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

———————— * ————————

GRADUATION THESIS PROPOSAL

USING MACHINE LEARNING METHODS IN TRANSLATING SIGN LANGUAGE INTO VIETNAMESE

Council: Software Engineering
Instructor: Assoc. Prof. Quan Thanh Tho

—o0o—

Student: Võ Tuấn Khanh (1810220)
Nguyễn Trí Nhân (1810390)

Ho Chi Minh City, December 2021

Declaration Of Authenticity

TODO: Viết sơ sơ về việc nội dung báo cáo không phải là false, ăn cắp này kia nọ
ví dụ:

Nhận diện hướng nhìn trong ảnh (Nhận diện vật thể trong ảnh) không phải là một đề tài mới nhưng vẫn là một thách thức bởi: trong các ứng dụng: việc nhận diện hướng nhìn của con người qua hình ảnh đòi hỏi kết quả chính xác cao, ở Việt Nam, hiện tại không thực sự có nhiều nghiên cứu chuyên sâu về đề tài. Trong quá trình nghiên cứu đề tài có rất nhiều kiến thức không nằm trong chương trình giảng dạy ở bậc Đại học tuy vậy chúng tôi xin cam đoan đây là công trình nghiên cứu của riêng tôi dưới sự hướng dẫn của tiến sĩ Nguyễn Đức Dũng. Nội dung nghiên cứu và các kết quả đều là trung thực và chưa từng được công bố trước đây. Các số liệu được sử dụng cho quá trình phân tích, nhận xét được chính tôi thu thập từ nhiều nguồn khác nhau và sẽ được ghi rõ trong phần tài liệu tham khảo.

Ngoài ra, tôi cũng có sử dụng một số nhận xét, đánh giá và số liệu của các tác giả khác, cơ quan tổ chức khác. Tất cả đều có trích dẫn và chú thích nguồn gốc.

Nếu phát hiện có bất kì sự gian lận nào, tôi xin hoàn toàn chịu trách nhiệm về nội dung luận văn của mình. Trường đại học Bách Khoa thành phố Hồ Chí Minh không liên quan đến những vi phạm tác quyền, bản quyền do tôi gây ra trong quá trình thực hiện.

Acknowledgment

TODO: Viết sau cùng -> về việc cảm ơn này kia

ví dụ:

Để hoàn thành kì đề cương luận văn này, tôi tỏ lòng biết ơn sâu sắc đến tiến sĩ Nguyễn Đức Dũng đã hướng dẫn tận tình trong suốt quá trình nghiên cứu.

Chúng tôi chân thành cám ơn quý thầy, cô trong khoa Khoa Học Và Kỹ Thuật Máy Tính, trường đại học Bách Khoa thành phố Hồ Chí Minh đã tận tình truyền đạt kiến thức trong những năm chúng tôi học tập ở trường. Với vốn kiến thức tích lũy được trong suốt quá trình học tập không chỉ là nền tảng cho quá trình nghiên cứu mà còn là hành trang để bước vào đời một cách tự tin.

Cuối cùng, tôi xin chúc quý thầy, cô dồi dào sức khỏe và thành công trong sự nghiệp cao quý.

Abstract

TODO: Viết sau cùng

ví dụ:

Nội dung chính của luận văn nhằm tìm hiểu, nghiên cứu xây dựng hệ thống nhận diện hướng nhìn thông qua ảnh chụp dựa trên những công trình, công nghệ mới được nghiên cứu và phát triển trong những năm gần đây của lĩnh vực Deep Learning. Trong quá trình nghiên cứu, tôi đã tiến hành tổng hợp, đánh giá ưu và nhược điểm của cách phương pháp, công nghệ đã và đang được nghiên cứu, sử dụng. Tiếp cận vấn đề theo nhiều hướng khác nhau, tôi thực hiện một số phương pháp sử dụng học sâu (CNN) để phát hiện hướng nhìn của con người qua hình ảnh. Bên cạnh việc hoàn thành nội dung của đề tài, nhóm chúng tôi đã nghiên cứu thêm một số phần để từ đó đặt nền móng cho các nghiên cứu sau này. Phần còn lại của luận văn tập trung vào việc đánh giá mô hình, kết quả đạt được, đồng thời phân tích ưu nhược điểm của mô hình thực hiện và thảo luận những vấn đề mà mô hình còn gặp phải. Cuối cùng, nhóm chúng tôi đề xuất hướng phát triển tiếp theo của đề tài trong tương lai.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Problem statement

TODO: Recheck problem statement

TLDR: It is hard for the deaf and mute to communicate with normal people. And there is not many ways for them to express their thought.

"Each deaf person is a separate world, and they feel more self-deprecating and alone when they do 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.

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 dependent 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.

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, there still exist inequalities in living standards and social participation for people with disabilities [6]. Many organizations are founded to support, help, and create better living conditions for people with disabilities to develop. However, this work still has many difficulties and inadequacies as there is no formal school or class. Moreover, there is no specific profession for this group of people, and the number of translators who know sign language is insufficient, 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 have an excellent 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 com-

munication, 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 resolve to problems of the deaf and hard of hearing.

1.2 Goals

TODO: Write Goals

TLDR: It is crucial to find out a way that help we connect more easily, the deaf and mute can convey their thoughts much comfortably.

Mục tiêu của đề tài là nghiên cứu, hiểu và hiện thực một số phương pháp học sâu để phát hiện hướng nhìn của con người qua hình ảnh.

Một số vấn đề đặt ra:

- Làm thế nào để giải quyết bài toán trên?
- Cách tiếp cận như thế nào?
- Những công nghệ nào đã và hiện đang được sử dụng?
- Hướng cải tiến?...

Như vậy để thực hiện theo đúng mục tiêu của đề tài cần xác định một số công việc phải giải quyết như sau:

- Tìm kiếm và thu thập dữ liệu phù hợp với nội dung đề tài.
- Tìm hiểu các phương pháp tiếp cận đã được hiện thực
- Lựa chọn mô hình phù hợp
- Lên kế hoạch hiện thực, phát triển hệ thống nhận diện huấn luyện và kiểm thử.

1.3 Scopes

TODO: Write Scopes

TLDR: In this case study, we will build a system including an app and camera module to translate at least 100 words from sign language into Vietnamese.

