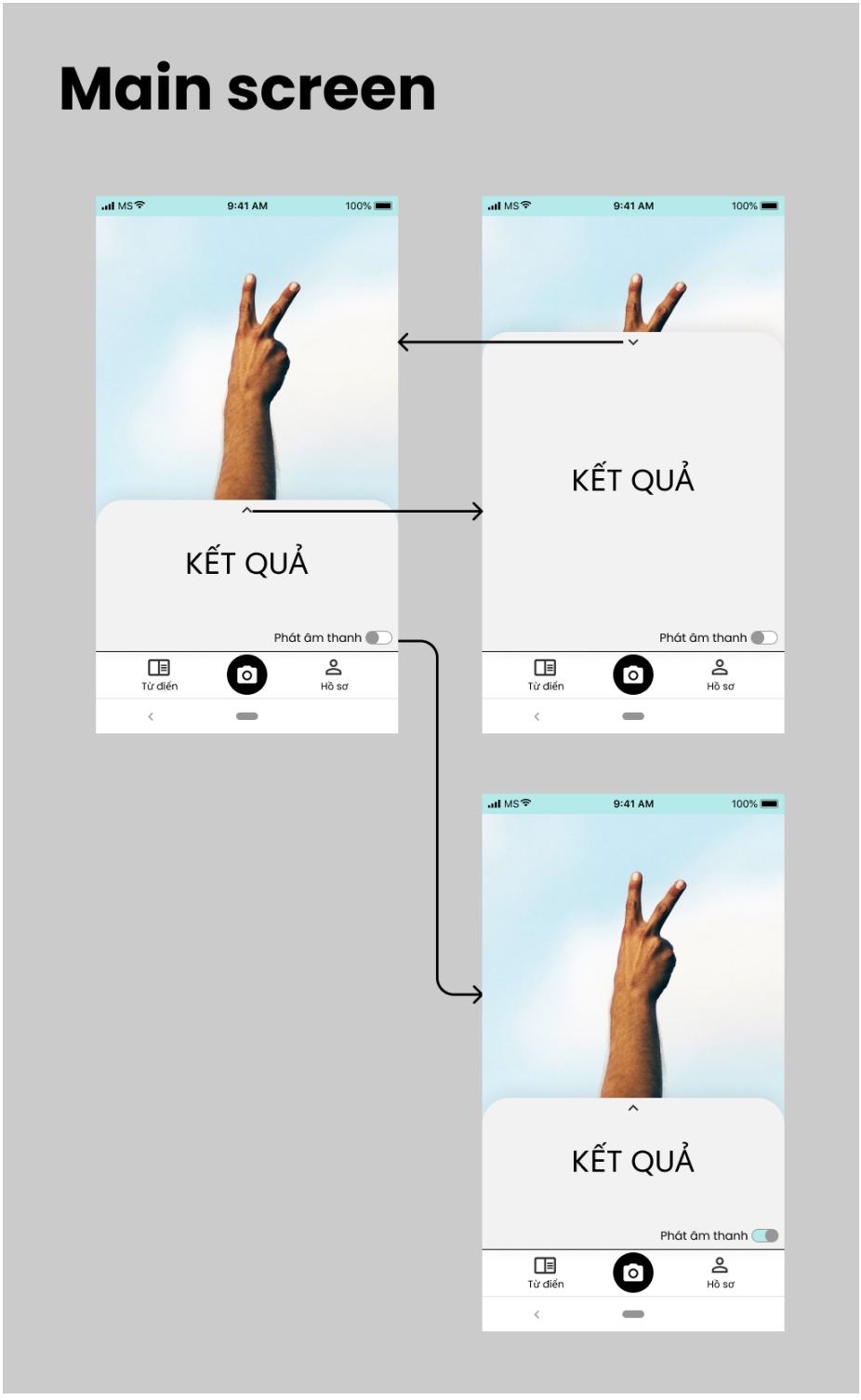


Figure 48: The main screen of the application

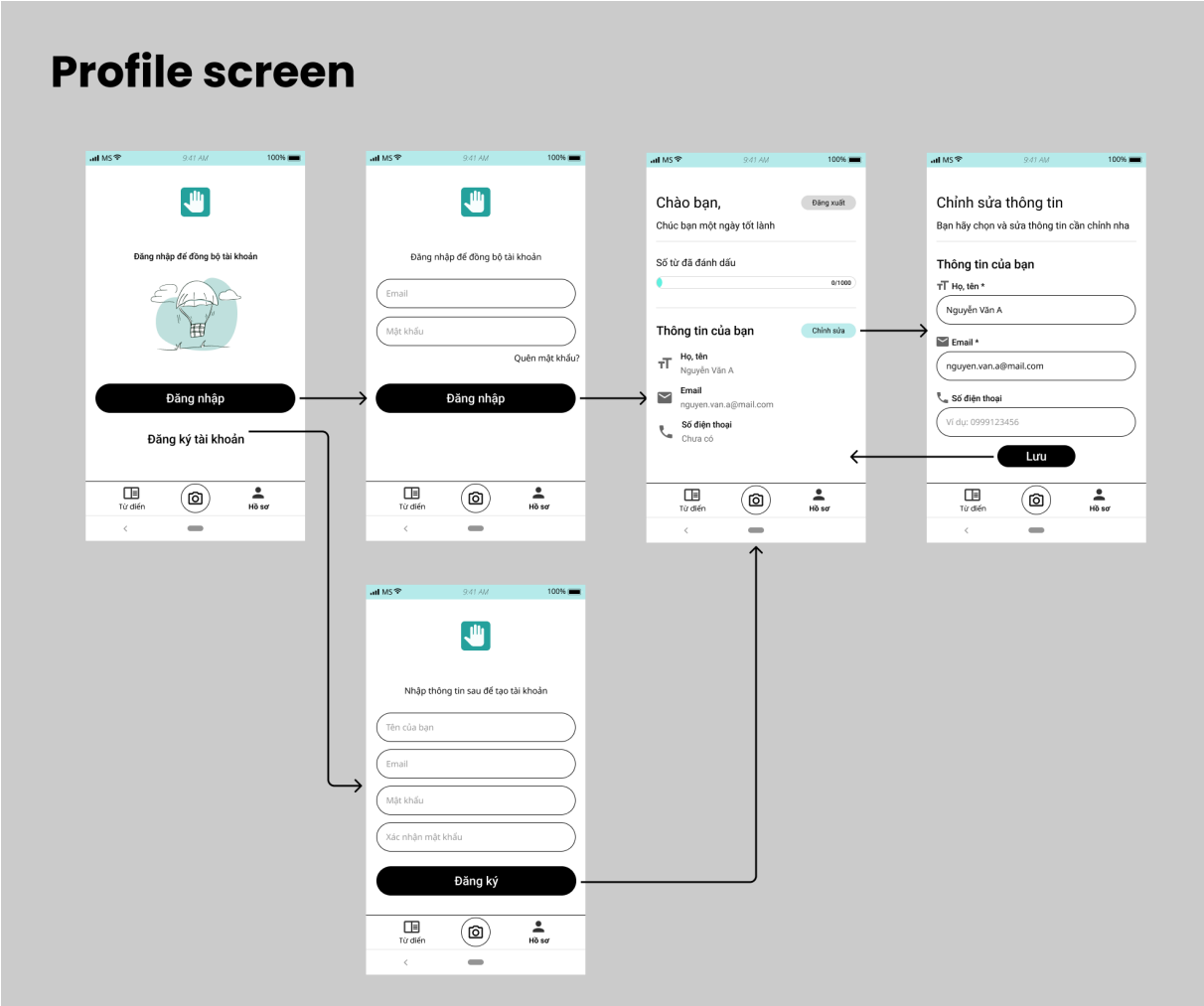


Figure 49: The profile screen

