

BỘ GIÁO DỤC VÀ ĐÀO TẠO  
TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HCM  
KHOA CÔNG NGHỆ THÔNG TIN



## KHOÁ LUẬN TỐT NGHIỆP

**Đề tài: NHẬN DẠNG THỦ NGỮ**

GIÁO VIÊN HƯỚNG DẪN: TIẾN SĨ TRẦN NHẬT QUANG  
TRỊNH TẤN ĐẠT: 19133001

KHÓA 2019

BỘ GIÁO DỤC VÀ ĐÀO TẠO  
TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HCM  
KHOA CÔNG NGHỆ THÔNG TIN



## **KHOÁ LUẬN TỐT NGHIỆP**

**Đề tài: NHẬN DẠNG THỦ NGŨ**

GIAO VIÊN HƯỚNG DẪN: TIẾN SĨ TRẦN NHẬT QUANG  
TRỊNH TẤN ĐẠT: 19133001

**KHÓA 2019**



\*\*\*\*\*

\*\*\*\*\*

## PHIẾU NHẬN XÉT CỦA GIÁO VIÊN PHẢN BIỆN

Họ và tên Sinh viên: **Trịnh Tấn Đạt**

MSSV: 19133001

**Ngành: Kỹ Thuật Dữ Liệu**

Tên đề tài: **Nhân dạng thủ ngữ**

Họ và tên Giáo viên phản biện: **Nguyễn Thành Sơn**

Học vi: **Tiến sĩ**

**NHÂN XÉT:**

### 1. Nội dung đề tài và mức độ thực hiện đề tài:

.....

.....

## 2. Ưu điểm:

.....

.....

### 3. Khuyết điểm:

.....

4. Đề nghị cho bảo vệ hay không? .....

5. Đánh giá loại: .....

4. Điểm: .....

TP HCM, ngày .... tháng .... năm 2023

### **Giáo viên phản biện**

**TS. Nguyễn Thành Sơn**

## **LỜI CAM ĐOAN**

Luận văn này là công trình nghiên cứu của riêng tác giả, được thực hiện dưới sự hướng dẫn khoa học của TS. Trần Nhật Quang. Các số liệu, những kết luận nghiên cứu được trình bày trong luận văn này hoàn toàn trung thực. Kết quả nghiên cứu này chưa từng được công bố trong bất kỳ công trình nào khác.

Tôi xin hoàn toàn chịu trách nhiệm về lời cam đoan này.

**Sinh viên**

**Trịnh Tấn Đạt**

## LỜI CẢM ƠN

Lời đầu tiên, em xin chân thành cảm ơn Ban Giám hiệu trường Đại Học Sư Phạm Kỹ Thuật Thành phố Hồ Chí Minh, Quý Thầy Cô Giảng viên Khoa Công nghệ Thông tin đã quan tâm và chia sẻ những kiến thức quý báu, không chỉ hỗ trợ cho đề tài này mà còn có tính ứng dụng cao trong tương lai và sự nghiệp của các thế hệ sinh viên.

Đặc biệt, em xin bày tỏ lòng biết ơn chân thành và sâu sắc đến Thầy TS. Trần Nhật Quang – giảng viên hướng dẫn chuyên môn trong suốt quá trình thực hiện luận văn này. Không những kịp thời đưa ra các góp ý chuyên môn mang tính định hướng cụ thể, Thầy luôn sẵn sàng giải đáp mọi thắc mắc của nhóm và chia sẻ những kinh nghiệm quý giá đầy hữu ích, đã giúp em hoàn thiện và bảo vệ thành công đề tài luận văn tốt nghiệp.

Ngoài ra, em xin chân thành cảm ơn Quý Thầy Cô tại Phòng thí nghiệm Hệ thống Thông minh (ISLab) thuộc Khoa Điện - Điện tử đã nhiệt tình hỗ trợ về cơ sở vật chất, cũng như tạo điều kiện thuận lợi để em có thể thực nghiệm kết quả luận văn.

Cuối cùng, em muốn cảm ơn gia đình, người thân và bạn bè đã luôn đồng hành, hỗ trợ về mọi mặt, giúp em hoàn thành đồ án tốt nghiệp trong điều kiện tốt nhất.

Trong suốt quá trình thực hiện đồ án, em luôn nỗ lực hết mình để đạt được kết quả tốt nhất ở các mục tiêu đã đặt ra. Tuy nhiên, do còn một số hạn chế về kiến thức và thời gian nên sẽ không tránh khỏi những sai sót không mong muốn. Em rất mong nhận được sự thông cảm và góp ý xây dựng từ Quý Thầy Cô và các bạn để em có thể rút kinh nghiệm và cải thiện luận văn này trong tương lai tới.

Em xin chân thành cảm ơn!

# ĐỀ CƯƠNG LUẬN VĂN TỐT NGHIỆP

Họ và tên sinh viên: **Trịnh Tấn Đạt**

MSSV: **19133001**

Thời gian làm luận văn từ: Tuần 1

Đến: Tuần 15

Ngành: **Kỹ Thuật Dữ Liệu**

Tên đề tài: **Nhận dạng thủ ngữ**

GV hướng dẫn: **TS. Trần Nhật Quang**

Nhiệm vụ của luận văn:

1. Nghiên cứu tổng quan về tình trạng trẻ em gặp khó khăn về thính lực
2. Nghiên cứu về các ngôn ngữ ký hiệu tại Việt Nam và thế giới.
3. Tìm hiểu về cấu trúc của mạng Deep Learning và RNN
4. Thu thập, xây dựng dữ liệu về các ký hiệu thủ ngữ
5. Huấn luyện và đánh giá mô hình.

Đề cương viết luận văn có bố cục như sau:

## PHẦN MỞ ĐẦU

1. Lý do chọn đề tài
2. Ý nghĩa khoa học và thực tiễn
3. Mục tiêu và nhiệm vụ của nghiên cứu
4. Mô tả bài toán
5. Kết quả dự kiến đạt được

## PHẦN NỘI DUNG

Chương 1: Cơ sở lý thuyết

- 1.1. Neural network
- 1.2. Deep learning
- 1.3. Tổng quan về xử lý ảnh
- 1.4. Thuật toán tối ưu

Chương 2: Áp dụng vào nhận diện hình ảnh bằng cử chỉ

- 2.1. Giới thiệu về tensorflow
- 2.2. Thu thập dữ liệu về các ký hiệu thủ ngữ Tiếng Việt
- 2.3. Áp dụng cho bài toán nhận diện hình vẽ bằng cử chỉ
- 2.4. Ứng dụng nhận diện hình ảnh vẽ bằng cử chỉ

**PHẦN KẾT LUẬN**

- 1. Kết quả đạt được
- 2. Hạn chế
- 3. Hướng phát triển

**DANH MỤC TÀI LIỆU THAM KHẢO**

**KẾ HOẠCH THỰC HIỆN**

STT	Thời gian	Công việc	Ghi chú
1	Tuần 1	Xác định mục tiêu và phạm vi của đề tài	
2	Tuần 2	Tìm hiểu các khái niệm cơ bản	
3	Tuần 3	Tìm hiểu mạng RNN và các biến thể	
4	Tuần 4	Chuẩn bị dữ liệu	
5	Tuần 5	Xác định các mô hình cần phải làm	
6	Tuần 6	Thực hành xây dựng bộ tập dữ liệu trên MediaPipe	
7	Tuần 7	Thu thập dữ liệu sau khi thực hành thành công trên MediaPipe	
8	Tuần 8	Sắp xếp phân bố dữ liệu sao cho hợp lý	
9	Tuần 9	Xây dựng mô hình RNN, LSTM	
10	Tuần 10	Xây dựng mô hình Bidirectional LSTM và GRU	
11	Tuần 11	Xây dựng mô hình Hierarchical Model	
12	Tuần 12	Tổng hợp các thông số để so sánh	
13	Tuần 13	Viết báo cáo	
14	Tuần 14	Tinh chỉnh ứng dụng, sửa và viết báo cáo	
15	Tuần 15	Hoàn thành dự án và báo cáo	

Ý kiến của giáo viên hướng dẫn

TP HCM, ngày 29 tháng 12 năm 2023

Người viết đề cương



TS. Trần Nhật Quang

Trịnh Tấn Đạt



## MỤC LỤC

DANH MỤC HÌNH ẢNH.....	1
DANH MỤC BẢNG.....	3
DANH MỤC CÁC TỪ VIẾT TẮT .....	4
TÓM TẮT KHÓA LUẬN .....	5
PHẦN MỞ ĐẦU .....	6
1. LÝ DO CHỌN ĐỀ TÀI.....	6
2. Ý NGHĨA KHOA HỌC VÀ THỰC TIỄN .....	6
2.1. Ý nghĩa khoa học.....	6
2.2. Ý nghĩa thực tiễn.....	7
3. MỤC TIÊU VÀ NHIỆM VỤ CỦA NGHIÊN CỨU .....	7
4. MÔ TẢ BÀI TOÁN .....	8
5. KẾT QUẢ DỰ KIẾN ĐẠT ĐƯỢC .....	8
PHẦN NỘI DUNG.....	9
CHƯƠNG 1: CƠ SỞ LÝ THUYẾT.....	9
1.1. NEURAL NETWORK.....	9
1.1.1. Giới thiệu sơ lược về Neural Network.....	9
1.1.2. Mô hình tổng quát.....	12
1.2. DEEP LEARNING .....	13
1.2.1. Action Recognition.....	14
1.2.2. Mạng hồi quy RNN .....	15
1.2.2.1. Mạng hồi quy RNN là gì? .....	15
1.2.2.2. Phương thức hoạt động .....	15
1.2.2.3. Một số dạng RNN thường gặp .....	17
1.2.3. Mạng hồi quy LSTM và biến thể Bi-directional LSTM .....	18
1.2.3.1. Mạng hồi quy LSTM là gì? .....	18
1.2.3.2. Biến thể Bidirectional LSTM .....	21
1.2.4. Mô hình phân cấp Hierarchical Model .....	22
1.2.5. Mô hình GRU .....	23
1.3. THƯ VIỆN MEDIAPIPE HAND .....	24
1.4. THUẬT TOÁN TỐI ƯU .....	25
1.4.1. Thuật toán tối ưu là gì? .....	25
1.4.2. Giới thiệu thuật toán Adam.....	26

1.4.3. Nguyên tắc hoạt động của Adam .....	26
1.4.4. Công thức .....	27
1.4.5. Ưu và nhược điểm của thuật toán Adam.....	27
1.5. TỐI ƯU HOÁ MÔ HÌNH .....	27
1.6. CÁC PHƯƠNG PHÁP ĐÁNH GIÁ .....	28
<b>CHƯƠNG 2: ỨNG DỤNG VÀO NHẬN DIỆN THỮ NGỮ BẰNG CỬ CHỈ.....</b>	<b>30</b>
2.1. GIỚI THIỆU VỀ TENSORFLOW .....	30
2.2. THU THẬP DỮ LIỆU VÀ XÂY DỰNG MÔ HÌNH .....	30
2.2.1. Giới thiệu.....	30
2.2.2. Xây dựng mô hình.....	33
2.2.2.1. Mô hình RNN.....	33
2.2.2.2. Mô hình LSTM .....	33
2.2.2.3. Mô hình Bidirectional LSTM.....	33
2.2.2.4. Mô hình Hierarchical Model.....	34
2.2.2.5. Mô hình GRU.....	34
2.3. KẾT QUẢ MÔ HÌNH.....	34
2.3.1. Mô hình RNN .....	35
2.3.2. Mô hình LSTM .....	37
2.3.3. Mô hình Bidirectional LSTM .....	39
2.3.4. Mô hình Hierarchical Model.....	41
2.3.5. Mô hình GRU .....	43
2.4. LỰA CHỌN MÔ HÌNH.....	44
2.5. KẾT QUẢ THỰC TẾ .....	45
<b>PHẦN KẾT LUẬN .....</b>	<b>52</b>
1. KẾT QUẢ ĐẠT ĐƯỢC.....	52
2. HẠN CHẾ.....	52
3. HƯỚNG PHÁT TRIỂN .....	52
<b>DANH MỤC TÀI LIỆU THAM KHẢO.....</b>	<b>53</b>

## DANH MỤC HÌNH ẢNH

Hình 1: Cấu trúc mạng nơ-ron được lấy cảm hứng từ tế bào nơ-ron của não người.....	9
Hình 2: Các thành phần cơ bản của mạng nơ-ron nhân tạo .....	10
Hình 3: Cấu trúc đơn giản của mạng nơ-ron .....	12
Hình 4: Cấu trúc mạng nơ-ron tích chập .....	13
Hình 5: Cấu trúc cơ bản của mạng RNN.....	15
Hình 6: Mô hình RNN ba lớp.....	16
Hình 7: Cấu trúc cơ bản của LSTM .....	18
Hình 8: Reset gate.....	19
Hình 9: Cập nhật giá trị cho ô trạng thái .....	19
Hình 10: Ô trạng thái mới.....	20
Hình 11: Đầu ra .....	20
Hình 12: Mô hình Bidirectional LSTM .....	21
Hình 13: Cấu trúc của mô hình phân cấp Hierarchical Model.....	22
Hình 14: Kiến trúc mô hình GRU .....	23
Hình 15: Mô hình phát hiện lòng bàn tay.....	24
Hình 16: Hand Landmark Model. ....	25
Hình 17: Mô phỏng thuật toán Adam.....	26
Hình 18: Tensorflow .....	30
Hình 19: Cấu trúc bộ dữ liệu được thu thập .....	31
Hình 20: Cấu trúc của dữ liệu sau khi tổng hợp xong .....	32
Hình 21: Bảng kí hiệu ngôn ngữ Tiếng Việt. ....	32
Hình 22: Kiến trúc mô hình RNN .....	33
Hình 23: Kiến trúc mô hình LSTM .....	33
Hình 24: Kiến trúc mô hình Bidirectional LSTM .....	33
Hình 25: Kiến trúc Hierarchical Model .....	34
Hình 26: Kiến trúc mô hình GRU .....	34
Hình 27: Đồ thị biểu thị độ chính xác và độ sai số của mô hình RNN.....	35
Hình 28: Confusion Matrix của mô hình RNN .....	36
Hình 29: Đồ thị biểu thị độ chính xác và độ sai số của mô hình RNN.....	37
Hình 30: Confusion Matrix của mô hình LSTM .....	38
Hình 31: Đồ thị biểu thị độ chính xác và độ sai số của mô hình Bidirectional LSTM.....	39

<b>Hình 32: Confusion matrix của Bidirectional LSTM .....</b>	<b>40</b>
<b>Hình 33: Đồ thị biểu thị độ chính xác và độ sai số của Hierachical Model .....</b>	<b>41</b>
<b>Hình 34: Confusion Matrix của Hierachical Model .....</b>	<b>42</b>
<b>Hình 35: Đồ thị biểu thị độ chính xác và độ sai số của mô hình GRU .....</b>	<b>43</b>
<b>Hình 36: Confusion matrix của mô hình GRU .....</b>	<b>44</b>
<b>Hình 37: Nhận biết nothing .....</b>	<b>45</b>
<b>Hình 38: Nhận biết chữ a .....</b>	<b>45</b>
<b>Hình 39: Nhận biết chữ aw .....</b>	<b>46</b>
<b>Hình 40: Nhận biết chữ aa .....</b>	<b>46</b>
<b>Hình 41: Nhận biết chữ o .....</b>	<b>47</b>
<b>Hình 42: Nhận biết chữ oo .....</b>	<b>47</b>
<b>Hình 43: Nhận biết chữ u .....</b>	<b>48</b>
<b>Hình 44: Nhận biết chữ uw .....</b>	<b>48</b>
<b>Hình 45: Nhận biết chữ e .....</b>	<b>49</b>
<b>Hình 46: Nhận biết chữ ee .....</b>	<b>49</b>
<b>Hình 47: Nhận biết dấu sắc .....</b>	<b>50</b>
<b>Hình 48: Nhận biết dấu huyền .....</b>	<b>50</b>
<b>Hình 49: Nhận biết dấu ngã .....</b>	<b>51</b>
<b>Hình 50: Nhận biết dấu nặng .....</b>	<b>51</b>

## DANH MỤC BẢNG

<b>Bảng 1: Các hàm kích hoạt thường dùng .....</b>	<b>11</b>
<b>Bảng 2: Các loại RNN thường gặp.....</b>	<b>17</b>
<b>Bảng 3: Bảng thông số của mô hình RNN .....</b>	<b>36</b>
<b>Bảng 4: Bảng thông số của mô hình LSTM.....</b>	<b>38</b>
<b>Bảng 5: Bảng thông số của mô hình Bidirectional LSTM .....</b>	<b>40</b>
<b>Bảng 6: Bảng thông số của Hierarchical Model.....</b>	<b>42</b>
<b>Bảng 7: Bảng thông số của mô hình GRU .....</b>	<b>44</b>

## DANH MỤC CÁC TỪ VIẾT TẮT

AI	Trí tuệ (Trí thông minh) nhân tạo: Artificial Intelligence
API	Giao diện lập trình ứng dụng: Application Programming Interface
RNN	Mạng nơ-ron hồi quy
LSTM	Long-short Term Memory
GPU	Đơn vị xử lý đồ họa: Graphics Processing Unit
GRU	Gated recurrent unit
MLP	Mạng nơ-ron truyền thẳng nhiều lớp
NLP	Xử lý ngôn ngữ tự nhiên
CNN	Mạng tích chập
ANN	Mạng thần kinh nhân tạo
TN	True Negative
TP	True Positive
FN	False Negative
FP	False Positive
FC	Tầng kết nối đầy đủ

## TÓM TẮT KHÓA LUẬN

Khóa luận này tập trung vào xây dựng một hệ thống nhận dạng và dự đoán để xác định ký hiệu ngôn ngữ dựa vào thị giác máy tính, qua đó có thể đánh giá được khả năng học tập của học sinh cũng như những người có nhu cầu.

Đối tượng nghiên cứu bao gồm:

- Bảng chữ cái tiếng Việt bao gồm: alphabet (a-z), chữ số (0-9) và các dấu thanh trong ngôn ngữ tiếng Việt
- Thuật toán nhận diện các hành động diễn tả ngôn ngữ từ tay phải.

Phạm vi nghiên cứu:

- Các mô hình nhận diện các kí hiệu tay liên tục bao gồm: RNN, Bi-directional, Hierarchical Model, LSTM và GRU.
- Nhận diện các kí tự bao gồm: “a”, “â”, “ã”, “e”, “ê”, “o”, “ô”, “ơ” và các dấu thanh bao gồm dấu sắc, hỏi, ngã, nặng.