1.4 Thesis structure

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

Chapter	Content
1	A brief introduction about plan and objectives of thesis
2	Introduction of theoretical background as foundation knowledge that are applied in the project
3	Solution and design approach for problem statement of project
4	Summary of the thesis status and future plan

Chapter 2

Related Work

CheckList: [X] Sơ lược ý chính [X] Điều chỉnh [.] Translate [ ] Complete

Nowadays, research works related to the problem of converting sign language into text have been proposed by many researchers from all over the world, from many different approaches and perspectives. In which, two main approaches can be mentioned as follows: - Glove based approaches: With this approach, it requires deaf and mute people to wearing a sensor glove. When user has any different action or gesture, these sensor will be recorded. After that, data from sensor will analyze by analyzer component and return the output for user. - Vision based approaches: With this approach, image processing algorithms will be applied to be able to determine hand position, gestures and movements of the hand. The user will not have to wear necessary equipment like glove based approaches, which is convenient for user. However, with using library or algorithms of image processing, we need to deal with worst quality output, which is greatly affected by this algorithms. With both approaches above, there is has some problems, that is, they can only recognize a very small number of words. These words are mostly words with different hand shapes that can be classified like that. However, in sign language, there will be many words that use the same hand shape but will differ in many characteristics, such as position and orientation. To our knowledge, there is currently no model that can handle the conversion of sign language flexibly and conveniently for the deaf-mute, helping them to communicate effectively. natural to the common man. Therefore, by applying appropriate technologies, the authors carry out this graduation thesis with the goal of breaking down the barriers between deaf-mute people and normal people, helping them to become self-sufficient. more confident in daily communication.

Ngày nay, các công trình nghiên cứu liên quan đến vấn đề chuyển đổi ngôn ngữ ký hiệu thành văn bản đã được nhiều nhà nghiên cứu từ khắp nơi trên thế giới đề xuất , theo nhiều hướng tiếp cận và góc nhìn khác nhau. Trong đó có thể kể đến 2 hướng tiếp cận chính như sau: - Hướng tiếp cận sử dụng găng tay cảm biến: Đây là hướng tiếp cận mà người sử dụng sẽ đeo 1 chiếc găng tay được trang bị các cảm biến chuyển động chuyên dùng. Khi người sử dụng có các hành động hay cử chỉ khác nhau sẽ được các cảm biến này ghi nhận, sau đó qua một bộ phân tích và sẽ trả về kết quả cho người dùng -Hướng tiếp cận sử dụng xử lý hình ảnh: Trong hướng tiếp cận này, các thuật toán về xử lý hình ảnh sẽ được áp dụng để có thể xác định được vị trí bàn tay, các cử chỉ, chuyển động của bản tay như thế nào . Người sử dụng sẽ không phải mang các trang bị cần thiết như hướng tiếp cận sử dụng găng tay, thuận tiện cho người sử dụng. Tuy nhiên, độ hiệu quả của các thuật toán xử lý ảnh hướng rất nhiều đến chất lượng đầu ra.

Với cả hai cách tiếp cận trên đều có một đặc điểm chung, đó là đều chỉ có thể nhận diện được một số lượng rất ít từ vựng. Các từ vựng này hầu hết là các từ có sự khác nhau về hình dạng bàn tay thì mới có thể phân loại được như thế. Tuy nhiên, trong ngôn ngữ ký hiệu , sẽ có rất nhiều

từ sử dụng chung một hình dạng bàn tay nhưng sẽ khác nhau về nhiều đặc điểm , ví dụ như vị trí và hướng. Theo hiểu biết của chúng em thì hiện nay vẫn chưa có một mô hình nào có thể xử lý được việc chuyển đổi ngôn ngữ ký hiệu một cách linh hoạt và thuận tiện cho người câm-điếc, giúp họ có thể giao tiếp được một cách tự nhiên với người bình thường. Chính vì thế, bằng cách vận dụng những công nghệ phù hợp, nhóm tác giả tiến hành thực hiện đề tài luận văn tốt nghiệp này hướng đến mục tiêu phá bỏ các rào cản giữa người câm-điếc và người bình thường, giúp họ tự tin hơn trong việc giao tiếp hằng ngày.

In this category requires signers to wear a sensor glove or a colored glove. The task will be simplified during segmentation process by wearing glove. The drawback of this approach is that the signer has to wear the sensor hardware along with the glove during the operation of the system.

+ Hướng tiếp cận sử dụng xử lý hình ảnh

Chapter 3

Theoretical Background

3.1 Convolution Neural Network - CNN

Convolution Neural Networks are a special class of Neural Networks. 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 consist 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 certain properties into the architecture. These then make the forward function more efficient to implement and vastly reduce the amount of parameters in the network. Some of the main uses of CNN can be mentioned as: image classification, object detection, semantic segmentation, face recognition, etc.