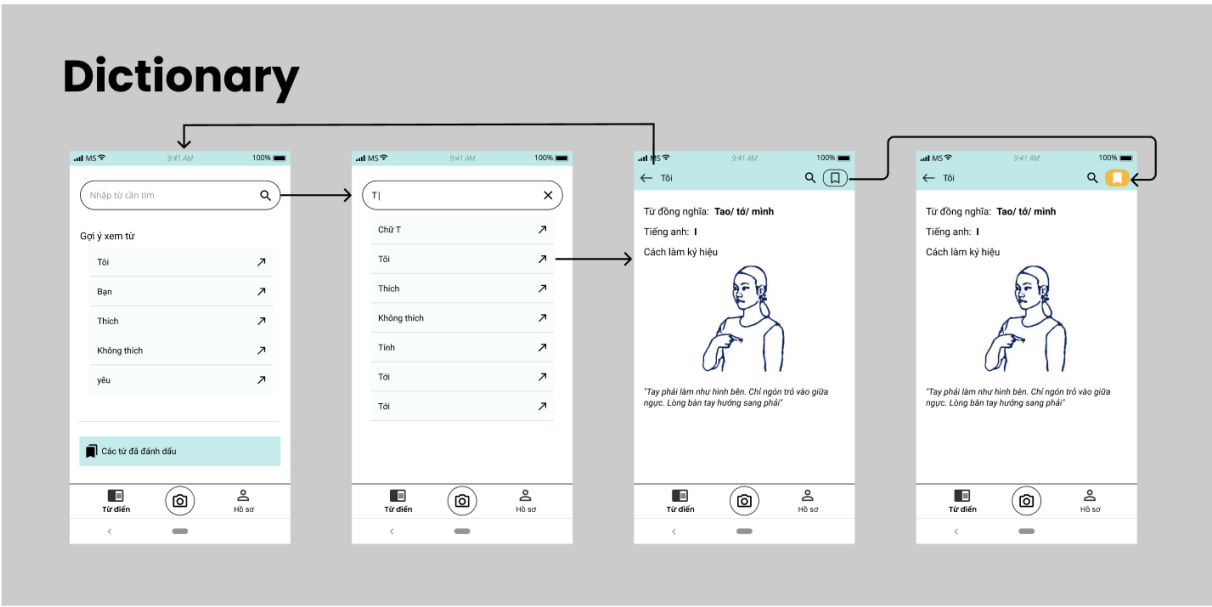


Figure 50: The dictionary screen

4.3.8 Use case

On the whole, Figure 51 illustrates the project's overall use-case.