## **PHẦN MỞ ĐẦU**

### **1. LÝ DO CHỌN ĐỀ TÀI**

Ước tính có khoảng 15,500 trẻ em Việt Nam dưới 6 tuổi đang gặp khó khăn liên quan đến tình trạng nói và thính lực. Ngôn ngữ chính là một rào cản lớn đối với hầu hết những đứa trẻ này, người thiếu quyền tiếp cận giáo dục mầm non. Những năm gần đây, đã có những nỗ lực, những thúc đẩy bởi những tiến bộ to lớn trong khoa học và công nghệ nhằm để phát triển các công nghệ giúp hỗ trợ các em hoà nhập xã hội. Các giải pháp hiện tại chủ yếu tập trung vào việc hỗ trợ họ sử dụng công nghệ dịch ngôn ngữ ký hiệu thành ngôn ngữ nói. Tuy nhiên, việc dạy ngôn ngữ ký hiệu cho các em này là một nhiệm vụ khó khăn hơn - điều mà chưa nhận được đủ sự chú ý.

Hơn nữa, để nâng cao chất lượng giáo dục, tác giả đề xuất thiết lập một hệ thống hỗ trợ cho giáo viên, cho phép quan sát liên tục trong quá trình giảng dạy. Hệ thống này có thể theo dõi tương tác của học sinh, đánh giá hiệu suất của giáo viên và đưa ra đề xuất cụ thể để cải thiện quá trình giáo dục bằng cách sử dụng công nghệ cảm biến và phân tích dữ liệu. Phương pháp này giúp giáo viên nhận biết những điểm mạnh và những lĩnh vực cần phát triển, từ đó tối ưu hóa các chiến lược giảng dạy và tăng cường hiệu suất trong việc truyền đạt kiến thức cho học sinh có khuyết tật về thính lực và nói.

### **2. Ý NGHĨA KHOA HỌC VÀ THỰC TIỄN**

#### **2.1. Ý nghĩa khoa học**

Về mặt công nghệ, tại Việt Nam, đã phát triển các mô hình cánh tay robot và thuật toán để hỗ trợ và làm cho việc học ngôn ngữ trở nên thuận tiện. Nhằm tạo ra một phương pháp nhận diện thủ ngữ, nhóm nghiên cứu của thầy Nguyễn Thiên Bảo[1] trường Đại học Sư Phạm Kỹ thuật đã xây dựng mô hình để nhận biết thủ ngữ tiếng Việt. Hơn nữa, nhóm nghiên cứu tại Trường Đại học Công nghệ thông tin [2] đã xây dựng được gang tay có thể nhận dạng các kí hiệu thủ ngữ tiếng Việt sử dụng gia tốc kế.

Trên toàn thế giới, hệ thống cánh tay robot đọc dấu được phát triển tại Đại học Antwerp ở Bỉ được gọi là ASLAN[3]. Mục tiêu chính của nó là đọc và diễn đạt từ ngữ bằng ngôn ngữ ký hiệu, đây là một bước tiến lớn trong việc tạo ra một loại giao tiếp mới để cải thiện cuộc sống của những người khuyết thính. Nhóm đứng đầu dự án này đã nói rằng mục tiêu cuối cùng của họ là xây dựng một robot có hai cánh tay và khuôn mặt biểu



cảm. Điều này tạo ra một nền tảng giao tiếp tự nhiên và trực quan hơn, đồng thời giúp robot truyền đạt được sự phức tạp của ngôn ngữ ký hiệu. Bằng cách kết hợp công nghệ tiên tiến với sự hiểu biết sâu sắc về ngôn ngữ ký hiệu, ASLAN đang mở ra một kỷ nguyên mới trong việc hỗ trợ giao tiếp cho những người mất thính giác, mang đến cho họ nhiều cơ hội hơn để trò chuyện một cách tự nhiên và không bị ràng buộc. Điều này là minh chứng cho sức mạnh của công nghệ trong việc cải thiện cuộc sống con người.

Các nhà nghiên cứu đã phát triển nhiều phương pháp nhận diện ngôn ngữ ký hiệu nhằm cải thiện giao tiếp giữa những người khiếm thính, câm và những người không có khuyết tật. Những hệ thống này có khả năng chuyển đổi ngôn ngữ ký hiệu thành văn bản hoặc lời nói, và ngược lại. Hai phương pháp chính để nhận diện ngôn ngữ ký hiệu là dựa trên dữ liệu từ các cảm biến và dựa trên thị giác máy tính. Một nghiên cứu đã giới thiệu một loại găng tay thông minh mới sử dụng bộ điều khiển và cảm biến để phân tích cử chỉ tay. Tuy nghiên cứu này cho thấy độ chính xác đáng kể, nhưng việc sử dụng hệ thống dây điện và các thiết bị điện tử gắn trên cơ thể người dùng có thể hạn chế khả năng vận động của họ.

Nhiều phương pháp sử dụng công nghệ nhận diện dựa trên thị giác máy tính đã được trình bày, sử dụng cả học máy và học sâu, và chúng đã thể hiện những kết quả tích cực. Tuy nhiên, do các bộ dữ liệu được sử dụng trong những nghiên cứu này chủ yếu tập trung vào ngôn ngữ ký hiệu địa phương và không đủ đa dạng, nên việc phân biệt giữa các đặc trưng độc đáo của nguyên âm, dấu thanh và các dấu phụ trong tiếng Việt là khó khăn.

## **2.2. Ý nghĩa thực tiễn**

Dựa vào các nghiên cứu ở quá khứ, tác giả đề xuất nghiên cứu đề tài xây dựng hệ thống nhận dạng thủ ngữ để giúp giáo viên cũng như học sinh có thể tự kiểm tra ngôn ngữ một cách tự động và dễ dàng hơn trên máy tính cá nhân.

## **3. MỤC TIÊU VÀ NHIỆM VỤ CỦA NGHIÊN CỨU**

### **3.1. Mục tiêu**

Từ thực trạng trên đòi hỏi các mục tiêu đặt ra:

- Tìm hiểu nghiên cứu về thị giác máy tính, về các phương pháp học sâu.
- Giao tiếp với máy tính thông qua camera và nhận diện được các ký hiệu thủ ngữ
- Xây dựng và huấn luyện mô hình máy học để nhận diện các ký hiệu thủ ngữ tương ứng.
- Cài đặt ứng dụng nhận diện thủ ngữ để máy tính nhận diện cử chỉ.

### **3.2. Nhiệm vụ**

Nhiệm vụ đặt ra khi nghiên cứu về đề tài gồm:

- Tìm hiểu về Deep Learning.
- Tìm hiểu về Thị giác máy tính.
- Tìm hiểu các thuật toán tối ưu.
- Xây dựng ứng dụng cơ bản nhận diện các kí hiệu thủ ngữ thông qua camera trên thiết bị không có GPU.

## **4. MÔ TẢ BÀI TOÁN**

Trong đề tài này được chia làm 2 phần.

*Thứ nhất*, nghiên cứu về các hệ thống thủ ngữ trên toàn thế giới. Trong phần này, các nghiên cứu về các hệ thống thủ ngữ trên toàn thế giới đang đóng một vai trò quan trọng trong việc nâng cao hiểu biết về cách mà cộng đồng người điếc và khiếm thính giao tiếp. Trong lĩnh vực này, việc đọc các nghiên cứu khoa học liên quan đóng một vai trò quan trọng trong việc hiểu rõ hơn về tiến triển, thách thức và tiềm năng của các hệ thống thủ ngữ. Các nghiên cứu này thường tập trung vào phân tích cú pháp, ngữ pháp và cách thức truyền đạt ý nghĩa trong các hệ thống thủ ngữ khác nhau. Điều này bao gồm cả việc nghiên cứu về cơ sở hạ tầng ngôn ngữ ký hiệu và các biểu hiện hình thức khác nhau trong cộng đồng. Đồng thời, nghiên cứu cũng chú trọng đến hiệu quả của các công nghệ hỗ trợ, như các ứng dụng di động và hệ thống dịch tự động, trong việc tăng cường giao tiếp cho người sử dụng hệ thống thủ ngữ.

*Thứ hai*, nghiên cứu về các mô hình thuật toán giúp nhận diện được ký hiệu thủ ngữ thông qua camera trên máy tính của người dùng để kiểm tra liệu ký tự đó có đúng hay không. Mô hình trong khoá luận bao gồm sử dụng thư viện MediaPipe để trích xuất 21 điểm trên bàn tay phải và trích xuất các tọa độ trên không gian ba chiều của khung xương bàn tay từ hình ảnh mà camera thu được.

## **5. KẾT QUẢ DỰ KIẾN ĐẠT ĐƯỢC**

Thông qua các tọa độ đã thu được bằng MediaPipe, xây dựng được mô hình vừa có độ chính xác cao cũng như vừa nhẹ để có thể áp dụng vào trên những chiếc máy tính không có GPU.

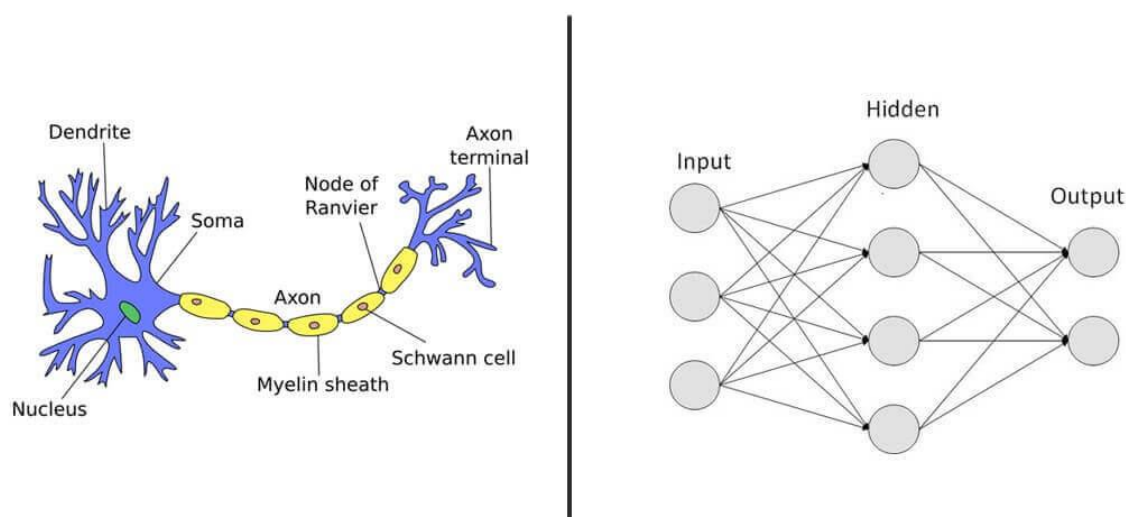
## PHẦN NỘI DUNG

### CHƯƠNG 1: CƠ SỞ LÝ THUYẾT

#### 1.1. NEURAL NETWORK

##### 1.1.1. Giới thiệu sơ lược về Neural Network

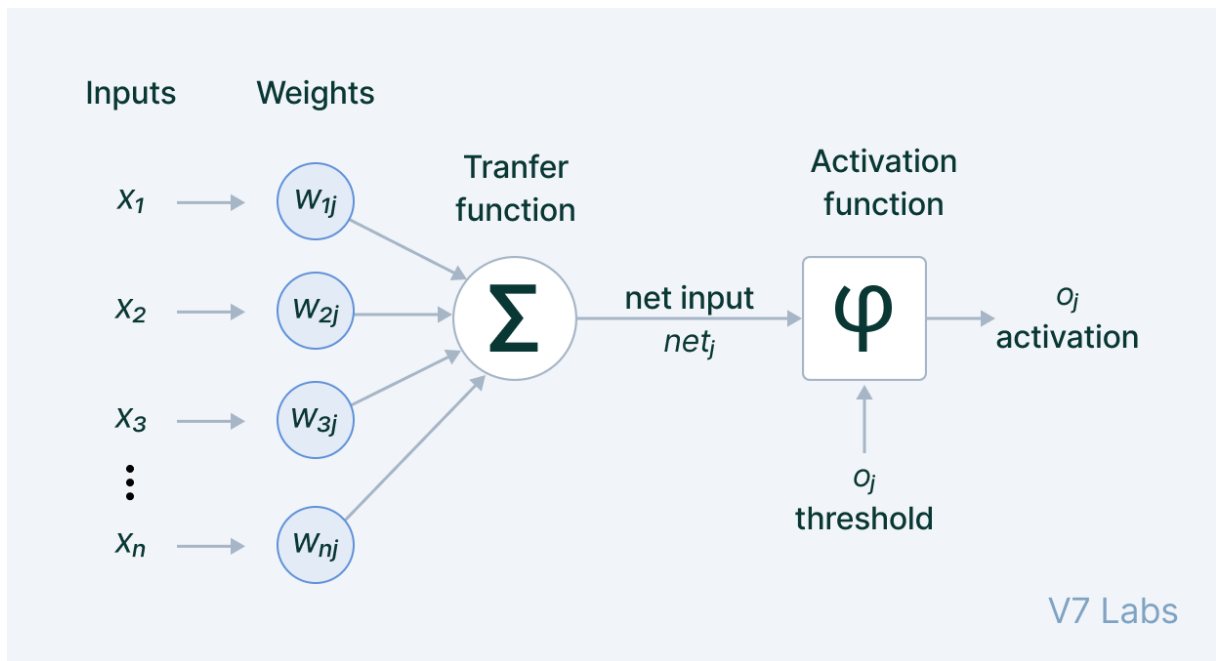
Mạng nơ-ron là một phương thức trong lĩnh vực trí tuệ nhân tạo, được sử dụng để dạy máy tính có thể xử lý dữ liệu và nó được lấy cảm hứng từ bộ não con người. Trong cấu trúc của Deep Learning, chúng sử dụng các nút hay còn gọi là nơ-ron liên kết với nhau trong một cấu trúc phân lớp tương tự như bộ não con người. Thông qua cách này, máy tính có thể học hỏi từ các sai lầm của chúng và liên tục cải thiện. Vì vậy, mạng nơ-ron nhân tạo có thể giải quyết các vấn đề phức tạp hơn so với máy học như nhận diện khuôn mặt, nhận diện vật thể, ... với độ chính xác cao hơn.



Hình 1: Cấu trúc mạng nơ-ron được lấy cảm hứng từ tế bào nơ-ron của não người

Mạng nơ-ron có thể giúp máy tính đưa ra các quyết định thông minh chỉ với sự hỗ trợ hạn chế của con người. Chúng chỉ có thể học hỏi và dựng mô hình qua các mối quan hệ giữa dữ liệu đầu vào và dữ liệu đầu ra. Mạng nơ-ron có thể hiểu rõ dữ liệu phi cấu trúc và đưa ra các nhận xét chung.

Mạng nơ-ron được sử dụng trong nhiều trường hợp trải dài khắp các lĩnh vực từ y khoa, dự báo chứng khoán hay xử lý ảnh.



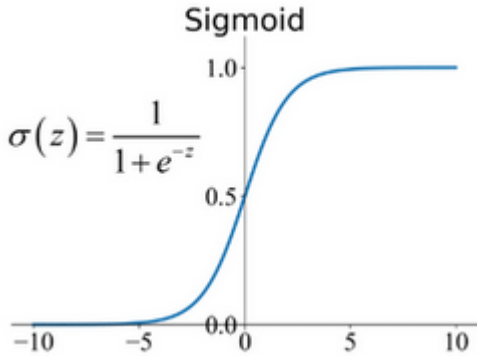
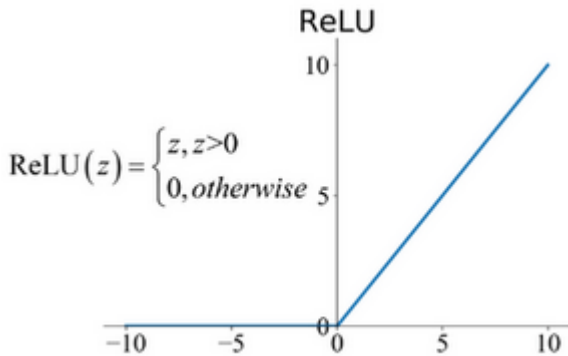
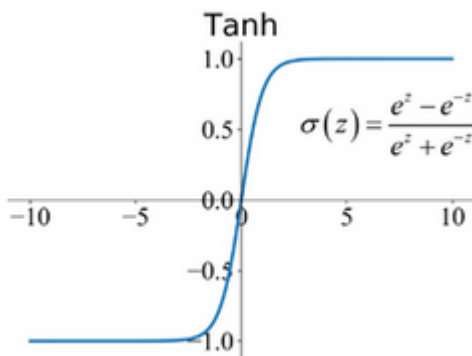
Hình 2: Các thành phần cơ bản của mạng nơ-ron nhân tạo

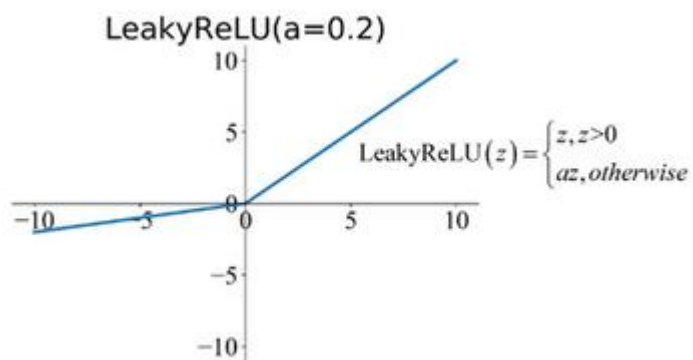
Các hình trên sẽ có các thành phần cơ bản[4] bao gồm:

- Dữ liệu đầu vào (input): Mỗi nút mạng nơ-ron sẽ nhận dữ liệu từ những nơ-ron ở lớp đầu tiên sau đó dữ liệu sẽ được truyền từ nơ-ron lớp trước sang lớp sau. Đầu vào có thể là các giá trị scalar hoặc vector n chiều.
- Trọng số (weight): Giữa các nơ-ron có các liên kết, mỗi liên kết sẽ có một trọng số tương ứng. Thông tin càng quan trọng, trọng số càng lớn. Trọng số này sẽ quyết định mức độ quan trọng của đầu vào đối với đầu ra của nơ-ron nhân tạo.
- Hàm tổng (summing function): Thường được dùng để tính tổng các tích của đầu vào và trọng số liên kết với nó.
- Ngưỡng (threshold): Chính là điều kiện để chọn rằng tín hiệu từ nơ-ron lớp trước có thể truyền sang nơ-ron lớp sau hay không.
- Hàm truyền hay hàm kích hoạt (activation function): Hàm này sẽ thực hiện nhiệm vụ tính tổng trọng số đầu vào và tạo ra đầu ra của nơ-ron, ngoài ra activation function có thể giới hạn đầu ra trong một phạm vi cụ thể.
- Đầu ra (output); Thông qua kết quả từ activation function, kết quả sẽ được tạo ra từ nó. Đầu ra có thể là một số, một vector hoặc là sẽ là đầu vào cho nơ-ron trong lớp tiếp theo.