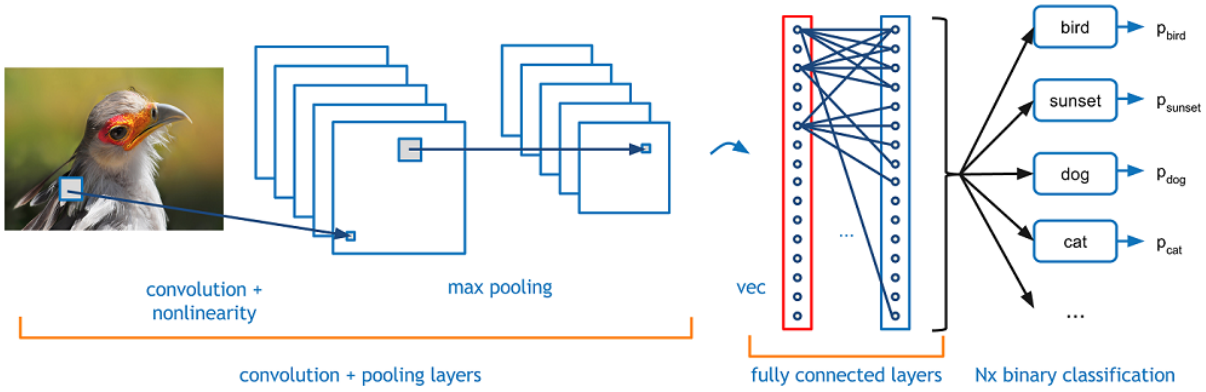


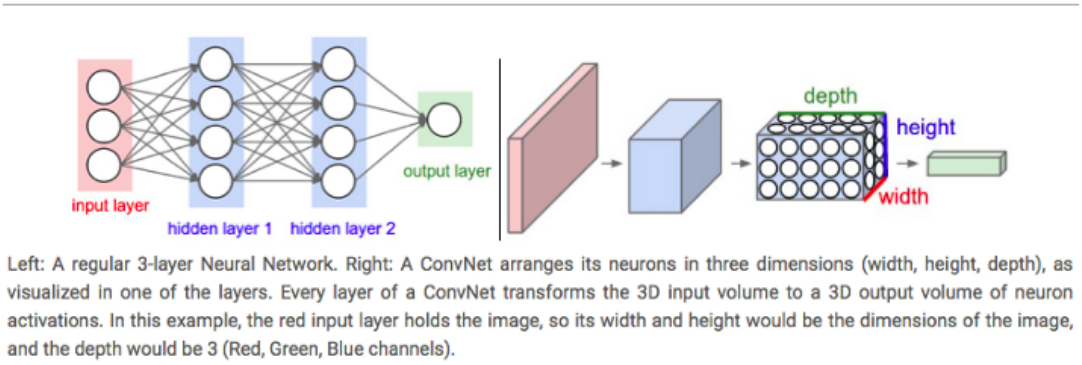
Figure 3.1: Convolution Neural Network

The figure 3.1 above shows an example of 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 than regular Neural Networks, we can see this different in figure 3.2 below. Regular Neural Networks transform an input by putting it through a series of hidden layers. Every layer is made up of a set of neurons, where

each layer is fully connected to all neurons in the layer before. Finally, there is a last fully-connected layer (the output layer) that represent the predictions. With CNN architecture. First of all, the layers are organized in 3 dimensions: width, height and depth. Further, the neurons in one layer do not connect to all the neurons in the next layer but only to a small region of it. Lastly, the final output will be reduced 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 3.2: Different between Normal Neural Network and Convolution Neural Network

As we can see in figure 3.3. CNN can be divided into two parts:

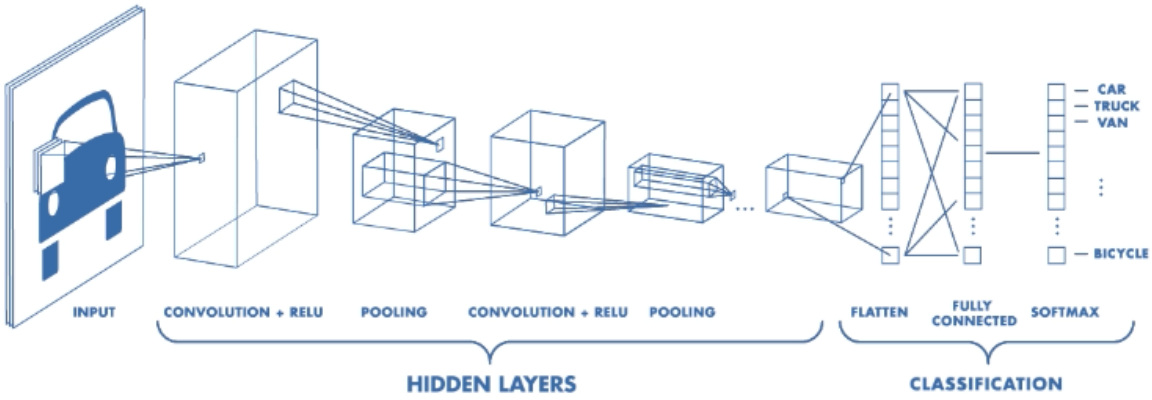


Figure 3.3: Convolution Neural Network Architecture

1. The hidden layers/ Feature extraction part
   - In this part, the network will perform a series of convolutions and pooling operations during which the features are detected. If you had a picture of a zebra, this is the part where the network would recognise its stripes, two ears, and four legs.
2. The Classification part
   - Here, the fully connected layers will serve as a classifier on top of these extracted features. They will assign a probability for the object on the image being what the algorithm predicts it is.

3.1.2 Feature extraction part

Convolutional Layer

Convolution Layer is the core building block of a Convolutional Network that does most the computational heavy lifting. A convolution is executed by sliding the filter over the input. At every location, a matrix multiplication is performed and sums the result onto the feature map. This process of extracting features from image happens throughout the CNN's convolutional layers. This process is illustrated in figure 3.4

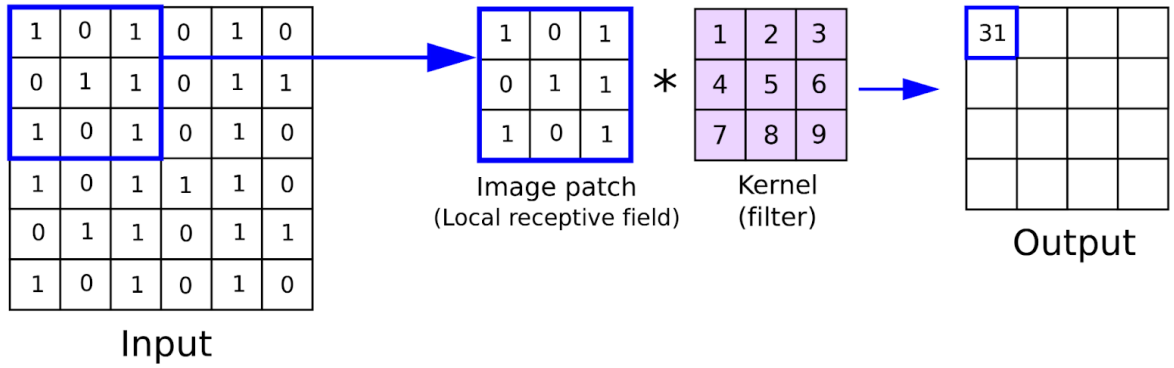


Figure 3.4: Convolution Neural Network Layer

When the feature map is made, we can pass each value in the feature map through a non-linearity function, such as ReLU, sigmoid, etc. 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 (3.5). 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 and makes sure the kernel and stride size will fit in the input.