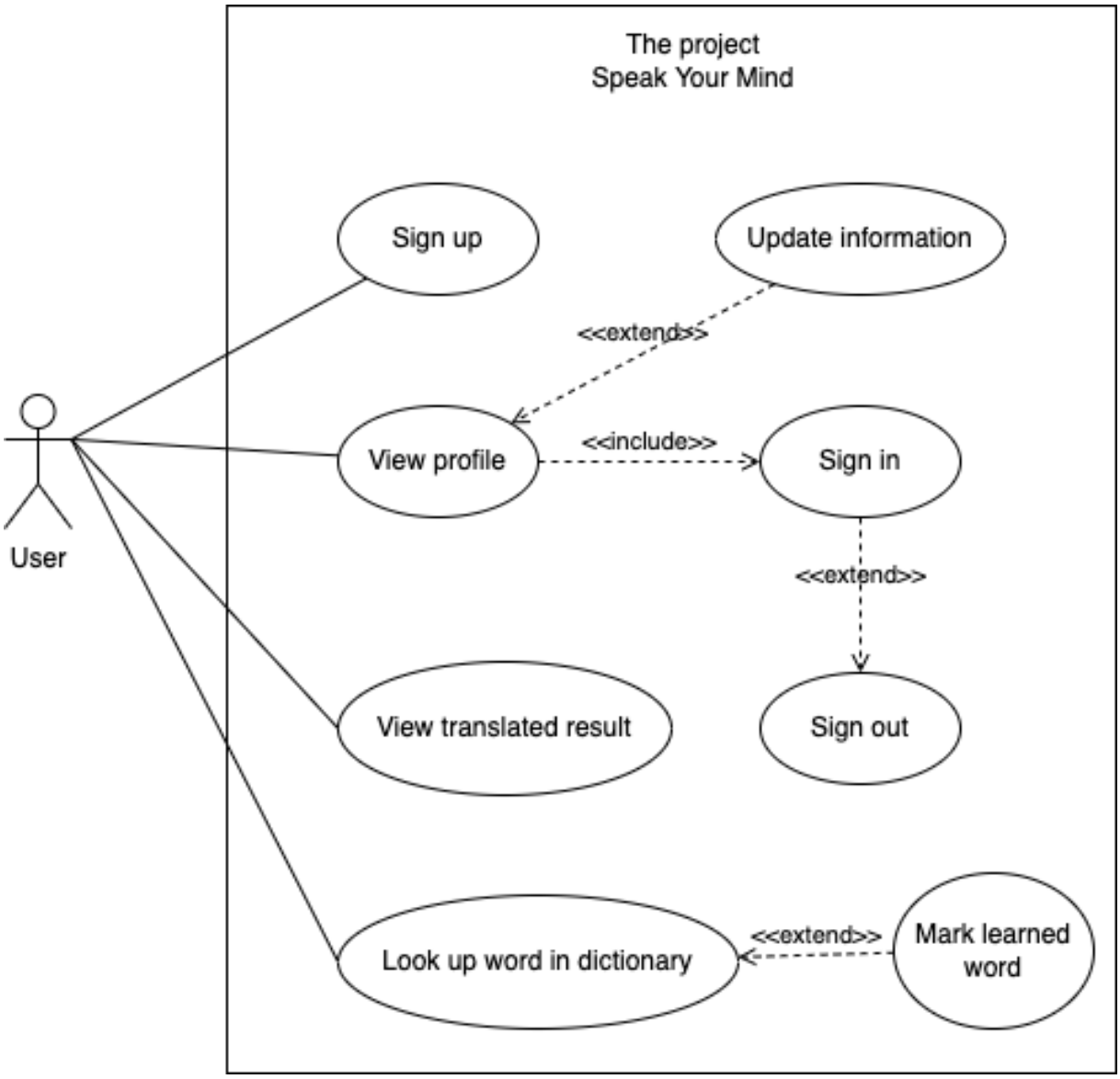


Figure 51: Use case diagram

4.3.9 Use case description

View translated result

Use case ID	1
Use case name	View translated result
Description	The user is informed about the hand gesture he just made
Actor	User
Post-condition(s)	Successful sign language detection and return to the user
Normal flow	1. The user visits the main page 2. The user signs the word 3. Application that records actions and makes predictions 4. The application displays the result after translated successfully
Exception flow	Exception1: 4a. The application cannot translate the sign language 5a. The application shows no more information and ends

Table 2: Use case view translated result

View profile

Use case ID	2
Use case name	View profile
Description	The user can view his personal information and edit the number of learned words and the number of new words learned in the day
Actor	User
Post-condition(s)	The user sees his information page
Normal flow	1. User visits the profile page 2. The application connects to the system in order to obtain information about the user 3. The application displays user data
Alternative flow	Null
Exception flow	Null

Table 3: Use case view profile

Sign up

Use case ID	3
Use case name	Sign up
Description	The user must first create a personal account to access the system
Actor	User
Pre-condition(s)	The device must be connected to the internet
Post-condition(s)	Successful account registration
Normal flow	1. The user taps on “Đăng ký tài khoản” button 2. The application displays the account registration view 3. The user enters the necessary data 4. The user taps the "Đăng ký" button 5. The application redirect the user to the login screen
Alternative flow	A. The user abandons the account creation process and wishes to return to the login screen 3.1 The user taps the "Đăng nhập” button and the application jumps to step 5