- Một số hàm truyền thông dụng[5], [6]:

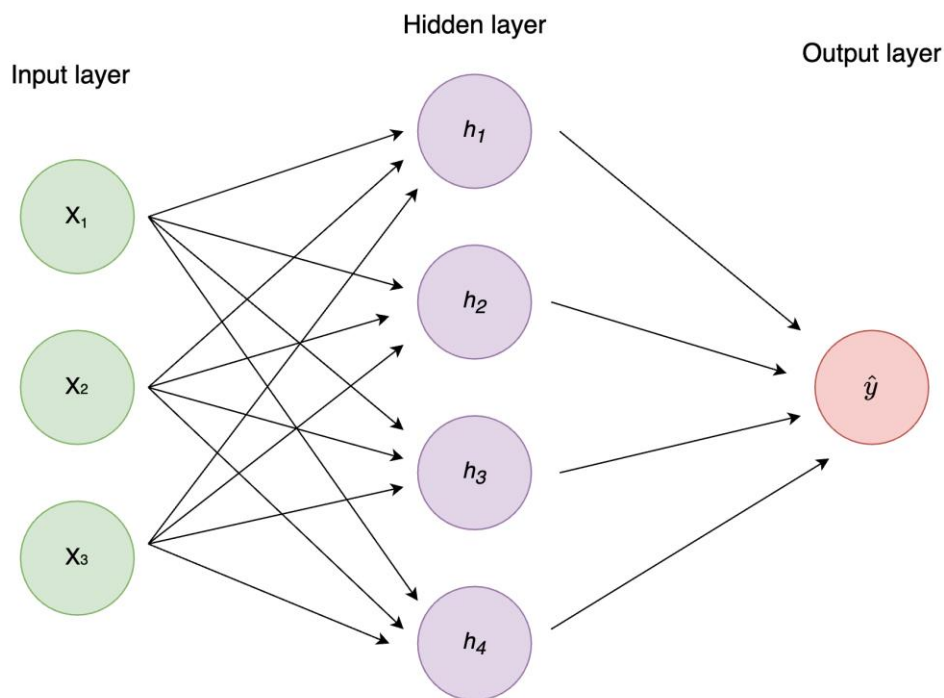
Bảng 1: Các hàm kích hoạt thường dùng

Tên hàm	Đồ thị	Định nghĩa
Sigmoid	 <p>Sigmoid</p> $\sigma(z) = \frac{1}{1 + e^{-z}}$	Hàm Sigmoid sẽ cho giá trị trả về nằm trong khoảng từ $[0, 1]$ . Sigmoid thường được sử dụng trong mô hình phân loại.
ReLU	 <p>ReLU</p> $\text{ReLU}(z) = \begin{cases} z, & z > 0 \\ 0, & \text{otherwise} \end{cases}$	Hàm ReLU sẽ thực hiện lọc các giá trị bé hơn 0. Và chúng thường được sử dụng trong mô hình hồi quy.
Tanh	 <p>Tanh</p> $\sigma(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$	Hàm Tanh sẽ nhận đầu vào là một số thực và chuyển chúng thành giá trị trong khoảng $(-1, 1)$ .

Tên hàm	Đồ thị	Định nghĩa
LeakyReLU		Thay vì trả về giá trị 0 với các đầu vào bé hơn 0, Leaky ReLU sẽ tạo đường dốc nhỏ. Và sẽ giải quyết vấn đề “dying ReLU”

### 1.1.2. Mô hình tổng quát

Mạng nơ-ron là một tập hợp nhiều các nơ-ron với nhau và được xây dựng thành từng lớp. Chúng còn có tên gọi là những tầng perceptron hay perceptron đa tầng. Và mỗi mạng nơ-ron thường sẽ có kiểu tầng như sau.



Hình 3: Cấu trúc đơn giản của mạng nơ-ron

- Tầng input hay input layer: Tầng này sẽ thực hiện đọc các dữ liệu vào mạng.
- Tầng mạng ẩn hay hidden layer: Tầng này nằm giữa tầng input và tầng output, nó sẽ thực hiện quá trình suy luận logic của mạng.

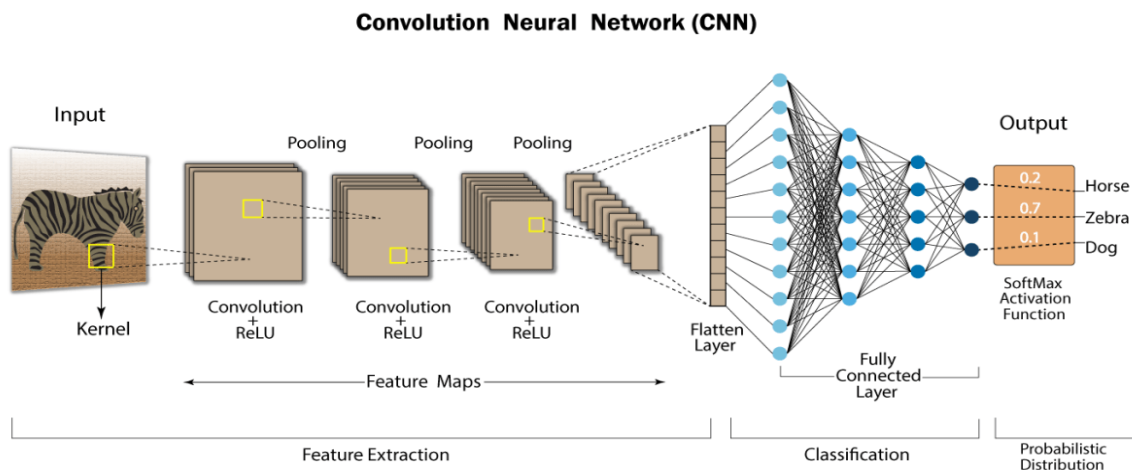
- Tầng output hay output layer: Tầng này sẽ thực hiện quá trình chuẩn hoá đầu ra của mạng.

Một cấu trúc neuron network có thể có một tầng vào và một tầng ra hoặc có rất nhiều tầng ẩn với nhau.

## 1.2. DEEP LEARNING

Một lĩnh vực con của trí tuệ nhân tạo được biết đến với tên gọi "Deep Learning" đã được thúc đẩy bởi sự tiến bộ của mạng nơ-ron đa tầng (MLP). Cuộc đua để xây dựng kiến trúc mô hình với đến vài chục, hàng trăm hoặc thậm chí nhiều hơn các tầng mạng đã bắt đầu với sự xuất hiện của Deep Learning, với mục tiêu giải quyết các vấn đề cụ thể trong lĩnh vực như thị giác máy tính và xử lý ngôn ngữ tự nhiên.

Trong lĩnh vực học sâu, mạng nơ-ron tích chập, hay CNN, là một mô hình quan trọng và thường xuyên được sử dụng. So với các phương pháp truyền thống, CNN đã tối ưu hóa quá trình trích xuất đặc trưng từ hình ảnh, giúp tiết kiệm thời gian. CNN mang lại một phương pháp mới để giải quyết các vấn đề liên quan đến nhận diện hình ảnh. Đặc biệt, CNN sử dụng phép nhân ma trận và các khái niệm đại số tuyến tính khác để nhận diện các mẫu trong hình ảnh. Điều này đồng thời đặt ra yêu cầu cao về xử lý tính toán, đòi hỏi sự hỗ trợ của các đơn vị xử lý đồ họa mạnh mẽ (GPUs) trong quá trình đào tạo mô hình.



**Hình 4: Cấu trúc mạng nơ-ron tích chập**

Các mô hình CNN thường xuyên tuân theo một kiến trúc mạng cụ thể. Mỗi mô hình bao gồm một chuỗi các lớp được kết nối, trong đó đầu ra của mỗi lớp trở thành đầu vào cho

lớp tiếp theo. Ba lớp mạng chính được thể hiện trong Hình trên là lớp convolution, lớp Pooling và lớp Fully connected[4].

- Lớp Pooling: Lớp Pooling là một lớp đặt sau các lớp tích chập. Nhiệm vụ của nó là giảm kích thước của các bản đồ đặc trưng, từ đó giúp giảm gánh nặng tính toán cho các lớp theo sau. Khi chúng ta tiến xa hơn vào mạng, lớp tích chập và lớp Pooling tương ứng đã cải thiện tính trừu tượng chung của các đặc trưng.

- Lớp Fully Connected (FC): Lớp này hoạt động tương tự như một mạng nơ-ron đa tầng (ANNs). Trong giai đoạn cuối của mạng, lớp FC biến đổi các đặc trưng và đưa ra kết quả.

- Lớp Convolution: Lớp này tạo ra các bản đồ đặc trưng bằng cách nhân chập ảnh đầu vào với các ma trận bộ lọc, sau đó chuyển qua các hàm kích hoạt.

### **1.2.1. Action Recognition**

Việc học để nhận diện và phân biệt giữa các hoạt động khác nhau là một vấn đề quan trọng trong thị giác máy tính cũng như đối với con người. Một số định nghĩa về hành động đã được đề xuất gần đây, tuy nhiên, trong luận văn này, thuật ngữ sẽ được sử dụng một cách mơ hồ và bao gồm các hành động như tương tác với một đối tượng, tương tác với một hoặc nhiều người khác, hoặc đơn giản là di chuyển cơ thể. Các hoạt động của con người dễ dàng quan sát trong các tình huống như chạy, đạp xe, uống nước, vẫy tay, hoặc thuyết trình luận văn.

Điều này làm chệch khỏi các tiêu chí nhận diện truyền thống, tạo ra nhiều thách thức đối với học sâu nói chung và thị giác máy tính nói riêng. Thay vì sử dụng các mô hình mạng để nhận diện đối tượng hoặc nhãn trong một bức ảnh, chúng ta hiện nay tìm kiếm sự tương quan qua một chuỗi hình ảnh và sử dụng những tương quan đó để dự đoán các hành động sẽ diễn ra.

Hiện nay, trên nhiều nền tảng trực tuyến, chúng ta đang chứng kiến một lượng lớn dữ liệu video. Năm 2015, thống kê cho thấy hơn 400 giờ video được tải lên YouTube mỗi phút, với số lượng lớn dữ liệu như vậy, việc quản lý và kiểm soát nội dung là một thách thức quan trọng, đặc biệt là để xử lý các hành vi có thể gây hại như bạo lực và tấn công.

Sự xuất hiện của công nghệ nhận diện hành động đã đóng góp quan trọng vào việc nâng cao chất lượng của các sản phẩm truyền thông văn hóa và thúc đẩy tôn trọng trong môi trường xã hội trực tuyến. Ngoài ra, nhận diện hành động còn được áp dụng rộng rãi trong

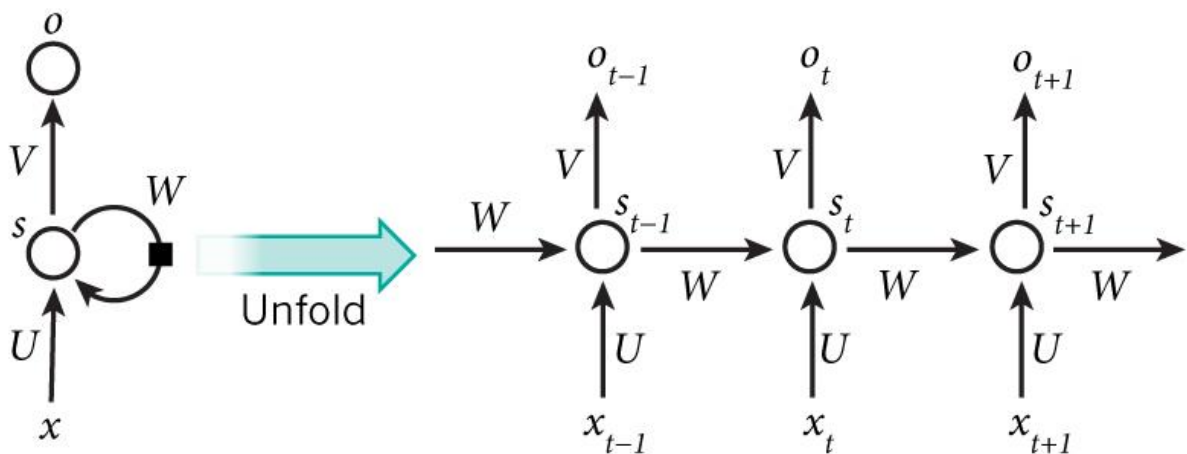


các lĩnh vực như tương tác con người và máy tính, truy xuất nội dung video và hệ thống hỗ trợ người. Tuy nhiên, nhiệm vụ này trở nên phức tạp khi phải đối mặt với các thách thức như các hoạt động chồng lấp, chuyển động tương tự nhau hoặc các góc quay camera khác nhau.

### 1.2.2. Mạng hồi quy RNN

#### 1.2.2.1. Mạng hồi quy RNN là gì?

Mạng hồi quy RNN là một dạng đặc biệt của mạng nơ-ron nhân tạo, được thiết kế để xử lý dữ liệu dạng chuỗi và dữ liệu có tính tuần tự. Khác với các mạng nơ-ron thông thường, mạng sẽ xử lý đầu vào và đầu ra một cách tuần tự. RNN có khả năng ghi nhớ trạng thái trước đó và sử dụng nó để xử lý đầu vào tiếp theo trong chuỗi. Điều này xuất sắc trong nhiều ứng dụng như xử lý ngôn ngữ tự nhiên, dịch máy, dự đoán chuỗi thời gian. Bên dưới sẽ là hình ảnh cấu trúc cơ bản và toán học của RNN.[6]



Hình 5: Cấu trúc cơ bản của mạng RNN

Trong đó:

- $x_t$  là giá trị đầu vào tại bước t
- $o_t$  là giá trị đầu ra tại bước t
- $s_t$  là trạng thái ẩn tại bước t

#### 1.2.2.2. Phương thức hoạt động

Thông qua Hình 5 bên trên, ta sẽ có phương thức hoạt động như sau[6]:

- Đầu vào  $x_t$ : Tại mỗi thời điểm t, mô hình nhận vào mọi giá trị đầu vào  $x_t$
- Trạng thái ẩn  $s_t$  là một trạng thái mà RNN sử dụng để chứa các thông tin từ quá khứ và cập nhật chúng sau mỗi lần đưa đầu vào mới để xử lý.

- Cập nhật trạng thái ẩn  $s_t$ :

- Cứ mỗi timestep  $t$ , trạng thái ẩn mới  $s_t$  được tính toán từ trạng thái ẩn trước đó là  $s_{t-1}$  và đầu vào hiện tại  $x_t$  bằng một hàm kích hoạt là Tang hoặc Sigmoid.

$$h_t = f(W_{hh} * h_{t-1} + W_{xh} * x_t + b_h)$$

● Trong đó:

- $W_{hh}$  là ma trận trọng số giữa các trạng thái ẩn.
- $W_{xh}$  là ma trận trọng số giữa đầu vào và trạng thái ẩn.
- $b_h$  là bias
- $f$  là hàm kích hoạt.

- Dự đoán đầu ra:

Tại trạng thái ẩn  $h_t$  sau khi được cập nhật thông qua các giá trị đầu vào, chúng có thể sử dụng để dự đoán giá trị đầu ra tiếp theo  $y^t$  tại thời điểm  $t$  theo công thức sau:

$$y^t = g(W_{yh} * h_t + b_y)$$

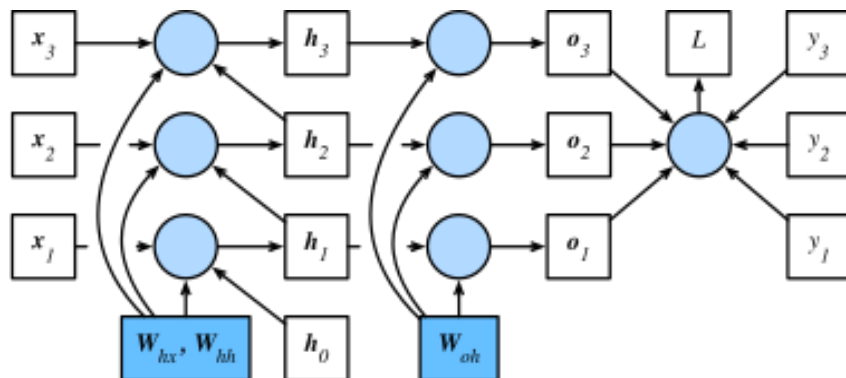
● Trong đó:

- $W_{yh}$  là ma trận trọng số giữa trạng thái ẩn và đầu ra
- $b_y$  là bias
- $g$  là hàm kích hoạt

- Quá trình kia sẽ lặp đi lặp lại quá trình như thế. Cập nhật liên tục trọng số và trạng thái ẩn cho mỗi điểm dữ liệu như thế cho đến hết dữ liệu.

Sau quá trình liên tục cập nhật trọng số và trạng thái ẩn bên trên, chúng ta sẽ xét đến quá trình tính toán lan truyền ngược (backpropagation through time).