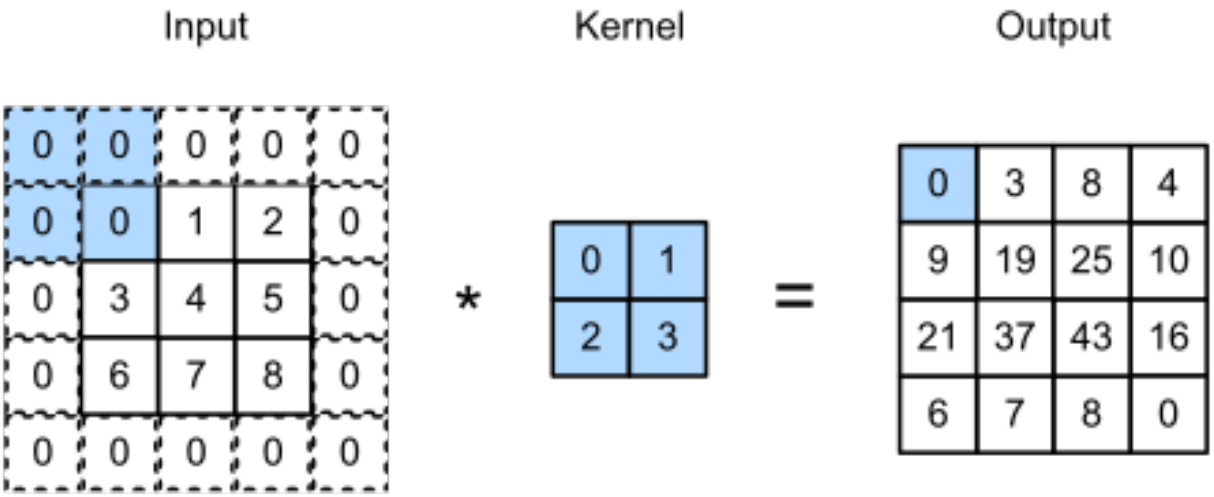


Figure 3.5: Using padding for strike one in Convolution Layer

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 shortens the training time and controls overfitting.

There are mainly two types of Pooling Layers in a CNN: Max Pooling and Average Pooling. The functionality of these two types of layers are demonstrated in figure 3.6 . Max Pooling restores the maximum value from the segment of the picture covered by the Kernel. Whereas, Average Pooling restores the average of the multitude of values from the bit of the picture covered by the Kernel.

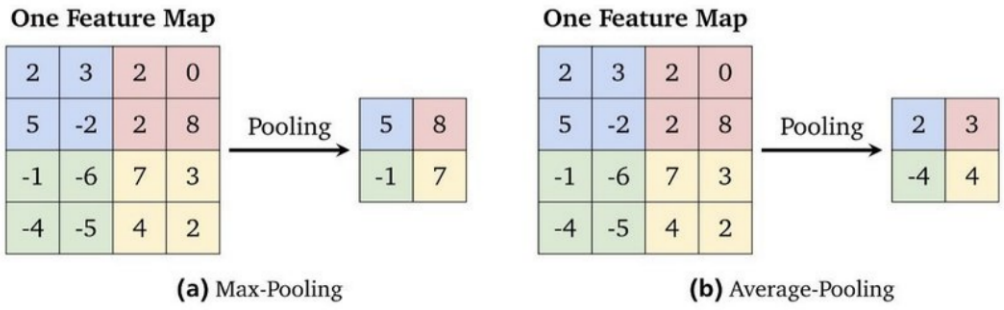


Figure 3.6: Max Pooling and Average Pooling

Activation Layers

Neural networks in general and CNNs in particular rely on a non-linear "trigger" function to signal distinct identification of likely features on each hidden layer. CNNs may use a variety of specific functions (figure 3.7), such as rectified linear units (ReLUs) and continuous trigger (non-linear) functions—to efficiently implement this non-linear triggering.

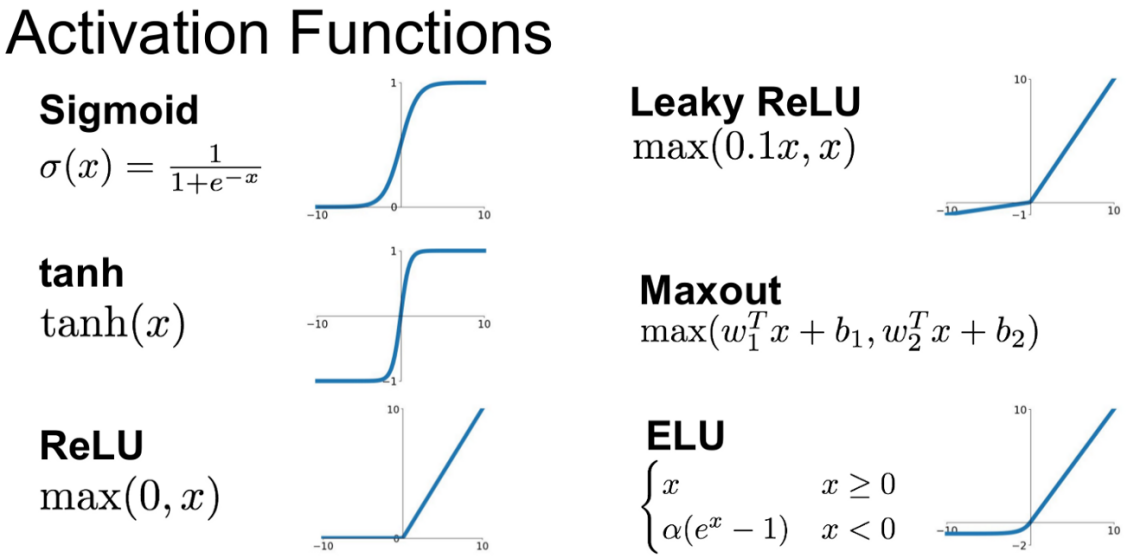


Figure 3.7: Some Active Function common used in CNN

3.1.3 Classification part

Fully connected layers

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

Figure 3.8 illustrates the way of input value stream into the fully connected layer. Because these fully connected layer can only accept one dimensional data. So, we need convert our 3D data to 1D data. After pass through some FC, we will get the result is the data classification.