Exception flow	If there is no internet connection, the application displays the message "Không có kết nối mạng, vui lòng thử lại sau."

Table 4: Use case sign up

Sign in

Use case ID	4
Use case name	Sign in
Description	Allow the user to sign in to his account in order to use the application's services
Actor	User
Pre-condition(s)	The device must be connected to the internet
Post-condition(s)	The user successfully logs in.
Normal flow	1. The application displays the login page 2. The user enters his email address and password in the appropriate fields. 3. The user taps the "Đăng nhập" button 4. The application shows "Logged in successfully" 4. The application direct the user to the main screen
Alternative flow	A. The user enters wrong email address 4.1 The application shows "Email không tồn tại" 4.2 Application goes back to step 2 B. The user enters invalid email address 4.1 The application shows "Email không đúng định dang" 4.2 Application goes back to step 2 C. The user enters wrong password 4.1 The application shows "Mật khẩu không đúng" 4.2 Application goes back to step 2
Exception flow	If there is no internet connection, the application displays the message "Không có kết nối mạng, vui lòng thử lại sau."

Table 5: Use case sign in

Sign out

Use case ID	5
Use case name	Sign out
Description	The user wants to sign out the current account
Actor	User
Pre-condition(s)	The device must be connected to the internet The user logged in successfully
Post-condition(s)	The user signs out successfully
Normal flow	1. The user taps the "Đăng xuất" button 2. The application returns to the login screen
Alternative flow	No
Exception flow	If there is no internet connection, the application displays the message "Không có kết nối mạng, vui lòng thử lại sau."