Trước khi phân tích quá trình lan truyền ngược, ta xem xét hình 6 bên dưới[7]



Hình 6: Mô hình RNN ba lớp

Ta tách  $W$  thành các tập hợp ma trận trọng số bao gồm  $W_{hx}$ ,  $W_{hh}$ ,  $W_{oh}$  ta có:

$$h_t = W_{hx}x_t + W_{hh}h_{t-1} \text{ và } o_t = W_{oh}h_t. [7]$$

Sau khi thực hiện tách ra, chúng ta thực hiện quá trình gradient  $\frac{\partial L}{\partial W_{hx}}$ ,  $\frac{\partial L}{\partial W_{hh}}$ ,  $\frac{\partial L}{\partial W_{oh}}$  với  $L$  là hàm mất mát đã chọn trước là:

$$L(x, y, W) = \sum_{t=1}^T l(o_t, y_t) [7]$$

Đối với đạo hàm  $W_{oh}$  ta có:

$$\partial_{W_{oh}} L = \sum_{t=1}^T prod(\partial_{o_t} l(o_t, y_t), h_t) [7]$$


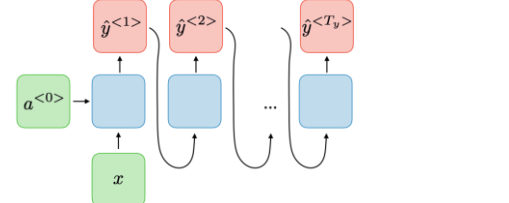
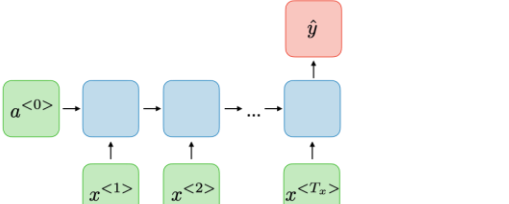
Đối với đạo hàm  $W_{hx}$  và  $W_{hh}$  ta sẽ có

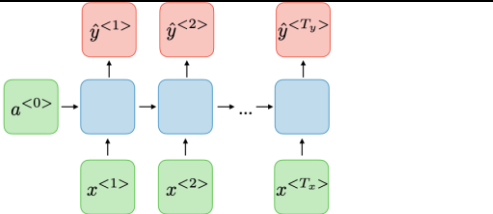
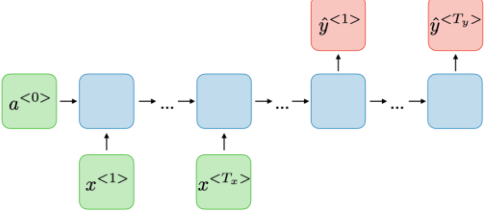
$$\partial_{W_{hh}} h_t = \sum_{j=1}^t (W_{hh}^T)^{t-j} h_j \text{ và } \partial_{W_{hx}} h_t = \sum_{j=1}^t (W_{hh}^T)^{t-j} x_j [7]$$

### 1.2.2.3. Một số dạng RNN thường gặp

Các mô hình RNN thường sẽ được sử dụng trong lĩnh vực xử lý ngôn ngữ tự nhiên và nhận diện âm thanh. Do đó các biến thể của RNN sẽ được tạo ra để phục vụ nhiều mục đích khác nhau. Bên dưới là bảng các loại RNN thường được sử dụng[8]

**Bảng 2: Các loại RNN thường gặp**

Các loại RNN	Hình minh họa	Ví dụ
One to one $T_x = T_y = 1$		Mạng nơ-ron truyền thống
One to many $T_x = 1, T_y > 1$		Sử dụng để sinh nhạc
Many to one $T_x > 1, T_y = 1$		Semantic segmentation

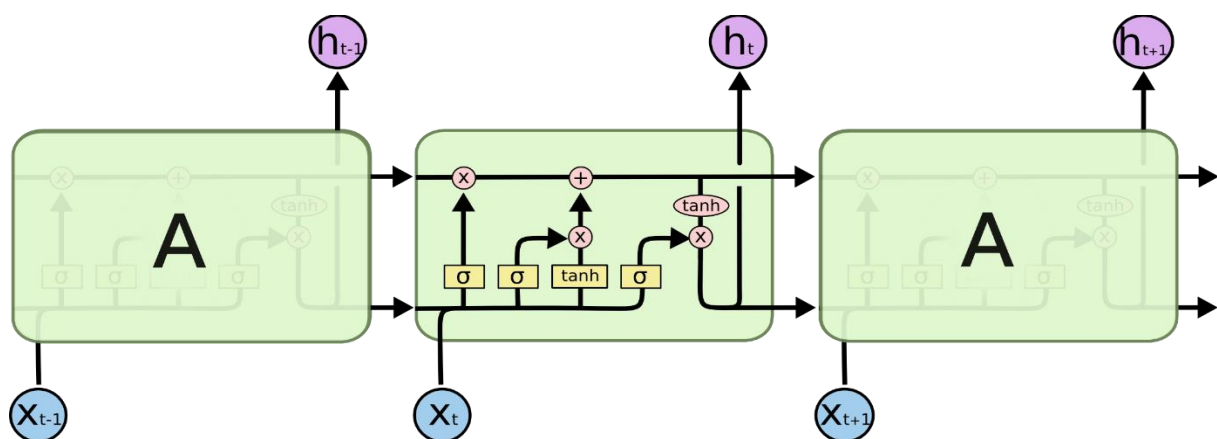
Các loại RNN	Hình minh họa	Ví dụ
Many to many $T_x = T_y$		Name entity recognition
Many to many $T_x \neq T_y$		Machine translation

### 1.2.3. Mạng hồi quy LSTM và biến thể Bi-directional LSTM

#### 1.2.3.1. Mạng hồi quy LSTM là gì

Điểm nổi bật của RNN chính là sử dụng dữ liệu từ quá khứ để dự đoán cho hiện tại, nhưng chỉ trong trường hợp thông tin ngắn thì việc sử dụng RNN khả thi, mà thực tế thông tin dài lại chiếm phần lớn. Do đó, RNN sẽ bắt đầu không thể nhớ và học được nữa, đó chính là vấn đề vanishing gradient của RNN và đã được Hocheriter và Bengio tìm hiểu khá sâu và tìm được những lý do căn bản để giải thích RNN không thể học được thông tin dài.

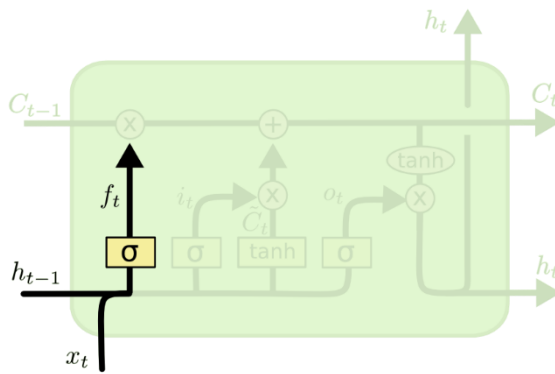
LSTM ra đời để giải quyết vấn đề đó, LSTM hay Long-short Term Memory là một dạng cải tiến của RNN. Nó có khả năng học được thông tin dài mà không bị vấn đề vanishing gradient như RNN từng mắc phải. LSTM được thiết kế để giải quyết vấn đề phụ thuộc xa (long-term dependence). Việc nhớ thông tin của LSTM giải quyết bằng cách sử dụng memory cell[9].



Hình 7: Cấu trúc cơ bản của LSTM

Điểm khác biệt chính là memory cell để có thể lưu trữ thông tin qua timestep. Ngoài cấu trúc của LSTM có ba cổng bao gồm: input gate, forget gate và output gate. Trong đó, input gate sẽ quyết định giá trị đầu nào sẽ lưu vào trong bộ nhớ, forget gate sẽ quyết định thông tin nào cần xoá khỏi bộ nhớ và output gate sẽ quyết định thông tin nào sẽ đi ra từ ô nhớ. Thông qua các cổng này, quá trình lan truyền ngược gradient sẽ không bị mất trong quá trình cập nhật các cổng cũng như bias. Với việc sử dụng cơ chế trên, LSTM có thể giải quyết được vấn đề vanishing gradient.

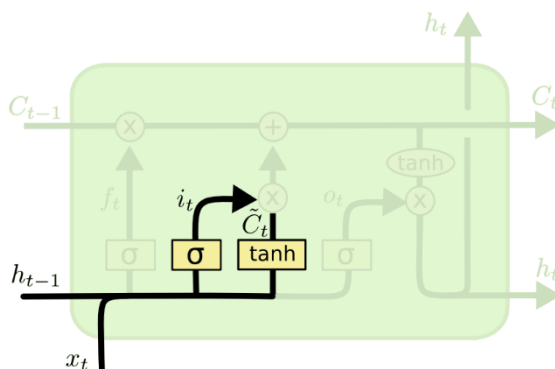
Đối với cổng output gate, LSTM quyết định xem thông tin nào cần bỏ. Quyết định được đưa ra bởi hàm Sigmoid. Giá trị của hàm Sigmoid sẽ trả về trong khoảng từ  $[0,1]$ . Nếu kết quả trả về là 1, thông tin sẽ được giữ lại toàn bộ, nếu kết quả trả về là 0 thì thông tin sẽ bị bỏ đi[8].



$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$$

Hình 8: Reset gate

Sau khi xác định thông tin nào sẽ được giữ lại hay bỏ đi, dữ liệu sẽ được thêm vào mới thông qua input gate.

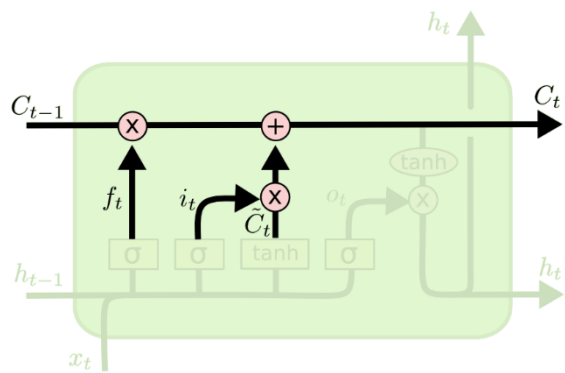


$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$$

Hình 9: Cập nhật giá trị cho ô trạng thái

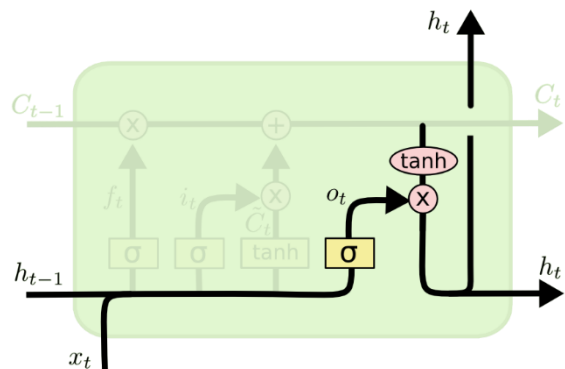
Sau khi thực hiện đưa dữ liệu vào, sẽ tới lúc cập nhật trạng thái. Với trạng thái cũ tương ứng sẽ thực hiện tính toán tương ứng để có thể cập nhật vào trạng thái[8].



$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Hình 10: Ô trạng thái mới

Sau đó đầu ra sẽ dựa trên ô trạng thái đó. Khi đó Sigmoid sẽ quy định phần nào ở ô trạng thái sẽ ở đầu ra. Ô trạng thái đó sẽ đi qua hàm Tanh và nhân với đầu ra của Sigmoid, đó là phần mà chúng ta quyết định[8]



$$o_t = \sigma (W_o [h_{t-1}, x_t] + b_o)$$

$$h_t = o_t * \tanh (C_t)$$

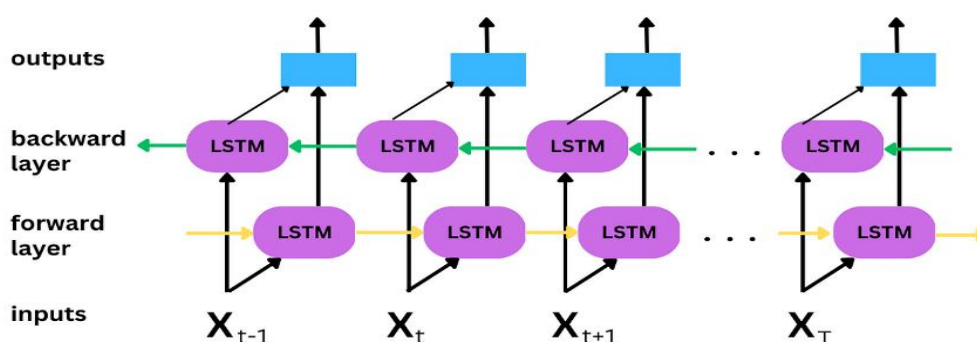
Hình 11: Đầu ra

Mặc dù mô hình LSTM có ưu điểm giải quyết được bài toán vanishing gradient, tuy nhiên mô hình LSTM vẫn còn một số nhược điểm lớn:

- Khả năng nắm bắt thông tin thấp: Đối với một mô hình hồi quy, việc sử dụng thông tin từ quá khứ là chưa đủ, nó cần phải biết sử dụng cả thông tin từ quá khứ và tương lai.
- Khi chỉ nắm bắt được thông tin từ quá khứ, cho nên khi sử dụng LSTM với một chuỗi dài vẫn có thể xảy ra vấn đề gradient explosion.
- Ngoài ra, mô hình với cơ chế nhiều cổng và nhiều thành phần, việc training mô hình sẽ lâu hơn và yêu cầu các giá trị siêu tham số cần chọn lựa kỹ lưỡng hơn. Với việc mô hình có tham số gấp bốn lần so với RNN, LSTM vẫn sẽ có nhiều thách thức khi gradient vẫn có thể biến mất khi mô hình quá sâu.

### 1.2.3.2. Biến thể Bidirectional LSTM

Bidirectional LSTM là một phiên bản mở rộng của mạng RNN. Mạng LSTM có 04 lớp giao tiếp khác nhau và cũng chính nhờ điều này mà LSTM sẽ hoạt động tốt hơn trong bài toán khi đầu vào là mỗi chuỗi dữ liệu dài. Đặc biệt, LSTM chỉ có thể lấy thông tin từ quá khứ để dự đoán cho tương lai nhưng Bidirectional LSTM lại có thể lấy thông tin từ hai hướng, cả tương lai và quá khứ giúp cho mô hình Bidirectional LSTM có thể nắm bắt được ngữ cảnh từ cả hai phía của dữ liệu.



Hình 12: Mô hình Bidirectional LSTM

Thông qua Hình 12 bên trên[10], hai mô hình LSTM hoạt động trên luồng dữ liệu đầu vào tuần tự và các luồng thông tin được truyền đi theo hai hướng ngược nhau. Tuy nhiên, một vấn đề chúng ta cần biết là giá trị tương lai là gì, làm sao để có thể biết được chính xác giá trị tương lai. Đối với các giá trị trong quá khứ, luồng thông tin được truyền trong backward layer. Vậy thì giá trị tương lai thực chất được tính từ forward layer được bổ sung. Bởi vì có hai layer khác nhau như thế, việc nắm bắt thông tin tuần tự sẽ thực hiện một cách dễ dàng hơn, do đó hiện này Bi-LSTM được sử dụng để giải quyết các vấn đề như phân loại cũng như dự đoán. Khi input hiện tại được đưa vào, đầu vào quá khứ vẫn được xem xét để phân loại và phân tích vấn đề hiện tại, do đó dự đoán cũng sẽ trở nên tốt hơn theo thời gian vì mô hình đã được đào tạo trước đó với các giá trị trước đó cũng sẽ được cập nhật trọng số tốt hơn để kết quả đầu ra tốt hơn. Vì vậy, việc sử dụng mô hình Bidirectional LSTM khả thi trong đề tài.

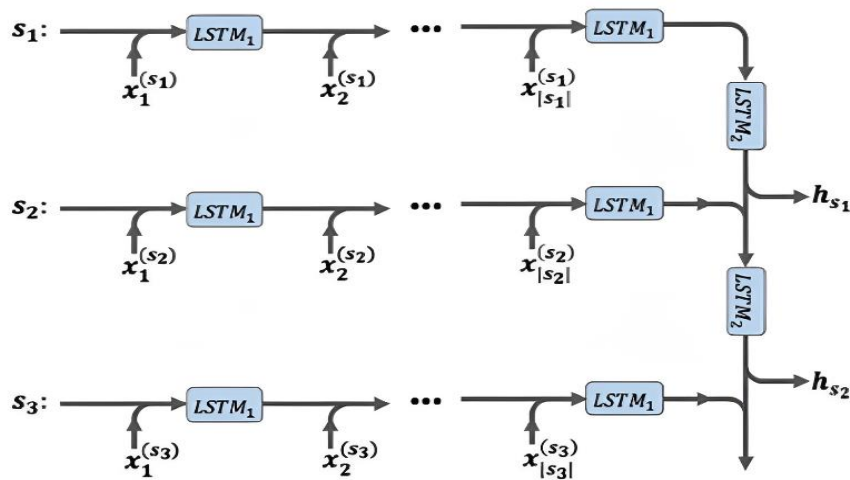
#### 1.2.4. Mô hình phân cấp Hierarchical Model

Hierarchical model hay còn gọi là mô hình phân cấp. Đúng như tên gọi của nó, mô hình được thiết kế để xử lý phân cấp hoặc tổ chức trong dữ liệu. Mô hình được áp dụng trong nhiều lĩnh vực như xử lý ngôn ngữ tự nhiên, xử lý ảnh hoặc các kiểu dữ liệu nhiều chiều.

Các mạng nơ ron sâu có thể sử dụng để thực hiện tác vụ học phân cấp hiệu quả một cách thực nghiệm, trong đó các lớp sẽ học các biểu diễn có sự hữu ích của dữ liệu.

Trong khoá luận, mô hình phân cấp ở đây được sử dụng là lớp “TimeDistributed” được áp dụng để xử lý dữ liệu theo thời gian[11]. “TimeDistributed” giúp mô hình có khả năng học các mối quan hệ phức tạp trong dữ liệu thời gian bằng cách cho vào một lớp trong lớp Dense. Lớp này sẽ bao bọc lớp Dense và được sử dụng cho từng bước trong dữ liệu chuỗi, giúp mô hình có thể học được biểu diễn phức tạp tại mỗi thời điểm làm cho mô hình trở nên mạnh mẽ và linh hoạt hơn trong việc xử lý dữ liệu dạng chuỗi. Lớp “TimeDistributed” có thể đặt ở nhiều vị trí, trong đó nếu:

- Đặt ở đầu mô hình: Việc đặt lớp “TimeDistributed” sẽ áp dụng một hoặc nhiều lớp fully connected cho mỗi bước thời gian ngay từ đầu của quá trình training.
- Đặt ở giữa các lớp: Giúp mô hình có thể học các mối quan hệ phức tạp giữa các bước thời gian sau khi đã trích xuất thông tin cấp cao từ các lớp LSTM.
- Đặt ở cuối mô hình: Giúp xử lý dữ liệu chuỗi thời gian trước khi đưa vào lớp fully connected cuối cùng. Ngoài xử lý ra, chúng còn giúp mô hình tập trung vào thông tin quan trọng từ chuỗi thời gian trước khi đưa ra dự đoán cuối cùng.