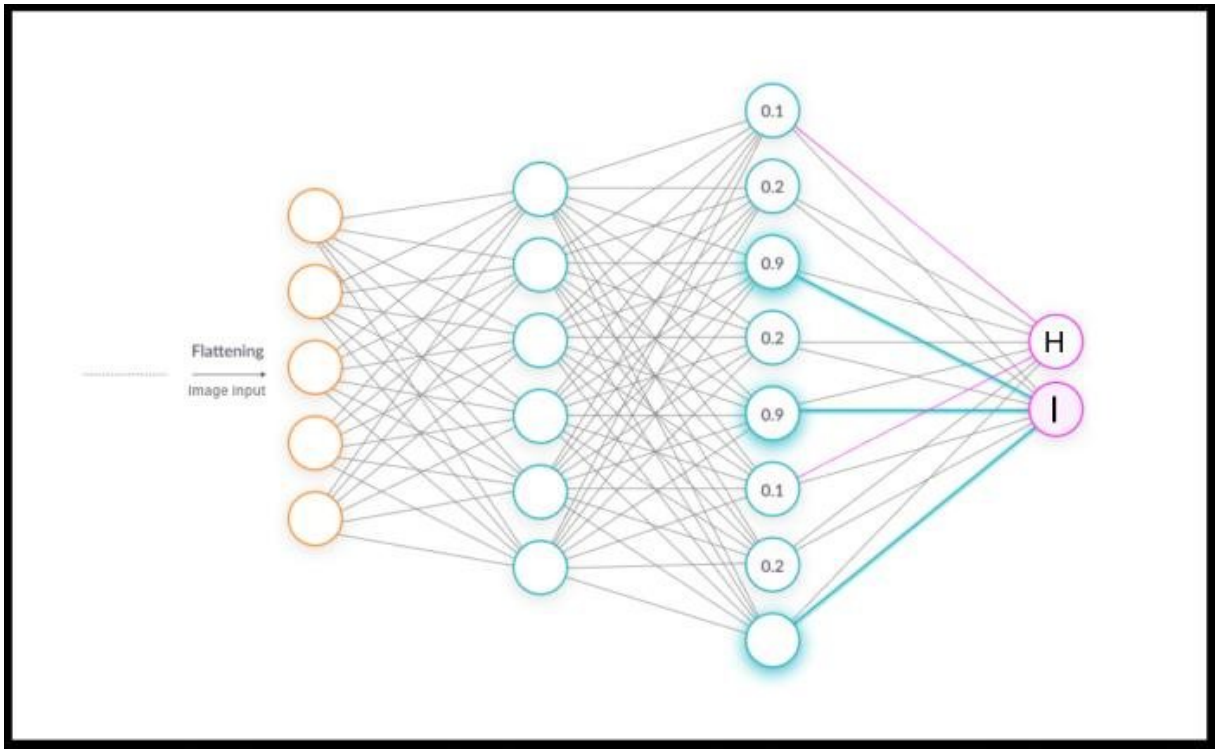


Figure 3.8: Fully connected Layer

3.2 Media Pipe

3.2.1 Introduction to Media Pipe Hands

MediaPipe Hands (3.9) is a high-resolution tracking system for hands and fingers. 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.

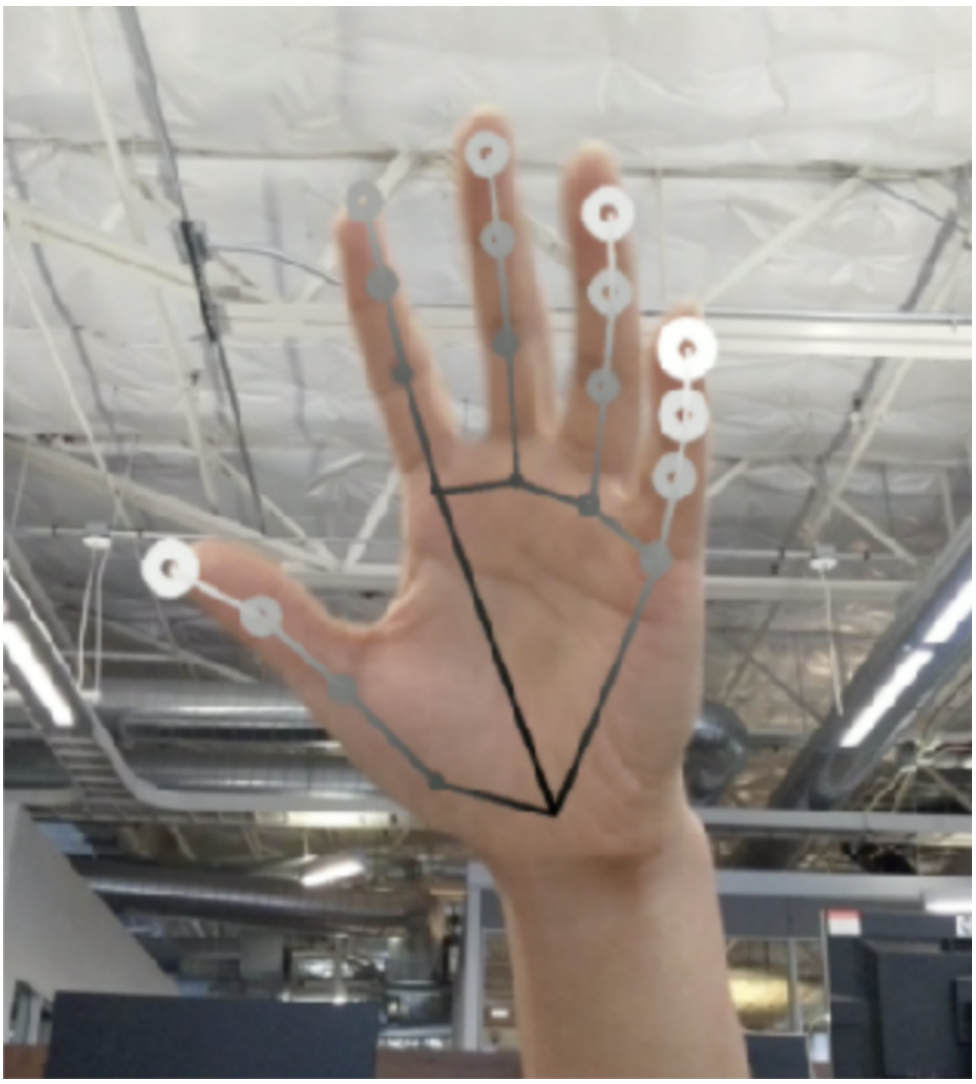


Figure 3.9: Media Pipe real time tracking 3D hand landmarks

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, 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 statement, the methods they use to address the above challenges vary in many strategies. First, instead of training a hand detector, they train a palm detector because estimating bounding boxes of inflexible objects like palms and fists is 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 3.10) 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.

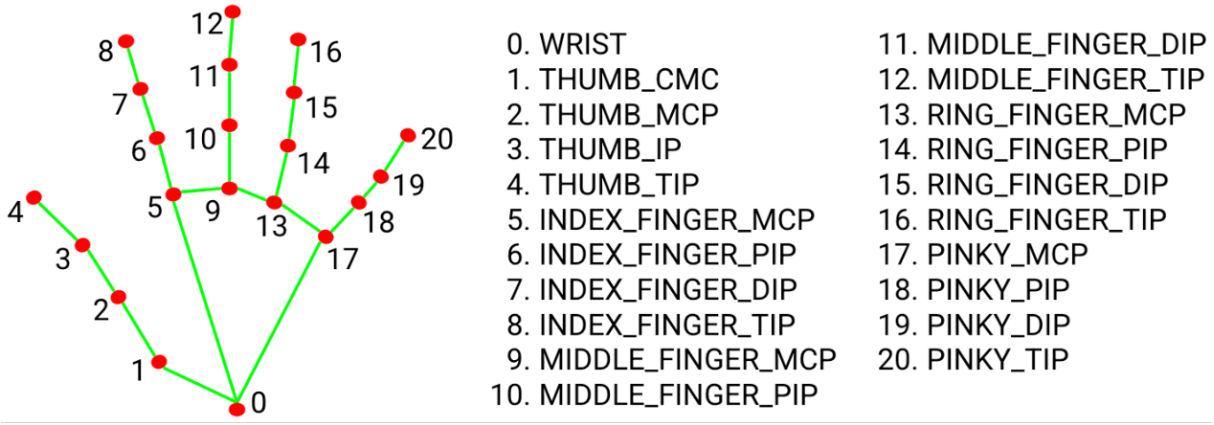


Figure 3.10: 21 Hand Landmarks

3.3 Distance Matrix

A distance matrix is a table that shows the distance between pairs of objects. For example, in the figure 3.11., we can see the distance of A and B is 16, B and C is 37 and so on. In the diagonal of table is the distance of 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 3.11: Distance Matrix