Table 6: Use case sign out

Look up word in dictionary

Use case ID	6
Use case name	Look up word in dictionary
Description	The user needs to look up a word in the sign language dictionary
Actor	User
Pre-condition(s)	For the better result, the device should have internet connection
Post-condition(s)	The user successfully finds out how to sign the word
Normal flow	1. The user taps dictionary in the navbar 2. The application directs the user to screen dictionary 3. The user inputs the word and taps it on screen 4. The application displays the word's corresponding information and includes an instructional video.
Alternative flow	No
Exception flow	If there is no internet connection, the application only shows the local information, and the video is note included.

Table 7: Use case look up word in dictionary

Mark learned word

Use case ID	7
Use case name	Mark learned word
Description	The user wishes to mark a word as learned in order to easily find it in the future.
Actor	User
Pre-condition(s)	Previously logged in
Post-condition(s)	The user marked the word, which increased the learning bar on the profile page
Normal flow	1. The user taps dictionary in the navbar 2. The application directs the user to screen dictionary 3. The user inputs the word and taps it on screen 4. The application shows the corresponding information of the word, includes the instruction video 5. User taps on the bookmark button on the top right 6. That bookmark will change from outline state to filled state
Alternative flow	No
Exception flow	If the user hasn't logged in, the application will notify that "You have to login to perform this"

Table 8: Mark learned word

Chapter 5

Result and evaluation

5.1 Result

To remove barriers for people who are deaf or hard of hearing. The team developed an artificial intelligence application that recognizes gestures, manipulates words, and directly translates them into Vietnamese.

To watch the application's demo video, scan the QR code below or copy this url to your browser: `https://www.youtube.com/watch?v=Gn8mdojQnqY`

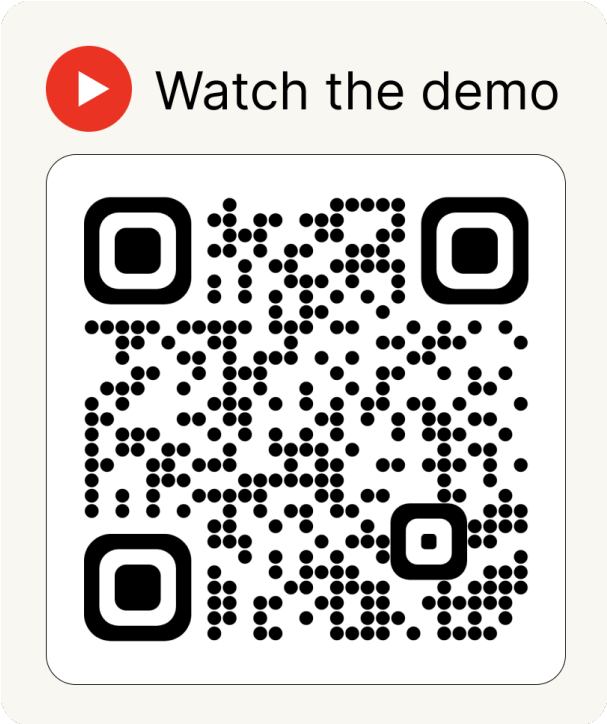


Figure 52: The QR Code of the demo video

Moreover, with the results obtained, the group has completed procedures to apply for intellectual property rights for the product and is awaiting approval.

Nevertheless, the system still has some drawbacks, such as long, complex terms and a large number of operations that the proposed system cannot recognize. The system still takes a long time to recognize a sign language, depending on its complexity.

5.2 Evaluation

Based on the actual results, the group has the following plans and orientations to make the system more complete:

- Convert the application to use react-native in the end to meet the needs of both the Android and IOS platforms.
- Improve the system and correct any errors.

Chapter 6

Summary

The thesis has applied image processing and artificial intelligence to research and develop a system capable of translating sign language into Vietnamese using only a camera module and a smartphone.

Aside from the system's two main features, the team is working on another feature that will help people learn sign language using the spaced repetition technique. There have been numerous studies and research papers published on this technique, and it is widely regarded as the most effective method for learning anything.

Because of the complexity of sign language and the diversity of ways people use it around the world, developing a sign language translating system has been a massive challenge for many scientists and engineers. Furthermore, when we were researching and developing the system, we discovered a few similar systems that only translated the sign language alphabet.