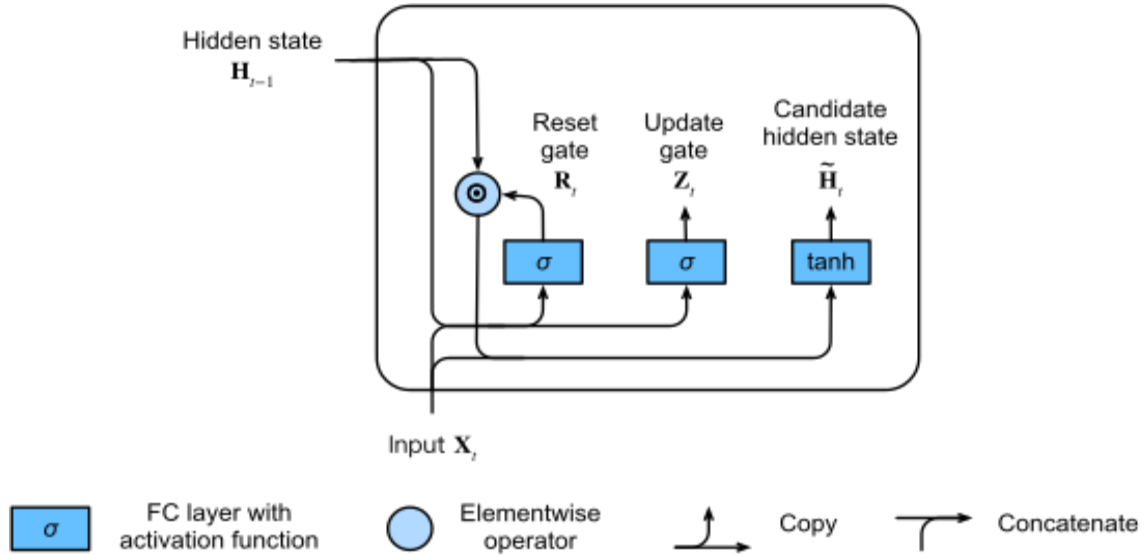


Hình 13: Cấu trúc của mô hình phân cấp Hierarchical Model



### 1.2.5. Mô hình GRU

Là một biến thể khác của mô hình RNN được thiết kế để giải quyết vấn đề vanishing gradient trong quá trình huấn luyện RNN. Phương pháp căn bản nhất của GRU để giải quyết vấn đề trên là sử dụng các cổng để điều chỉnh luồng thông tin, giúp mô hình có khả năng nhớ lâu hơn.[12]



Hình 14: Kiến trúc mô hình GRU

Điểm khác biệt giữa GRU và LSTM là GRU hỗ trợ việc kiểm soát trạng thái ẩn. Do đó, GRU có các cơ chế được học để quyết định khi nào nên cập nhật và khi nào nên xóa trạng thái ẩn. GRU có ba cổng là cổng reset, cổng update, cổng trạng thái ẩn tiềm năng (Candidate hidden state).

Đối với cổng reset, ta có công thức sau:

$$R_t = \sigma(X_t W_{xr} + H_{t-1} W_{hr} + b_r) [12]$$

Tuy nhiên, khi chúng ta kết hợp với candidate hidden state, khác so với cổng input gate của RNN, việc sử dụng hàm Tanh sẽ cho chúng ta kết quả trong  $(-1, 1)$ . Do đó, nếu muốn giảm ảnh hưởng của trạng thái trước đó, ta thực hiện nhân  $H_{t-1}$  với  $R_t$  theo từng phần tử. Nếu  $R_t$  có giá trị gần bằng 1, thì GRU sẽ giống RNN ban đầu, nếu  $R_t$  bằng 0 thì mô hình sẽ có đầu ra là một mạng perceptron đa tầng. Vậy, khi có candidate hidden state vào, ta có công thức.

$$\tilde{H}_t = \tanh(X_t W_{xh} + (R_t \odot H_{t-1}) W_{hh} + b_h) . [12]$$

Đối với cổng cập nhật, khi áp dụng tổ hợp logic giữa trạng thái cũ và trạng thái mới, ta có phương trình cập nhật như sau:

$$H_t = Z_t \odot H_{t-1} + (1 - Z_t) \odot \widetilde{H}_t. [12]$$

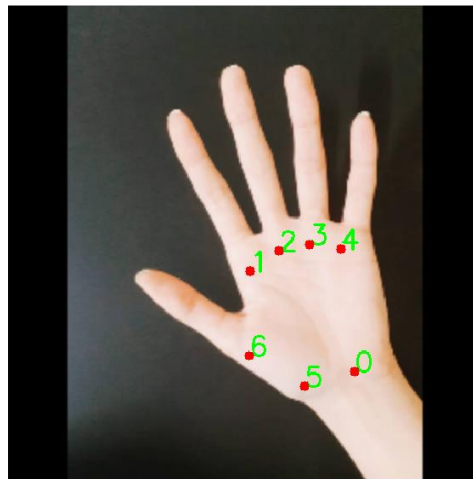
Nếu giá trị  $Z_t$  bằng 1, chúng ta sẽ giữ lại trạng thái cũ, nếu  $Z_t$  bằng 0 chúng ta sẽ cập nhật lại trạng thái mới.

Do đó việc sử dụng các cổng sẽ giúp nắm bắt các phụ thuộc ngắn hạn trong chuỗi thời gian khi sử dụng cổng reset, ngoài ra còn giúp nắm bắt các phụ thuộc dài hạn khi sử dụng cổng cập nhật.

### 1.3. THƯ VIỆN MEDIAPIPE HAND

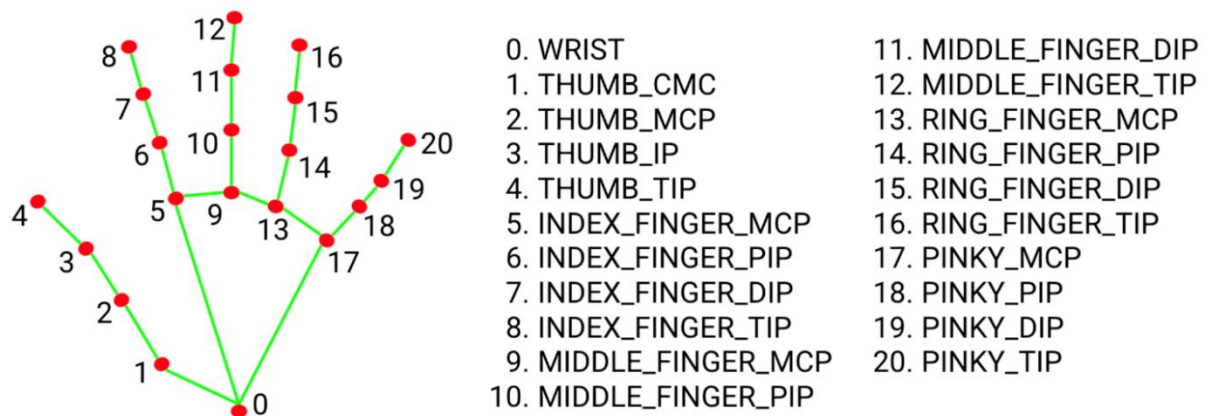
Mediapipe hand là một thư viện mã nguồn mở do Google phát triển. Thư viện cung cấp công cụ để thực hiện tác vụ phát hiện và đánh dấu 21 điểm trên tay thông qua hình ảnh. Công cụ có thể hoạt động trên dữ liệu hình ảnh tĩnh hoặc dòng hình ảnh liên tục và xuất ra tọa độ của 21 điểm đánh dấu tay trong hình ảnh.

Bên trong thư viện có hai mô hình sẽ hoạt động cùng lúc với nhau. Cùng một hình ảnh đầu vào, Palm detection model (mô hình phát hiện lòng bàn tay) sẽ trả về một giới hạn bao quanh bàn tay có 7 điểm trên lòng bàn tay. Mô hình có độ chính xác cao lên đến 95.7% [13]



Hình 15: Mô hình phát hiện lòng bàn tay.

Sau khi thực hiện quá trình detect lòng bàn tay, mô hình tìm các mốc bàn tay thông qua Hand Landmark Model sẽ giúp dự đoán 21 điểm mốc 3D của bộ xương bàn tay thông qua ảnh bên dưới [14].



**Hình 16: Hand Landmark Model.**

Với mỗi điểm trên bàn tay, mô hình sẽ trích xuất ra cho chúng ta 03 giá trị tọa độ  $x, y, z$  trong đó  $x, y$  là tọa độ của các điểm được đánh dấu trên bàn tay. Các giá trị  $x, y$  được chuẩn hoá trong khoảng  $[0,1]$  theo chiều rộng và chiều cao của hình ảnh.  $z$  đại diện cho độ sâu của điểm đánh dấu so với camera. Giá trị này được biểu thị của các điểm so với độ sâu ở cổ tay, giá trị càng nhỏ có nghĩa là mốc tại cổ tay càng gần với máy ảnh. Độ sâu của mốc cổ tay phải luôn bằng 0 và tất cả các mốc có giá trị độ sâu nhỏ hơn cổ tay sẽ có giá trị âm và các mốc có giá trị độ sâu lớn hơn cổ tay sẽ có giá trị  $z$  dương. Giá trị  $z$  được tính toán thông qua mô hình dự đoán là độ sâu tương đối hay relative depth[15] dựa trên  $Z_{avg}$ . Giá trị  $z$  sẽ sử dụng scale orthographic projection[15] hay phép chiếu trực giao tỷ lệ để có thể giữ các điểm ở vị trí xa không bị méo mó làm ảnh hưởng đến việc nhận diện và theo dõi. Cả ba giá trị  $x, y, z$  sẽ cung cấp vị trí tương đối của các khớp ngón tay và sẽ giúp cho việc nhận diện và theo dõi thuận lợi hơn.

Một điều đặc biệt khi sử dụng mô hình MediaPipe Hand là khả năng thực thi trong thời gian thực cực kì tốt và tốc độ rất cao.

## 1.4. THUẬT TOÁN TỐI ƯU

### 1.4.1. Thuật toán tối ưu là gì?

Trong suốt đề tài, việc sử dụng hàm mất mát và tối ưu hàm sẽ đem lại cho bài toán có độ chính xác tốt hơn. Hàm mất mát trong deep learning sẽ chính là mục tiêu chính khi chúng ta thực hiện quá trình tối ưu, đưa hàm mất mát có độ sai lệch thấp nhất. Đó chính là bài toán cực tiểu hoá.

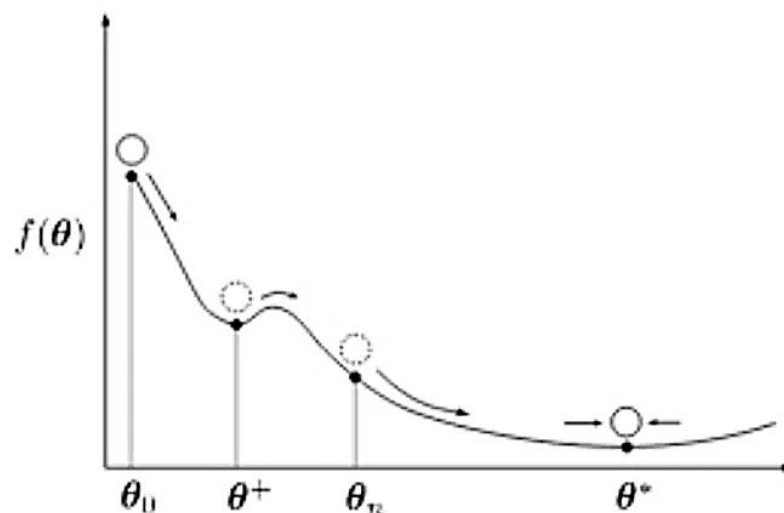
Mục đích khi thực hiện tối ưu là sẽ tìm được điểm cực tiểu của hàm đó khi thực hiện đạo hàm giúp lỗi của mô hình giảm xuống mức thấp nhất.

### 1.4.2. Giới thiệu thuật toán Adam

Diederik Kingma đã công bố thuật toán Adam[16] lần đầu tiên tại International Conference on Learning Representations vào năm 2015. Hiện tại, Adam là một trong những thuật toán tối ưu hóa mạnh mẽ nhất và được sử dụng rộng rãi trong cả nghiên cứu lẫn ứng dụng. Adam nổi bật với sự dễ dàng trong việc sử dụng và hiệu suất tính toán, làm cho nó trở thành lựa chọn lý tưởng cho các tập dữ liệu lớn và rời rạc. Thuật toán này cũng thường xuyên được sử dụng trong ngành thị giác máy tính. Không giống như SGD[17] (Stochastic Gradient Descent), một phương pháp tối ưu hóa hàm mục tiêu trong học sâu, Adam áp dụng các tốc độ học khác nhau cho từng tham số dựa trên các tham số  $\beta_1$  và  $\beta_2$ . Theo Diederik Kingma, Adam kết hợp được những ưu điểm của hai thuật toán tối ưu hóa phổ biến khác: Adaptive gradient algorithm và Root mean square propagation.

### 1.4.3. Nguyên tắc hoạt động của Adam

Thuật toán Adam hoạt động dựa vào hai khái niệm là động lượng và tốc độ đọc thích ứng. Tương tự như một quả bóng khi lăn xuống dốc, khi lăn xuống dốc quả bóng sẽ có động lượng. Khái niệm động lượng sẽ giúp tăng tốc thuật toán bằng cách điều hướng để hàm mất mát có thể vượt qua được điểm cực tiểu địa phương và có thể tới được điểm cực tiểu toàn bộ. Ngoài ra, khi đã đến được điểm cực tiểu toàn bộ, adam sẽ thực hiện điều chỉnh lại giá trị learning rate để có thể giúp điểm mất mát sẽ càng gần so với điểm cực tiểu.



Hình 17: Mô phỏng thuật toán Adam

Adam được thiết kế cho việc học sâu và thể hiện được ưu thế so với các thuật toán tối ưu khác. Adam không chỉ dễ dàng trong việc triển khai, độ phức tạp hiệu quả, không

yêu cầu bộ nhớ lớn, thích hợp với nhiều bài toán có nhiều kiểu dữ liệu khác nhau mà Adam còn có khả năng xử lý độ dốc nhiều. Độ dốc nhiều xảy ra khi việc lấy mẫu để tính không đại diện cho toàn bộ dữ liệu. Việc này khi xảy ra sẽ mang lại nhiều sự không ổn định trong quá trình học nhưng Adam giải quyết rất tốt.

#### 1.4.4. Công thức

Công thức cơ bản của thuật toán tối ưu Adam bao gồm:

$$v_t = \beta_1 * v_{t-1} + (1 - \beta_1) * \text{gradient}(x)[18]$$

$$s_t = \beta_2 * s_{t-1} + (1 - \beta_2) * \text{gradient}(x)^2[18]$$

Trong đó  $\beta_1$  và  $\beta_2$  là các thông số first momentum of gradient và second momentum of gradient với giá trị là  $\beta_1=0.9$  và  $\beta_2=0.999$ .

#### 1.4.5. Ưu và nhược điểm của thuật toán Adam

Thuật toán Adam có nhiều ưu điểm cũng như nhược điểm như sau:

- Về ưu điểm:

- Hiệu suất cao: Adam thường hoạt động hiệu quả trên các bài toán như thị giác máy tính, học máy, ...

- Khả năng tự điều chỉnh learning rate: Việc tự điều chỉnh giá trị learning rate cho từng tham số riêng lẻ giúp tốc độ học của mô hình giảm đáng kể và đem lại kết quả tốt.

- Khả năng ổn định: Việc Adam có khả năng lọc các giá trị nhiễu cho nên thuật toán sẽ giảm sự ảnh hưởng của outlier cũng như giúp thuật toán ổn định.

- Dễ dàng cài đặt và sử dụng: Hầu hết các thông số, thuật toán sẽ tự điều chỉnh cho nên việc chỉnh sửa thông số không nhất thiết phải tự cài đặt.

- Về nhược điểm:

Khi learning rate quá lớn hoặc quá nhỏ, việc này thì Adam có thể không hội tụ hoặc hội tụ chậm trong một vài trường hợp. Nếu tốc độ học quá lớn, thuật toán sẽ không thể hội tụ được, còn nếu quá nhỏ, mô hình khi training cực kì lâu.

### 1.5. TỐI ƯU HOÁ MÔ HÌNH

Sau khi thực hiện training mô hình, chúng ta có thể thực hiện quá trình tối ưu hoá mô hình. Việc tối ưu hoá mô hình có nhiều ưu điểm:

- Giảm kích cỡ: Khi mô hình nhỏ đi, lưu trữ sẽ đỡ tốn kém bộ nhớ hơn khi sử dụng.

- Tăng tính tương thích: Một số thiết bị đầu cuối sẽ hoạt động trơn tru hơn nếu được tối ưu đúng các thông số.

- Giảm độ trễ: Thời gian inference sẽ nhỏ đi khi khối lượng tính toán giảm bớt.
- Với nhiều ưu điểm trên, chúng ta cần phải đánh đổi độ chính xác của mô hình.
- Các phương pháp có thể dùng để tối ưu hoá mô hình bao gồm:
- Lượng tử hoá mô hình[19]: Sẽ thực hiện giảm không gian lưu trữ của mô hình.
  - Tỉa mô hình[20]: Sẽ đặt ra các ngưỡng, nếu các tham số nhỏ hơn ngưỡng sẽ loại khỏi mô hình.
  - Nén mô hình[21]: Sử dụng các thuật toán nén mô hình lại tuy nhiên khi thực hiện sử dụng mô hình chúng ta phải giải nén mô hình.
  - Phân cụm tham số[22]: Sử dụng các thuật toán để phân cụm các thông số của mô hình ra.
  - Chắt lọc tri thức: Sử dụng mô hình nhỏ hơn học kết quả của các mô hình lớn để đưa ra một mô hình nhỏ hơn nhưng kết quả tốt hơn.

Trong luận văn, sẽ sử dụng phương pháp lượng tử hoá mô hình để có thể giảm biến đổi trên đồ thị tính toán của mô hình và giảm kích thước của mô hình. Chúng sẽ thực hiện loại bỏ các nút không cần thiết, gộp các lớp lại với nhau và thực hiện tối ưu hoá hiệu suất.