3.3.1 Create Distance Matrix

A distance matrix is computed from a raw data table. In the example below (3.12), we can use high school math (Pythagoras) to work out that distance between A and B.

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

Figure 3.12: Calculating distance between A and B

We can use same formula with more than two variables, and this is known as the Euclidean distance. In result, we have the distance matrix represented like figure 3.13

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 3.13: The Distance Matrix is constructed from Raw Data

3.4 Beam search and Connectionist Temporal Classification

CheckList: [x] BeamSearch [x] CTC recap [x] Combination [x] Pseudo code

3.4.1 Connectionist Temporal Classification

Connectionist Temporal Classification (CTC) is a type of Neural Network output helpful in tackling sequence problems like handwriting (figure 3.14) 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.

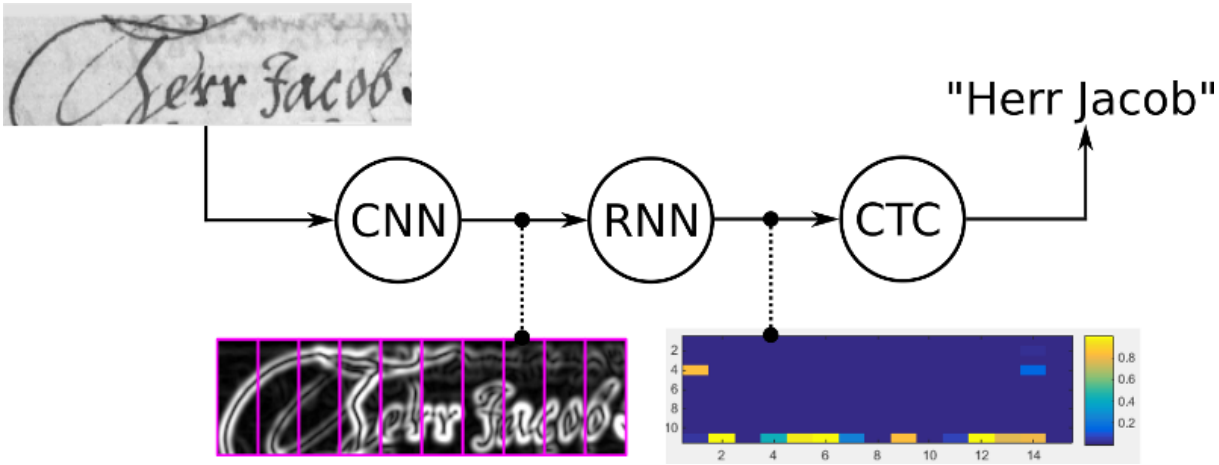


Figure 3.14: Overview of a Neural Network for handwriting recognition

3.4.2 Why we want to use CTC

In context of hand written recognition, we could create a data-set with images of text-lines, and then specify for each horizontal position of the image the corresponding character as shown in figure 3.15 Then, we could train a model to output a character-score for each horizontal position. However, there are two problem with this solution.

- It takes a lot of time, annotating dataset at the character level is a boring task.
- 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 Fig..... We have to remove all duplicate character like "t" and "o".

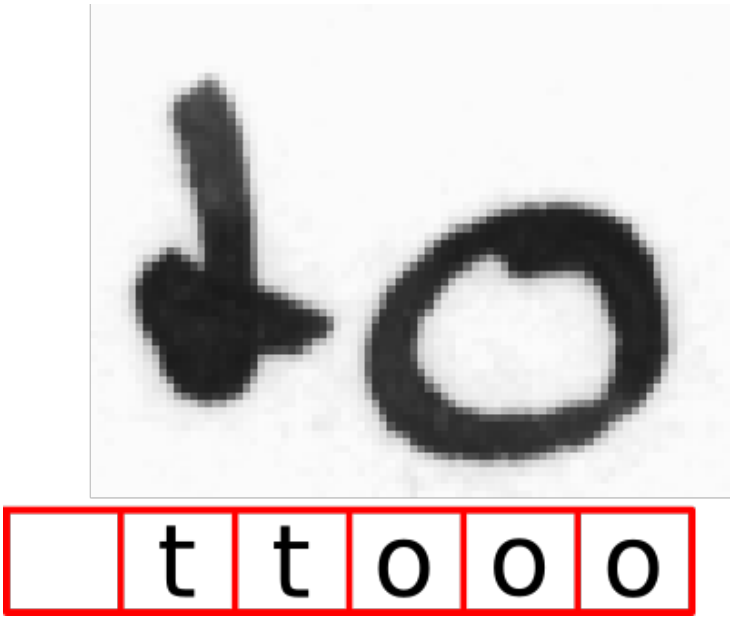


Figure 3.15: Annotation for each horizontal position of the image

CTC can solves both problem 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 image.
- Using decode techniques, we can directly get the result of the network and no further post-processing of the recongnized text is needed.