Additionally, in terms of human values, this system has the potential to address Vietnam's shortage of sign-language translators. It does, indeed, imply that people with disabilities will be able to live, work, and communicate in the same way as those who do not. They will receive a better education because the teacher will be able to understand and connect their thoughts more effectively. They can have better health because the health force will be able to learn more about how they are and what they are feeling, allowing us to better treat their problems. Their lives will be made easier as people around them can see and talk to them more clearly.

The deaf and deafeningly deafeningly deafeningly deafeningly deafeningly deafeningly deafeningly deafeningly deafeningly deafeningly deafeningly deafeningly deafeningly deafeningly deaf As a result, we believe that the deaf and mute deserve the opportunity to speak up, be heard, be seen, and be recognized. We can get to know each other and communicate fluently using this app, regardless of our level of sign language knowledge. Our bonds will eventually become more vital and profound, leading to a better world for the entire human race.

These human values emphasize the significance of this project in our world. Separated from the aforementioned, the aforementioned, and the aforementioned, and the aforementioned, and the aforementioned, and the aforementioned, and the We will have more resources to devote to completing our algorithm in the future, which will result in higher accuracy of the translation process and thorough completion of the thesis.

Bibliography

[1] Tim Bock. What is a distance matrix? https://www.displayr.com/what-is-a-distance-matrix/, 2021. (Accessed on 5/12/2021).

[2] Ketan Doshi. Foundations of nlp explained visually: Beam search, how it works. https://towardsdatascience.com/foundations-of-nlp-explained-visually-beam-search-how-it-works-1586b9849a24, 2021. (Accessed on 5/12/2021).

[3] Google. Text-to-speech. https://cloud.google.com/text-to-speech, 2021. (Accessed on 03/12/2021).

[4] Awni Hannun. Sequence modeling with ctc. *Distill*, 2017. https://distill.pub/2017/ctc.

[5] Nguyễn Yên Kiều Tuyết. Thiếu phiên dịch viên, bệnh nhân đặc biệt thiệt thòi về cơ hội chăm sóc y tế. https://vovgiaothong.vn/thieu-phien-dich-vien-benh-nhan-dac-biet-thiet-thoi-ve-co-hoi-cham-soc-y-te, 2021. (Accessed on 03/08/2021).

[6] Alina Kuznetsova, Laura Leal-Taixé, and Bodo Rosenhahn. Real-time sign language recognition using a consumer depth camera. In *Proceedings of the IEEE international conference on computer vision workshops*, pages 83–90, 2013.

[7] Sarfaraz Masood, Adhyan Srivastava, Harish Chandra Thuwal, and Musheer Ahmad. Real-time sign language gesture (word) recognition from video sequences using cnn and rnn. In *Intelligent Engineering Informatics*. Springer, 2018.

[8] MediaPipe. Mediapipe face mesh. https://google.github.io/mediapipe/solutions/face_mesh.html, 2021. (Accessed on 03/12/2021).

[9] Timothy F O'Connor, Matthew E Fach, Rachel Miller, Samuel E Root, Patrick P Mercier, and Darren J Lipomi. The language of glove: Wireless gesture decoder with low-power and stretchable hybrid electronics. *PloS one*, 12(7):e0179766, 2017.

[10] Harald Scheidl, Stefan Fiel, and Robert Sablatnig. Word beam search: A connectionist temporal classification decoding algorithm. In *2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, pages 253–258. IEEE, 2018.

[11] Central Deaf Services. Youtube channel: Cds. https://www.youtube.com/c/CDSDANANG, 2018.

[12] Hải Vân. Mở cánh cửa hy vọng cho người khiếm thính. https://nhandan.vn/baothoinay-xahoi/mo-canh-cua-hy-vong-cho-nguoi-khiem-thinh-313775/. (Accessed on 03/08/2021).

[13] Thomas Wood. What is the softmax function? https://deepai.org/machine-learning-glossary-and-terms/softmax-layer, 2021. (Accessed on 2/12/2021).

[14] Le Thi Thu Xuong. Youtube playlist: Dvd hỌc thỦ ngỮ - cÔ lÊ thỊ thu xƯƠng. https://youtube.com/playlist?list=PL32U0H_eykJ1f3g7S8pJ0YufNwzX9Eqt1, 2015.

[15] Fan Zhang, Valentin Bazarevsky, Andrey Vakunov, Andrei Tkachenka, George Sung, Chuo-Ling Chang, and Matthias Grundmann. Mediapipe hands: On-device real-time hand tracking. *arXiv preprint arXiv:2006.10214*, 2020.