## 1.6. CÁC PHƯƠNG PHÁP ĐÁNH GIÁ

Trong các bài toán liên quan đến việc xây dựng mô hình trong mảng machine learning cũng như deep learning, chúng ta cần phải thực hiện quá trình đánh giá mô hình sau khi training. Dưới đây là một số phương pháp được sử dụng trong khoá luận:

- True Negative (TN): Một ký tự được dự đoán không phải là ký hiệu đó và đã xác định đúng là như thế.
- True Positive (TP): Là một ký tự được dự đoán đúng là ký tự đó.
- False Negative (FN): Là một ký tự được dự đoán nhưng không phải đúng ký tự đó.
- False Positive (FP): Một ký tự tuy không phải là ký tự chính xác nhưng được nhận diện là ký tự.
- Accuracy được tính toán dựa trên tỉ lệ các mẫu dự đoán là đúng và tổng số mẫu được đưa vào tính toán. Accuracy được tính toán dựa trên công thức:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

- Precision là một thông số xác định độ tin cậy của mô hình thông qua các mẫu dữ liệu dự đoán đúng.

$$\text{Precision} = \frac{TP}{TP + FP}$$

- Recall là phương pháp đại diện độ nhạy của mô hình thông qua tỷ lệ số điểm TP trong số những được thực sự là positive.

$$\text{Recall} = \frac{TP}{TP + FN}$$

- F1 Score: Là giá trị trung bình điều hoà giữa Precision và Recall.

$$\text{F1 Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

## CHƯƠNG 2: ỨNG DỤNG VÀO NHẬN DIỆN THỦ NGỮ BẰNG CỬ CHỈ

### 2.1. GIỚI THIỆU VỀ TENSORFLOW

Tensorflow là một thư viện mã nguồn mở được phát triển bởi Google và được sử dụng rộng rãi trong lĩnh vực Machine Learning và Deep Learning. Chúng được thiết kế để hỗ trợ xây dựng và huấn luyện mô hình Deep Learning. Chúng cung cấp cho chúng ta các công cụ và API cho việc xây dựng các mô hình trở nên đơn giản hơn.



Hình 18: Tensorflow

Các công cụ chính của Tensorflow bao gồm: đồ thị tính toán, tensor, session, API và các công cụ khác. TensorFlow được ứng dụng trong nhiều ứng dụng đa dạng nhiều lĩnh vực khác nhau từ Computer Vision, NLP, ... và được áp dụng vào các hệ thống như xe tự hành, y học cũng như các hệ thống dự đoán sở thích người dùng.

### 2.2. THU THẬP DỮ LIỆU VÀ XÂY DỰNG MÔ HÌNH

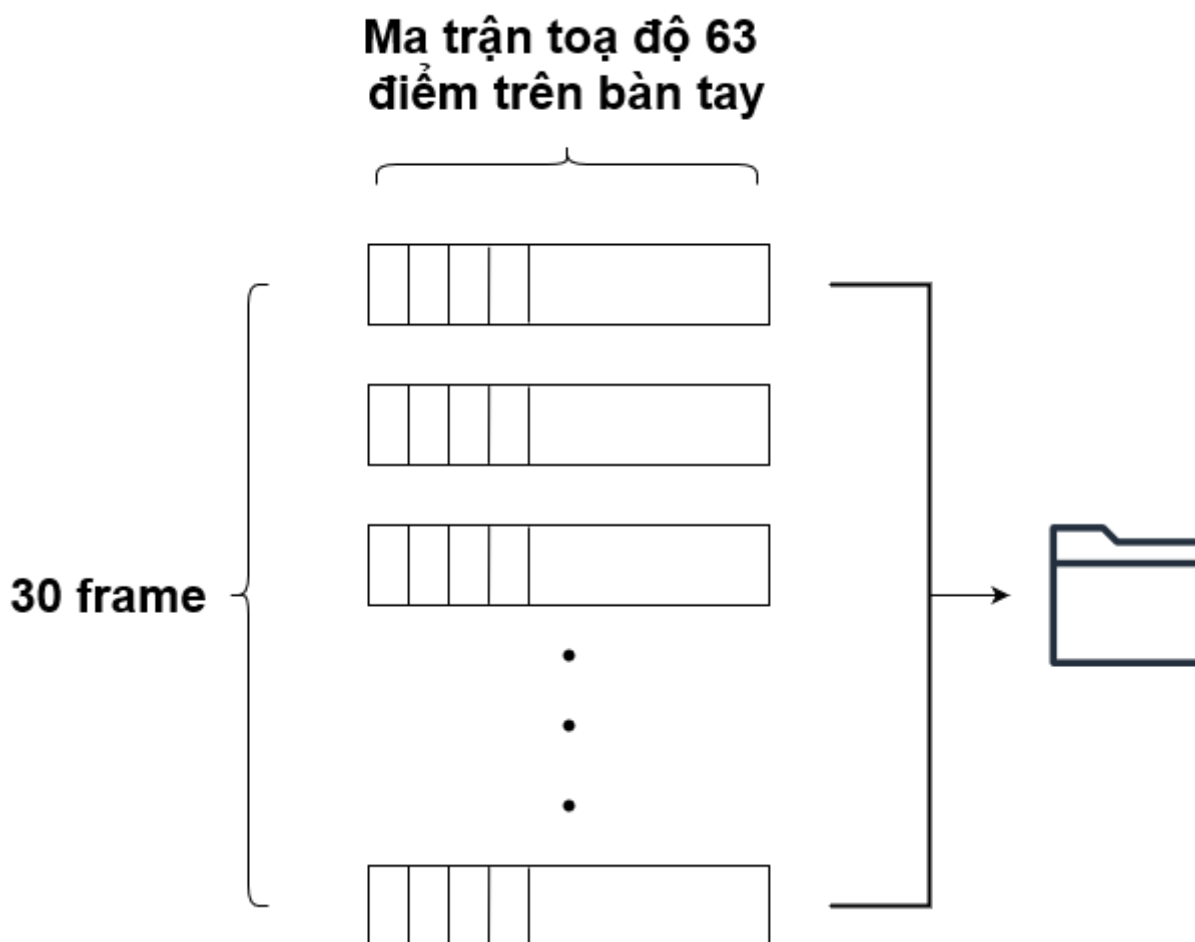
#### 2.2.1. Giới thiệu

Trong bài toán nhận diện thủ ngữ, dữ liệu là một phần cực kỳ quan trọng. Dữ liệu được thu thập từ camera máy tính với kích thước ảnh đầu vào là 640x480. Tuy nhiên, thay vì việc sử dụng hình ảnh để lưu trữ và training dữ liệu, em sử dụng các thông số  $x$ ,  $y$ ,  $z$ .

Dữ liệu cho nhận diện cử chỉ động của 16 hành động thủ ngữ bao gồm 11 ký tự thủ ngữ: “nothing”, “a”, “aa”, “aw”, “e”, “ee”, “o”, “oo”, “ow”, “u”, “uw” và 06 hành động về các thanh dấu bao gồm: “sac”, “hoi”, “nga”, “nang”.



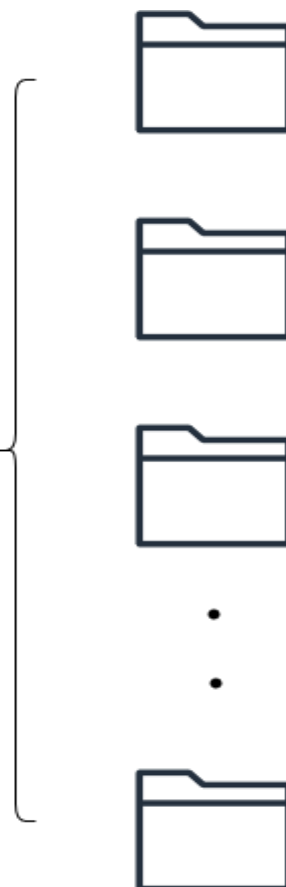
Thông qua thư viện OpenCV, thư viện MediaPipe sẽ thực hiện trích xuất tọa độ 3D của các điểm trên bàn tay. Quá trình thu được thực hiện tự động, dữ liệu trả về sẽ được diễn tả bằng sơ đồ bên dưới.



Hình 19: Cấu trúc bộ dữ liệu được thu thập

Trong 30 frame hình ảnh, mỗi frame sẽ thu được 63 điểm tọa độ khác nhau. Mỗi ma trận sẽ được lưu vào 01 file khác nhau sau đó tổng hợp lại thành một folder.

**Một kí tự sẽ gồm  
100 folder khác nhau**



**Hình 20: Cấu trúc của dữ liệu sau khi tổng hợp xong**

Sau khi thực hiện 100 lần ký hiệu thủ ngữ về dữ liệu, chúng ta sẽ có được 100 folder, mỗi folder sẽ chứa 30 file dữ liệu, mỗi file dữ liệu sẽ chứa 63 điểm tọa độ các điểm trên bàn tay. Một số mẫu dữ liệu được thu thập theo như hướng dẫn của bảng ngôn ngữ ký hiệu dành cho Người điếc và Người khiếm thính Việt Nam.

1 A	2 B	3 C	4 D	5 Đ	6 E
7 G	8 H	9 I	10 K	11 L	12 M
13 N	14 O	15 P	16 Q	17 R	18 S
19 T	20 U	21 V	22 X	23 Y	24 DẤU THANH GIẾC
25 DẤU RÁU	26 DẤU MỖ	A + DẤU RÁU = Ä	A + DẤU RÁU = Ä	E + DẤU MỖ = Ê	U + DẤU RÁU = Û
O + DẤU RÁU = Ơ	O + DẤU MỖ = Ô				

VÍ DỤ: TÔI 19+14+26+9 11+1+24 13+8+13+7

TÔI LÀ NHUNG

**Hình 21: Bảng kí hiệu ngôn ngữ Tiếng Việt.**

### 2.2.2. Xây dựng mô hình

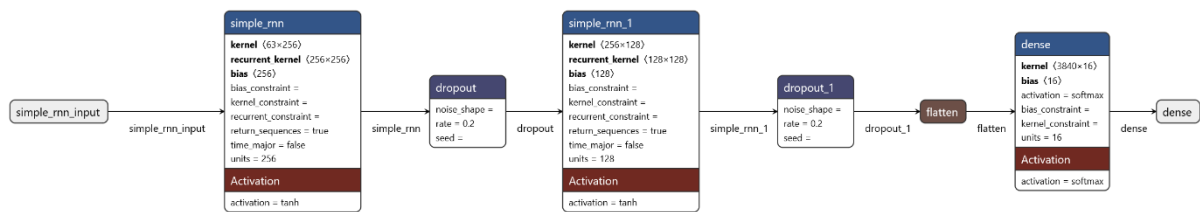
Bài toán sẽ thực hiện năm mô hình bao gồm các mô hình sau. Tất cả các mô hình đều có đầu vào có shape là 63x30. Trong đó 63 là số điểm dữ liệu trên một file dữ liệu, 30 là số file mà mỗi folder chứa.

Đồng thời quá trình training sẽ thực hiện trên máy tính không có GPU.

Toàn bộ quá trình training sẽ được tối ưu bằng thuật toán Adam và sai số chính là sai số toàn phương trung bình Mean Square Error.

#### 2.2.2.1. Mô hình RNN

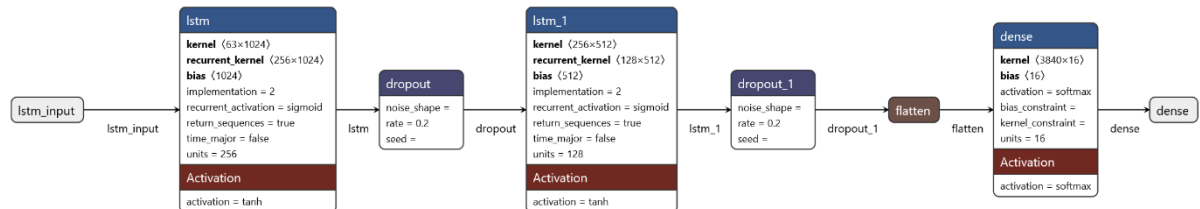
Mô hình được xây dựng như ảnh dưới:



Hình 22: Kiến trúc mô hình RNN

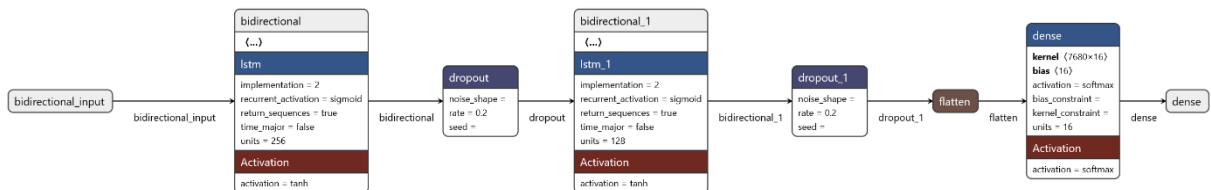
#### 2.2.2.2. Mô hình LSTM

Đối với mô hình LSTM sẽ được xây dựng tương tự như mô hình RNN.



Hình 23: Kiến trúc mô hình LSTM

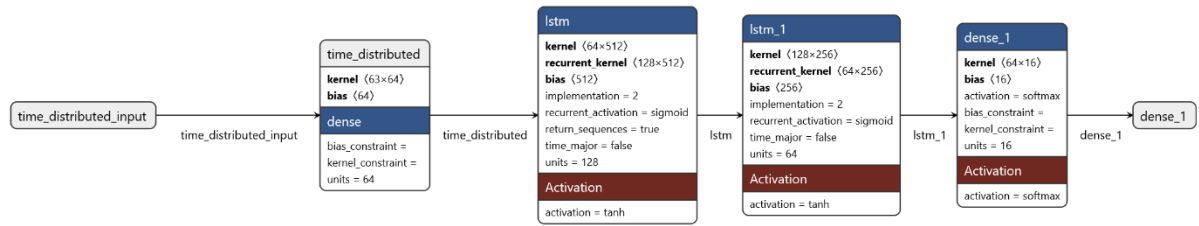
#### 2.2.2.3. Mô hình Bidirectional LSTM



Hình 24: Kiến trúc mô hình Bidirectional LSTM

#### 2.2.2.4. Mô hình Hierarchical Model

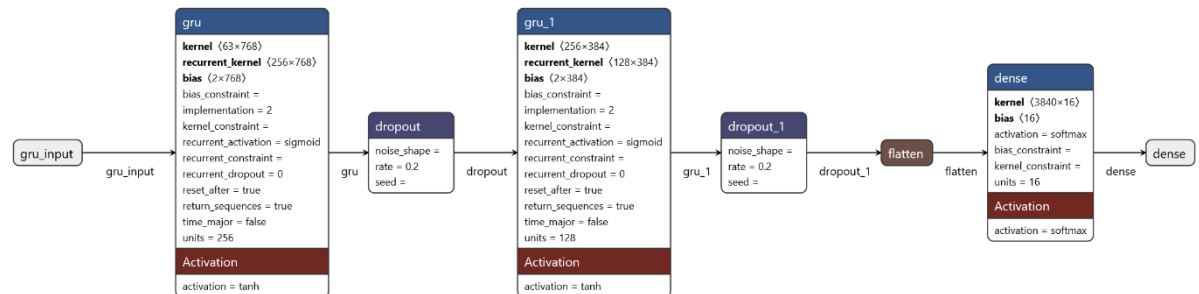
Mô hình được xây dựng như ảnh bên dưới:



Hình 25: Kiến trúc Hierarchical Model

#### 2.2.2.5. Mô hình GRU

Mô hình GRU được xây dựng như sau:

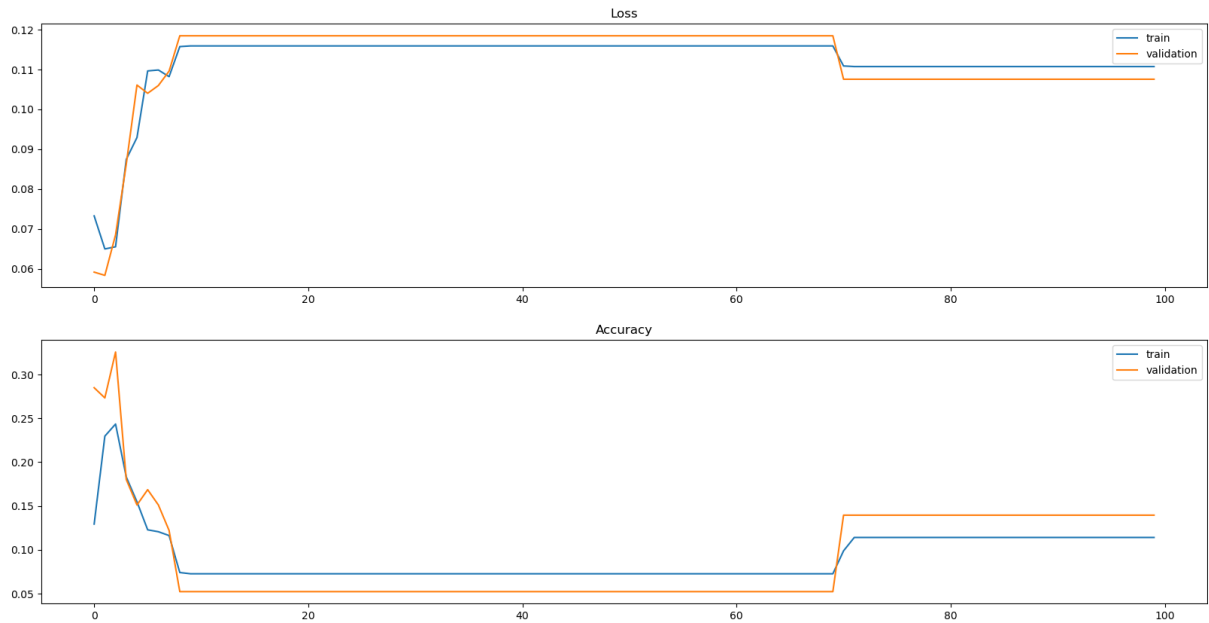


Hình 26: Kiến trúc mô hình GRU

### 2.3. KẾT QUẢ MÔ HÌNH

Tất cả các mô hình đều sử dụng chung thuật toán tối ưu Adam, độ sai lệch đều sử dụng sai số toàn phương trung bình cũng như đều training trên dữ liệu 100 epochs. Chúng ta sẽ có những kết quả sau đây.

### 2.3.1. Mô hình RNN



Hình 27: Đồ thị biểu thị độ chính xác và độ sai số của mô hình RNN

Khi nhìn vào biểu đồ bên trên, có thể thấy trong giai đoạn đầu, độ chính xác cũng như độ sai lệch tăng nhanh và sau đó ổn định. Tuy nhiên, độ chính xác chỉ đạt được 11.41% do đó khả năng dự đoán của mô hình không cao. Ngoài ra, khi nhìn vào đồ thị sai lệch, việc độ sai lệch tăng thay vì giảm.

Sẽ có nhiều vấn đề xảy ra khi sử dụng mô hình RNN như dữ liệu chưa đa dạng, mẫu chọn chưa đại diện đúng cho toàn bộ tập dữ liệu. Tuy nhiên, khi đưa sang tập dữ liệu testing ta có các thông số sau.