3.4.3 Beam Search with CTC decoder

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

Beam search

In computer science, beam search 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 which orders all partial solutions (states) according to some heuristic. But in beam search, only a predetermined number of best partial solutions are kept as candidates. Pseudo-code for basic version of beam-search is shown is figure 3.16

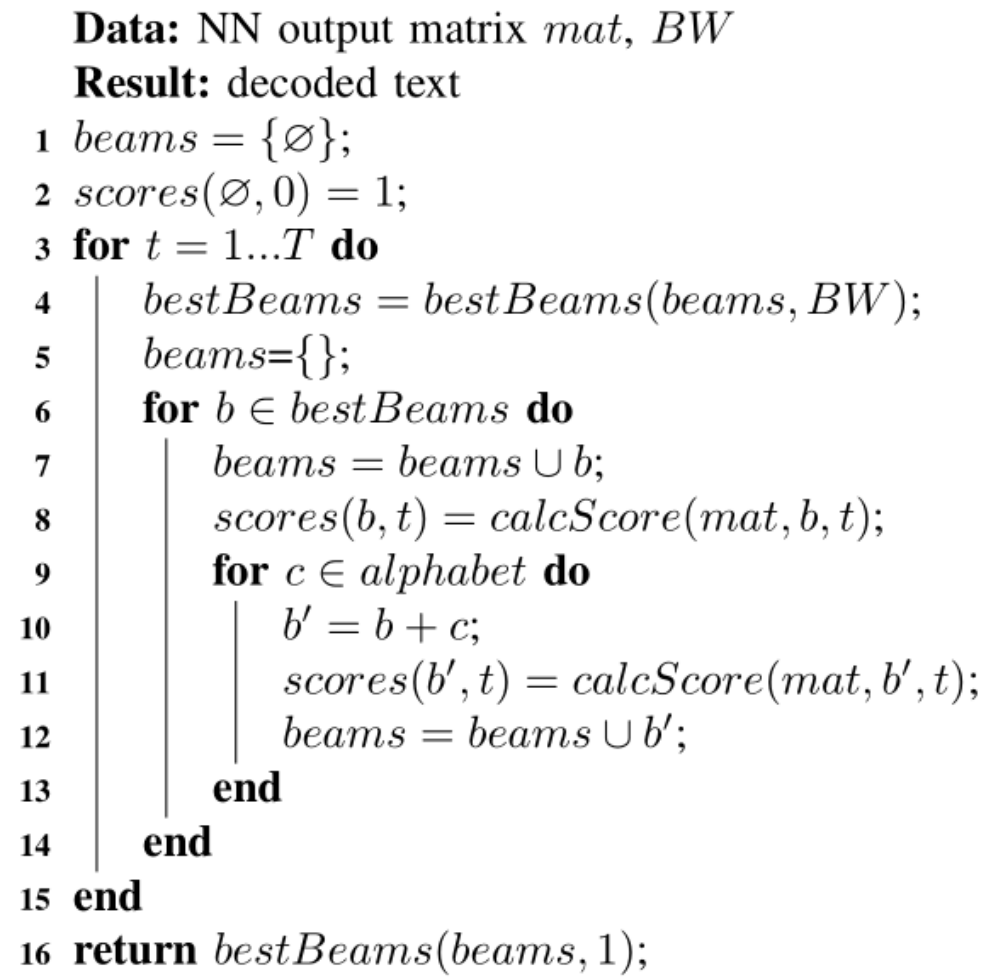


Figure 3.16: Basic version of Beam Search

Beam search algorithm will be implemented through the following steps, with two parameter will be included: output matrix and beam width (BW) which specifies the number of beams to keep . First of all, the list of beam and corresponding score is initialized (line 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). Each of beam, we calculate the score and get result (line 8), we will cover this step in more detail 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 is returned as a result (line 16).

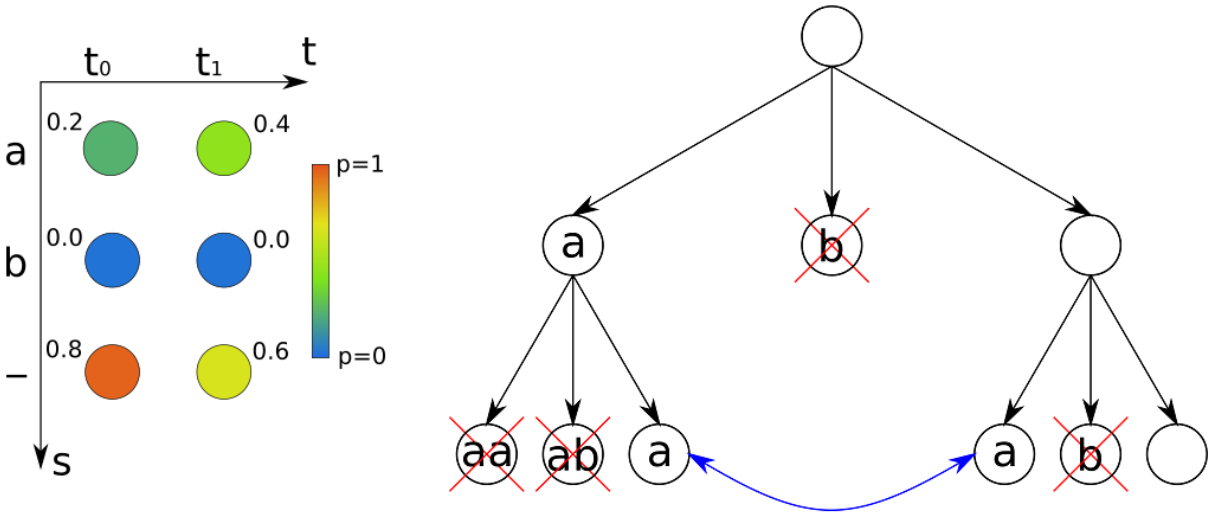


Figure 3.17: NN output and tree of beams with alphabet = “a”, “b” and BW = 2

As we can see, in figure 3.17, both output matrix to be decoded and the tree of beams are 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 we just discuss above, in this part, we will talk about how to scoring 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 Pb(b,t) and by Pnb(b,t) for the non-blank case.
- The probability Ptot(b,t) of a beam b at time-step t is simply the sum of Pb and Pnb, for example: Ptot(b,t) = Pb(b,t) + Pnb(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 3.18: The effect of appending a character to paths ending with blank and non-blank

In figure 3.18, we will see what happens when we extend a path. Three main case we can mention is:

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

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

- The unchanged (copied) beam ('a' -> 'a'):
  - To copy a beam, we can extend corresponding paths by a blank and get paths ending with a blank: Pb (n, t) += Ptot(b, t-1)*mat(blank, t)
  - Beside, with non-blank ending paths case, if we extend it by the last character (the beam is not empty): Pnb(b,t) += Pnb(b,t-1)*mat(b[-1],t) with -1 indexes the last character in the beam
- An extended beam ('a' -> '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: Pnb(b+c,t) += Ptot(b,t-1)*mat(c,t)
  - Or the last character of beam is repeated, we must ensure that the paths end with a blank: Pnb(b+c,t) += Pb(b,t-1)*mat(c,t)
  - We don't need t care about Pb(b+c,t) because we added a non-blank character

Putting it all together

Figure 3.19 depicts the CTC beam search algorithm. It is similar to the basic version previously displayed. However, it includes the code to score the beams: copied beams (lines 7-10) and extended beams(line 15-19). Finally, when we looking for the best scoring beams, the programs ranks them according to Ptot (line 4) and then take the BW best ones.