		Confusion Matrix															
True	nothing	8	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
	a	0	0	0	0	0	0	0	0	7	0	0	0	0	0	0	0
	aa	0	0	0	0	0	0	0	0	15	0	0	0	0	0	0	0
	aw	0	0	0	0	0	0	0	0	18	0	0	0	0	0	0	0
	e	0	0	0	0	0	0	0	0	9	0	0	0	0	0	0	0
	ee	0	0	0	0	0	0	0	0	10	0	0	0	0	0	0	0
	u	0	0	0	0	0	0	0	0	8	0	0	0	0	0	0	0
	uw	0	0	0	0	0	0	0	0	10	0	0	0	0	0	0	0
	o	0	0	0	0	0	0	0	0	6	0	0	0	0	0	0	0
	oo	0	0	0	0	0	0	0	0	12	0	0	0	0	0	0	0
	ow	0	0	0	0	0	0	0	0	11	0	0	0	0	0	0	0
	sac	0	0	0	0	0	0	0	0	6	0	0	0	0	0	0	0
	hoi	0	0	0	0	0	0	0	0	14	0	0	0	0	0	0	0
	huyen	0	0	0	0	0	0	0	0	10	0	0	0	0	0	0	0
	nang	0	0	0	0	0	0	0	0	14	0	0	0	0	0	0	0
	nga	0	0	0	0	0	0	0	0	13	0	0	0	0	0	0	0
		nothing	a	aa	aw	e	ee	u	uw	o	oo	ow	sac	hoi	huyen	nang	nga
		Predicted															

Hình 28: Confusion Matrix của mô hình RNN

Và

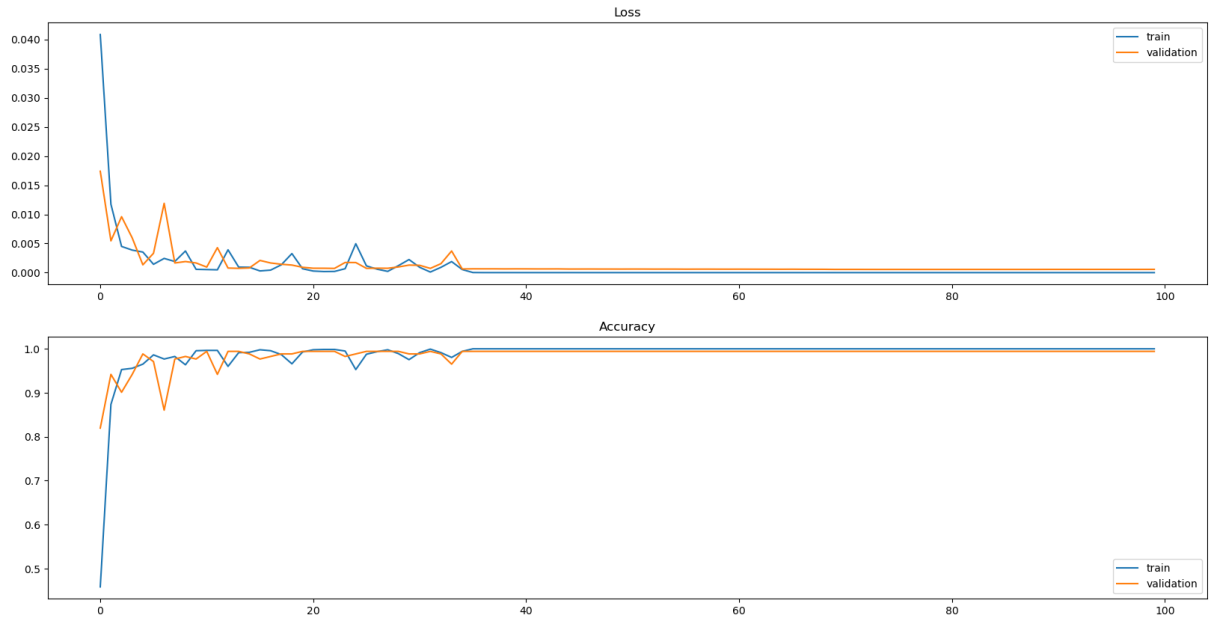
Bảng 3: Bảng thông số của mô hình RNN

Thông số	Giá trị
Accuracy	0.081395
Precision	0.064787
Recall	0.118056
F1-Score	0.063235
ROC AUC	0.487194

Dựa vào hai bảng trên có thể rút ra được mô hình RNN không khả thi trong trường hợp này. Trừ dữ liệu “nothing” và “o”, toàn bộ các kí tự khác đều dự đoán sai.

Các thông số khi thực hiện xây dựng mô hình bị underfitting và không thể tăng lên cao nữa do đó việc sử dụng một mô hình phức tạp hơn sẽ là một phương pháp tốt hơn để có thể tối ưu mô hình.

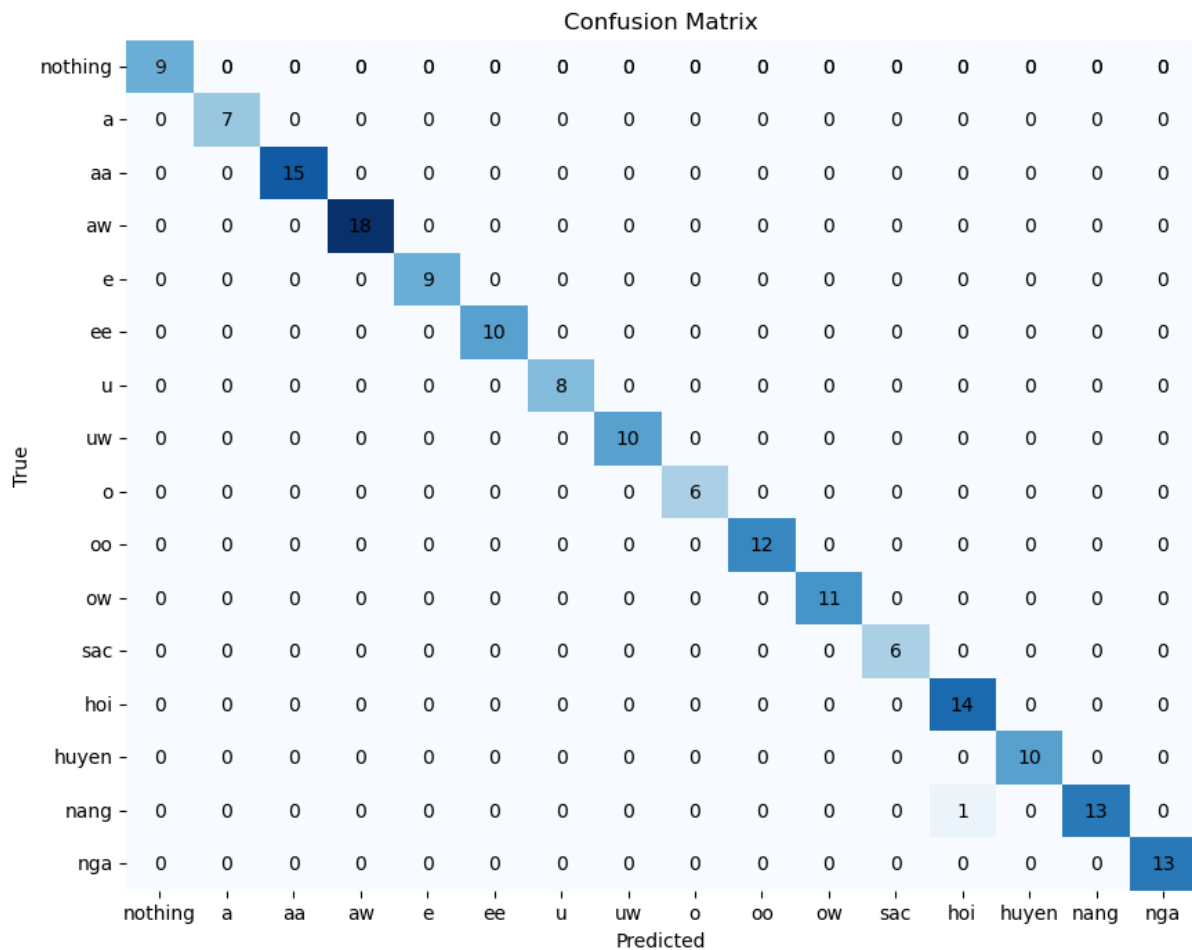
### 2.3.2. Mô hình LSTM



Hình 29: Đồ thị biểu thị độ chính xác và độ sai số của mô hình RNN

Với biểu đồ độ sai lệch, sai số giảm nhanh chóng và ổn định sau khoảng 20 epochs. Điều này cho thấy mô hình đang học tốt dữ liệu và dự đoán chính xác dữ liệu theo thời gian. Với biểu đồ độ chính xác, biểu đồ thể hiện độ chính xác của mô hình qua từng epoch. Mô hình dần dần đạt được độ chính xác cao và duy trì ở mức ổn định. Do đó, có thể cho chúng ta thấy việc áp dụng mô hình vào dự đoán sẽ là một quyết định hợp lý.

Tuy nhiên để cho chắc chắn, sẽ tiến hành xét thêm các thông số được biểu thị ở bảng và hình bên dưới.



**Hình 30: Confusion Matrix của mô hình LSTM**

Và

**Bảng 4: Bảng thông số của mô hình LSTM**

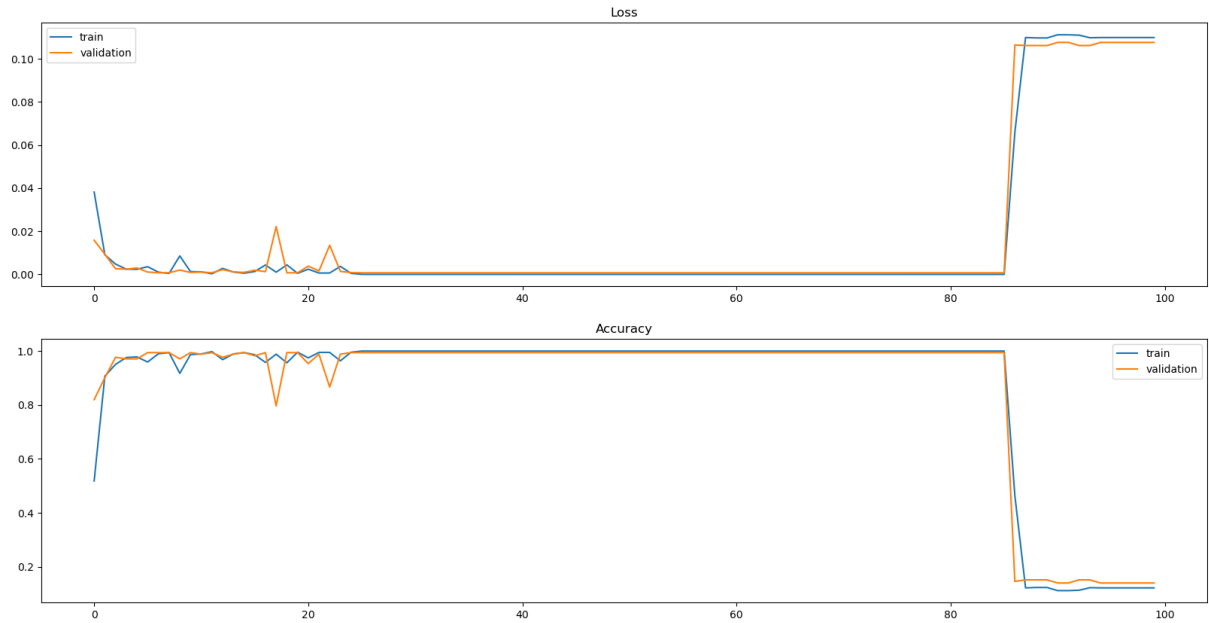
Thông số	Giá trị
Accuracy	0.994186
Precision	0.995833
Recall	0.995536
F1-Score	0.995530
ROC AUC	1.000000

Dựa vào confusion matrix, đường chéo được tô màu xanh cho thấy số lần dự đoán chính xác, có nghĩa là mô hình đã đạt được hiệu suất tốt vì hầu hết các dự đoán đều nằm trên đường chéo này.

Tuy nhiên vào một số trường hợp, mô hình đưa ra các dự đoán chưa chính xác. Điều đó thể hiện rõ trên bảng thông số bên trên. Với độ chính xác loanh quanh 99.5% nên việc sử dụng mô hình để dự đoán là khả thi trong đề tài.



### 2.3.3. Mô hình Bidirectional LSTM

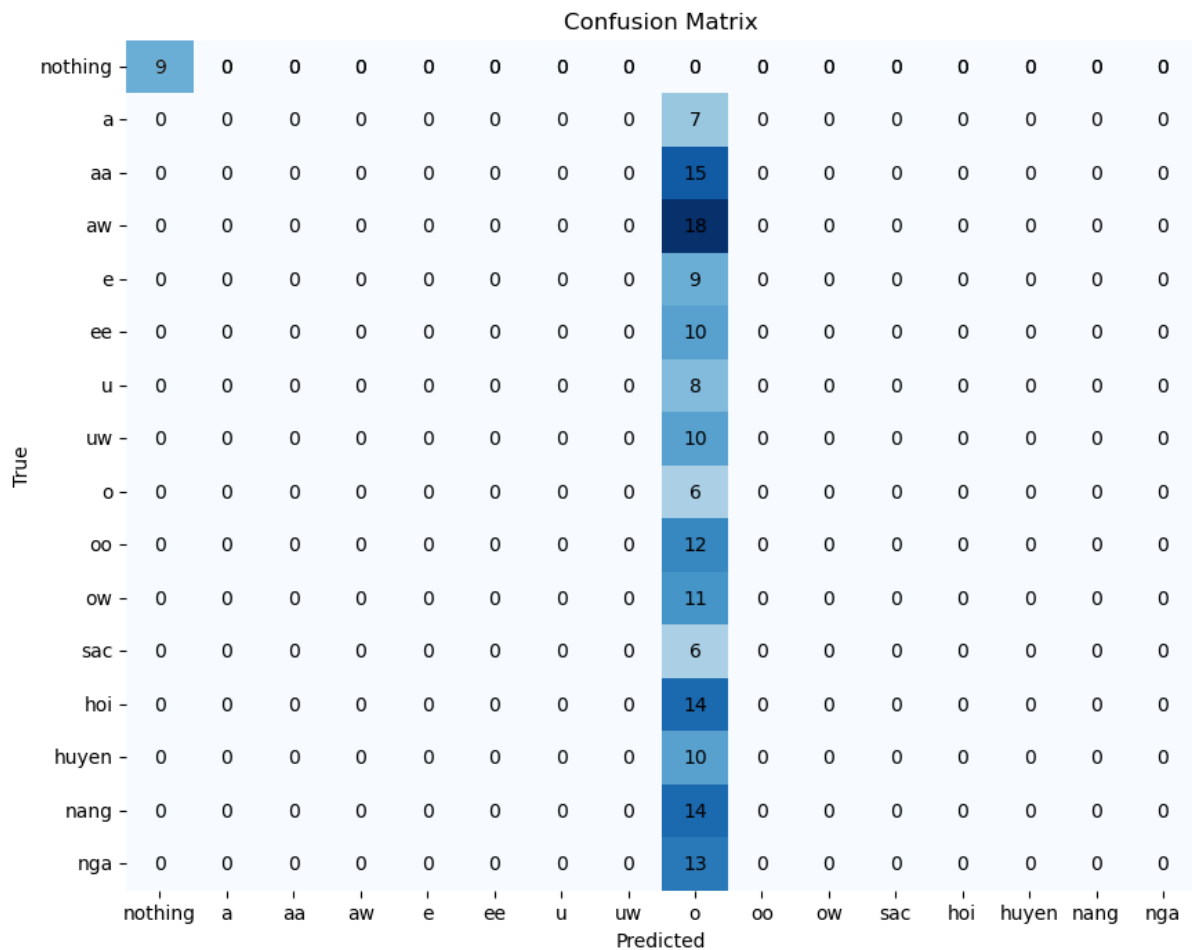


Hình 31: Đồ thị biểu thị độ chính xác và độ sai số của mô hình Bidirectional LSTM

Dựa vào hình bên trên, độ chính xác của mô hình bắt đầu tăng dần trong những epochs đầu tiên, độ sai lệch có giảm ở giai đoạn đầu tiên. Nhưng trong khoảng epochs 30, độ chính xác đạt cực đại, chứng tỏ mô hình bắt đầu học được những thông tin quan trọng.

Tuy nhiên, giai đoạn từ epochs thứ 80, độ chính xác đột ngột giảm xuống, độ sai lệch bắt đầu tăng lên, đồng nghĩa với việc mô hình đã bắt đầu bị vanishing gradient.

Tiếp tục xét đến các thông số khác của mô hình.