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 3.19: CTC beam search

Chapter 4

Design and Solution

4.1 System Structure

Overall, the whole system includes three parts of hardware modules, naming camera module, user's smartphone, and the server.

Our sign language translating AI system includes six main modules: hand pattern recognition, direction determination, location detection, action detection, word decoder, and text to speech (Figure 4). Firstly, the system continuously captures the hand's motion, processes it with the hand landmark model, and then puts it into those modules. Each of them has a unique role, and after combining the first four modules' results (hand pattern, direction, location, and action detection), the word decoder module will take the output data and bring out the corresponding result. Then, the result will show up on the main screen (Figure 16); meanwhile, the phone will speak out that word. In the below sections, we will discuss each module's role and how it works.

TODO: Replace with new structure

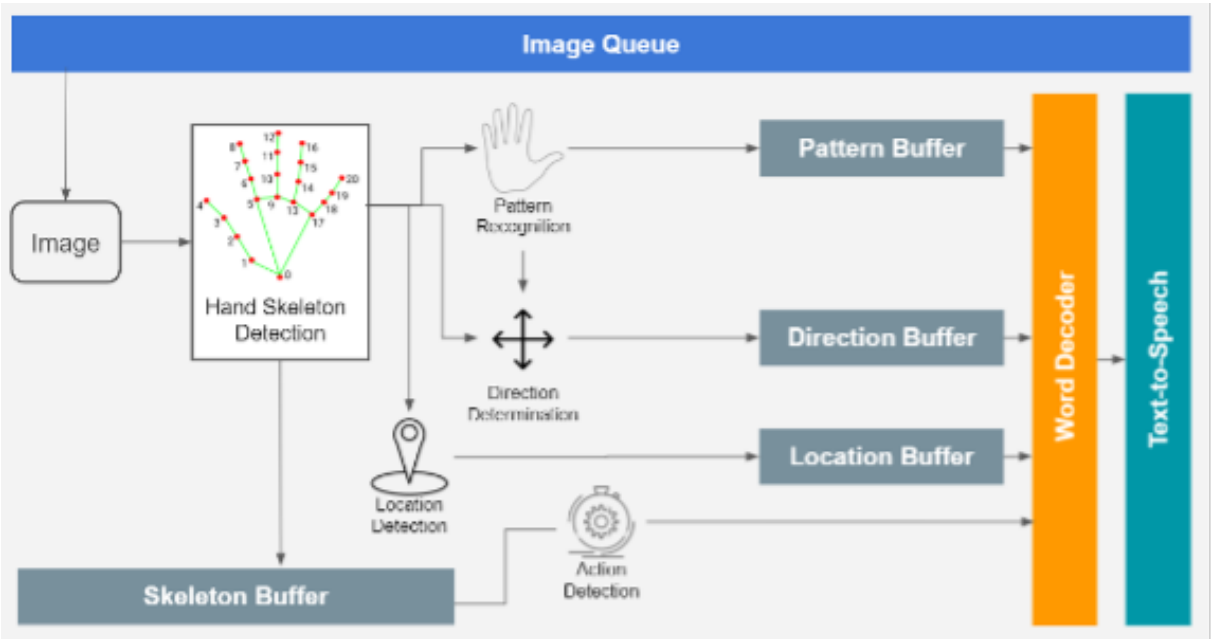


Figure 4.1: Overview of the system modules

4.2 Detail Implementation

4.2.1 Hand pattern recognition

Hand pattern recognition is the first and basic module of this system. While a person with disabilities does signs of sign language, his hands perform a series of different movements, where their hand may be spread out, clenched, or his fingers pointing out at something. Therefore, the role of this module is to recognize the pattern of the hands. Then combining the outcome with other modules, the system can give out the final result.

This module uses the output of the hand landmark model, which is a matrix size of 21. After calculating all the values in that matrix, we get a new matrix representing the distance between those 21 coordinates. Using the distance matrix as the input of CNN [2] with the designed structure (see Figure 4.2), as seen in Figure 4.1, will tell us the pattern of the hand at the moment it is captured.

Figure 4.2: Structure of convolutional neural network

4.2.2 Direction determination

The directions of the hand include four directions, i.e., right, left, up, down, front, and back. Each hand's pattern combined with different directions leads to a different meaning. For example, the pattern that points at someone means the word "you"; on the other hand, when we point it to ourselves, it means the word I (see Figure 4.3 and Figure 4.4).

Figure 4.3: Word "You" (bạn) in sign language

Figure 4.4: Word “I” (tôi) in sign language

To determine the hand’s direction, we use the hand landmark model provided in MediaPipe (see section 2 TK). The inception here is that we calculate the distance between the tip of the index finger and the wrist (called TK), then project it to the axis Ox, Oy, Oz, respectively. After that, we take each of those coordinates and compare them with the others. Finally, the one with the immense value will tell which axis the hand is on; besides, with the direction from the wrist to the tip of the index finger projected on that corresponding axis, we will know which direction the hand is.

For instance, a hand is known to be pointing toward the left direction. The value of the distance, when projected on the axis Ox, will be the biggest one among the three projected values. Then, calculate the vector drawn from the wrist to the tip of the index finger; we will know the direction of the hand itself.

4.2.3 Location detection

Locations of hand vary, is the hand put at forehead, mouth or the chest level, and so on. Every hand pattern that goes with every location will result in different words. Nevertheless, it is hard for the AI to know the hand’s location with only one camera, and its view is from above (see Figure 4.5). However, we came up with some solutions to this issue.

Firstly, we will take pictures of the hand and calculate the size of the hand in every frame in order to know whether that hand is getting bigger or smaller. Hence, if that hand is smaller than before, it means the hand is getting far away from the camera, and its location is somewhere at the chest level or the stomach level.

Figure 4.5: View from the camera module

Nonetheless, the above solution still has an issue: every man's hand has a different size, and the system does not know the correct position of the hand. Therefore, another solution is to use a wide-angle camera and set it away from the forehead. With this solution, the camera can have a much broader view. However, since we only have a normal-angle camera, we could not try out this solution and confirm its suitability.

4.2.4 Word decoder

TODO: Finish Word decoder

4.2.5 Text to speech

Besides displaying the translated sign language in text form, we included a text-to-speech module to know the result without looking into the screen. This module makes use of a free API provided by Google, named Text-to-Speech [4]. It converts arbitrary strings, words, and sentences into the sound of a person speaking the same things.