Hình 32: Confusion matrix của Bidirectional LSTM

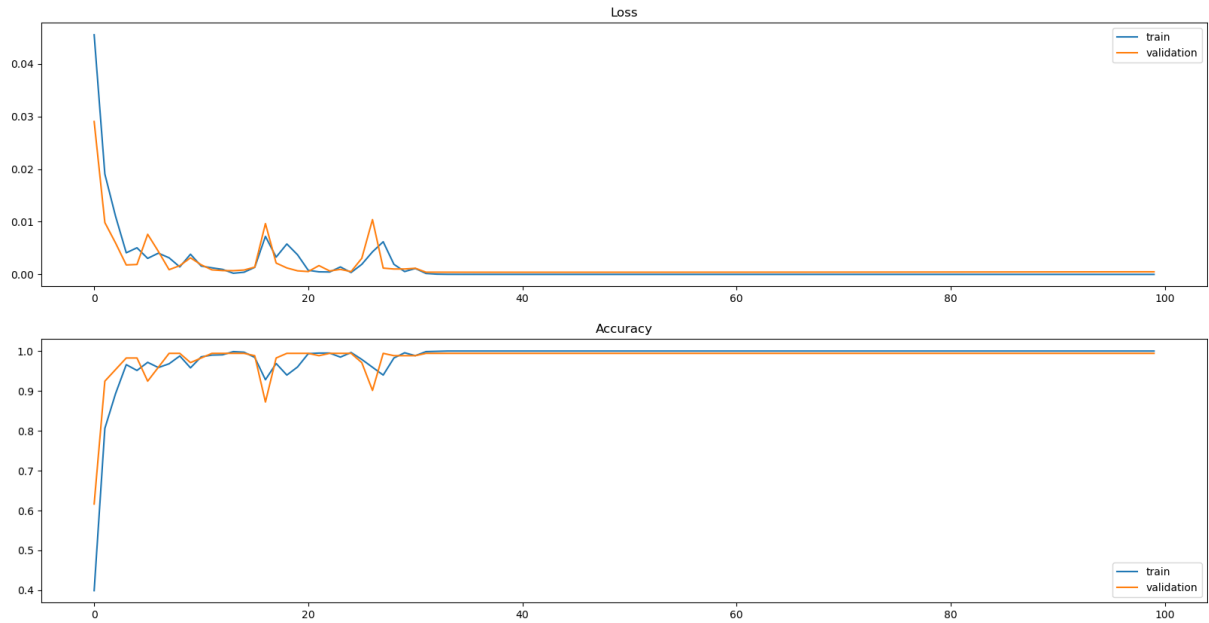
Và

Bảng 5: Bảng thông số của mô hình Bidirectional LSTM

Thông số	Giá trị
Accuracy	0.087209
Precision	0.064801
Recall	0.125000
F1-Score	0.066938
ROC AUC	0.530658

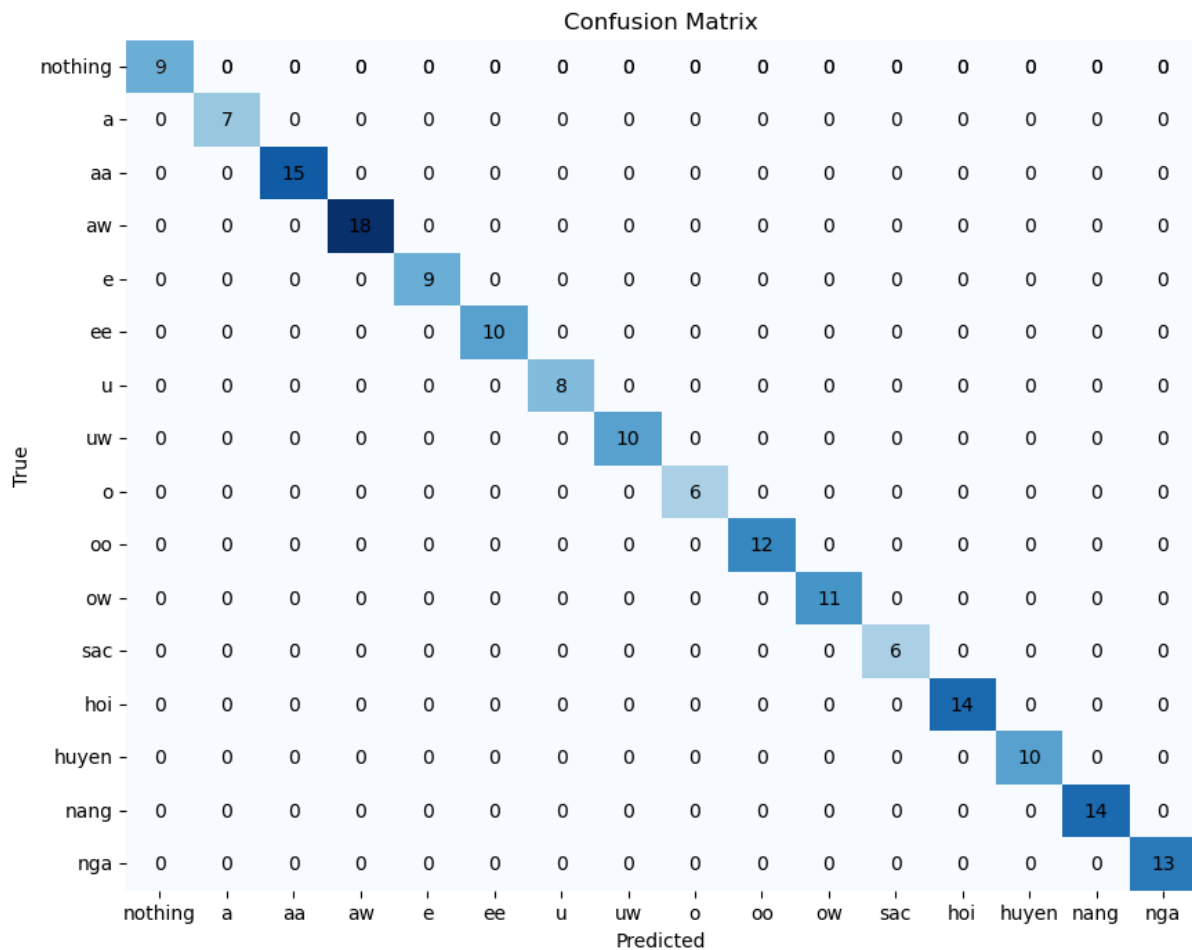
Tại đây, dựa vào các thông số ở confusion matrix cũng như bảng thông số trên, độ chính xác giảm cực kỳ mạnh tại những epochs cuối. Vanising gradient đã làm độ chính xác của mô hình giảm mạnh mặc dù thông số của mô hình rất lớn. Với độ chính xác như trên, việc áp dụng trong thực tế không khả thi.

### 2.3.4. Mô hình Hierachical Model



Hình 33: Đồ thị biểu thị độ chính xác và độ sai số của Hierachical Model

Dựa vào biểu đồ trên, độ chính xác cực kỳ cao cũng như độ sai lệch cực kỳ thấp. Tại những epochs đầu tiên, độ chính xác của mô hình từ bắt đầu tăng lên cao và ổn định hơn, mặc dù có những điểm epochs độ chính xác không ổn định. Tuy nhiên, để có thể nhìn rõ hơn, chúng ta sẽ xem xét các trường hợp dưới đây.



Hình 34: Confusion Matrix của Hierarchical Model

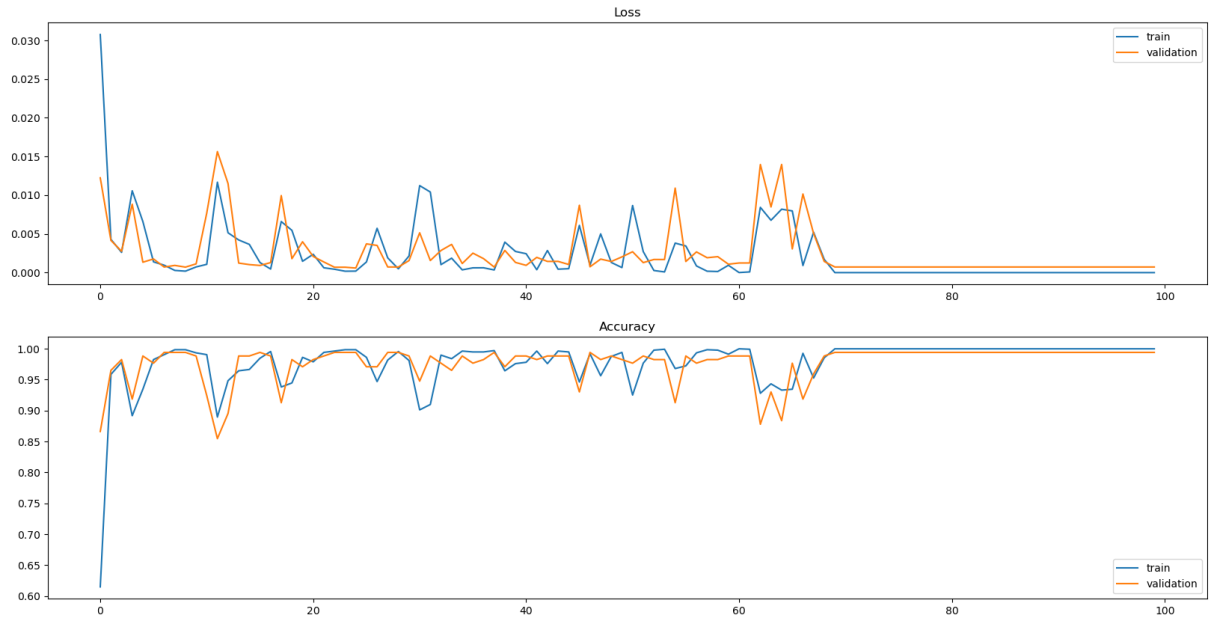
Và

Bảng 6: Bảng thông số của Hierarchical Model

Thông số	Giá trị
Accuracy	1.000000
Precision	1.000000
Recall	1.000000
F1-Score	1.000000
ROC AUC	1.000000

Độ chính xác của mô hình đều đạt 100%, chứng tỏ mô hình bị overfitting, khi đó ta sẽ xem xét thêm độ phức tạp của mô hình cũng không đủ để thực hiện. Với bộ dữ liệu test đều đúng hết 100%. Do đó, việc sử dụng mô hình sẽ khả thi trong đề tài.

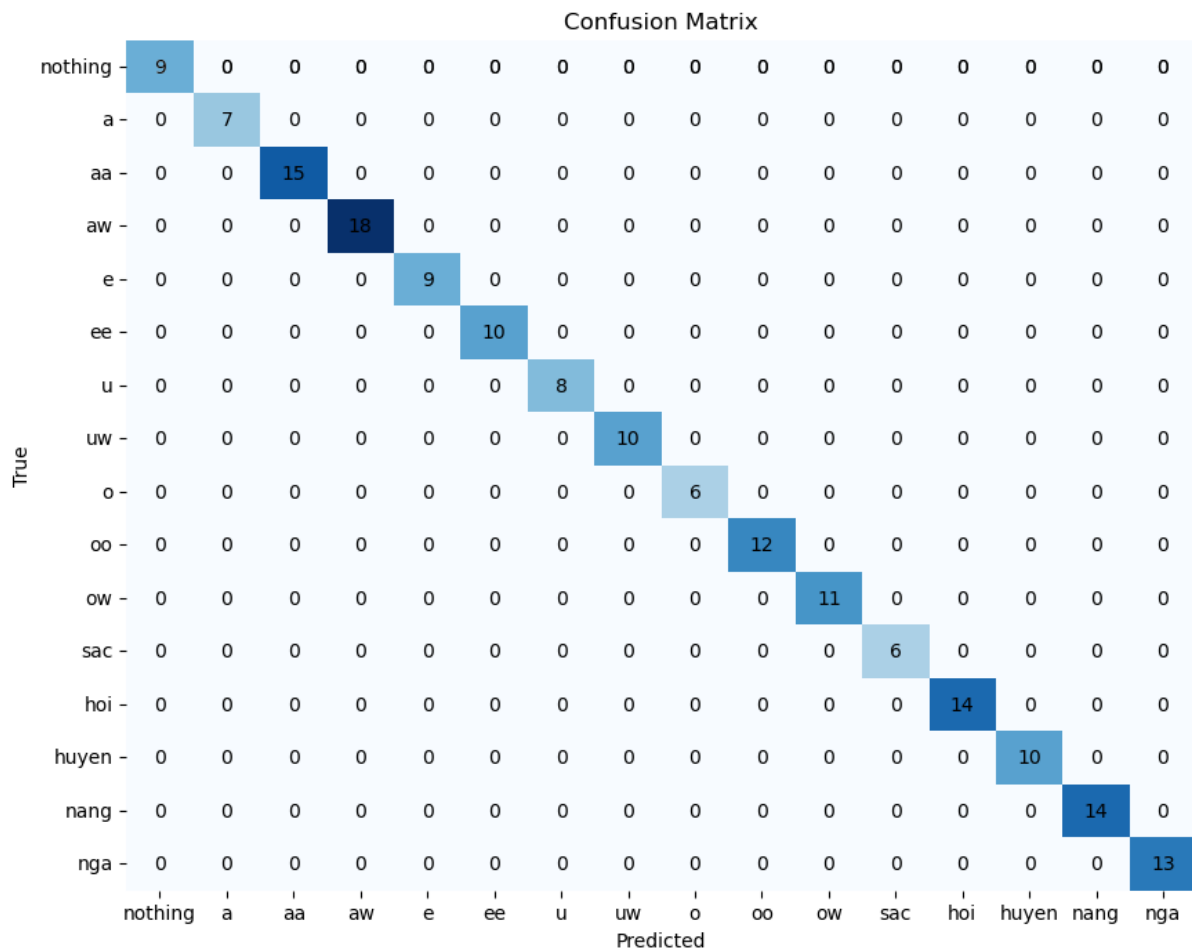
### 2.3.5. Mô hình GRU



Hình 35: Đồ thị biểu thị độ chính xác và độ sai số của mô hình GRU

Dựa vào biểu đồ trên, mô hình tăng mạnh độ chính xác trong những epochs đầu tiên, nhưng trong suốt khoảng từ epochs từ 0 đến epochs thứ 70, độ chính xác cũng như sai lệch không ổn định.

Tuy nhiên, bắt đầu từ những epochs thứ 70 trở đi, độ chính xác cũng như sai số ổn định ở mức cao nhất. Để bảo quát hơn, chúng ta sẽ xem xét các thông số tiếp theo.



Hình 36: Confusion matrix của mô hình GRU

Và

Bảng 7: Bảng thông số của mô hình GRU

Thông số	Giá trị
Accuracy	1.000000
Precision	1.000000
Recall	1.000000
F1-Score	1.000000
ROC AUC	1.000000

Dựa vào confusion matrix cũng như bảng thông số đều cho kết quả đúng 100% trên bộ dữ liệu testing. Do đó việc áp dụng mô hình trong thực tế khả thi.

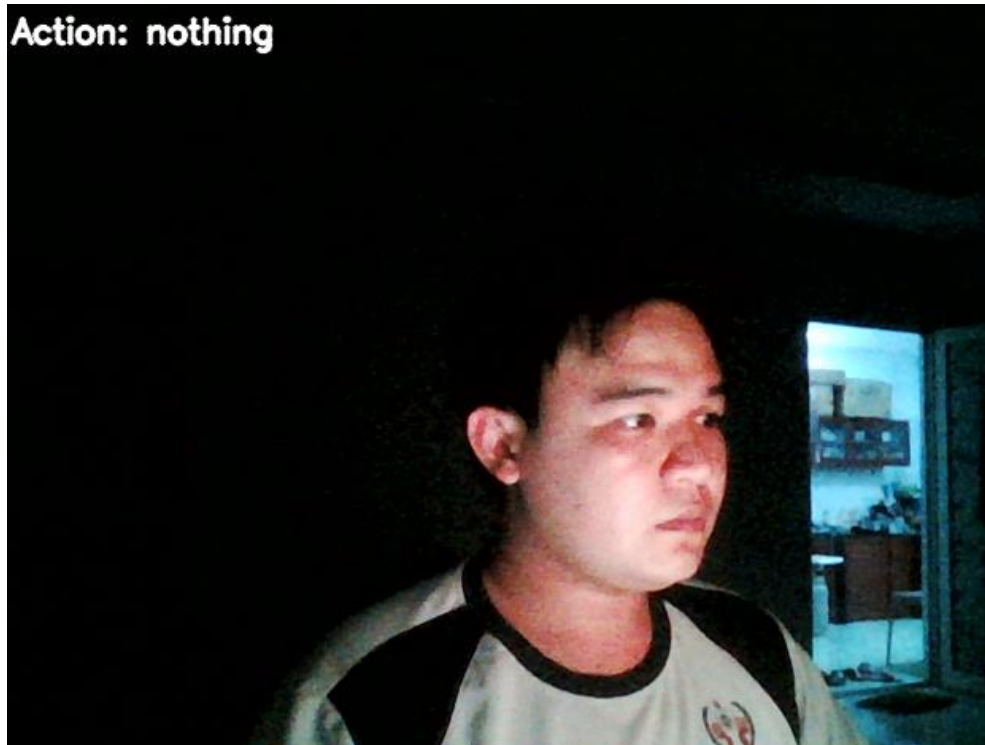
## 2.4. LỰA CHỌN MÔ HÌNH

Dựa vào các thông số cũng như kết quả bên trên, với bộ dữ liệu của đề tài, ta có thể sử dụng ba mô hình là LSTM, Bidirectional LSTM và GRU để áp dụng trong thực tế.

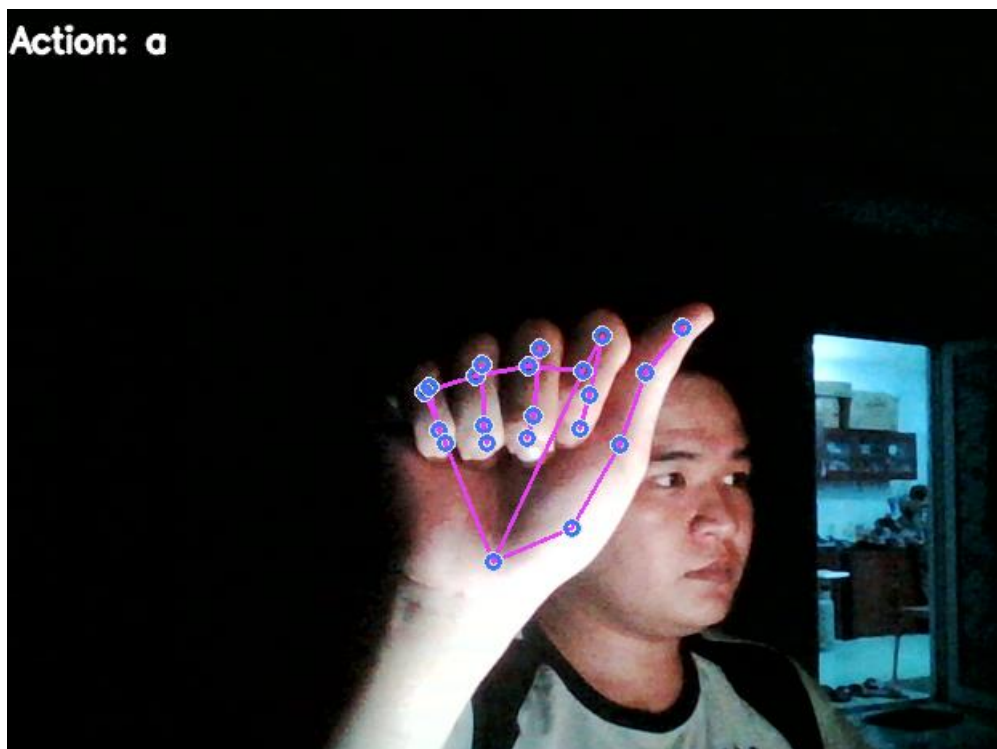
Sau khi thực hiện lưu lại mô hình, để đáp ứng cho việc sử dụng trên máy tính cá nhân không có sử dụng GPU, tác giả thực hiện quá trình lượng tử hoá mô hình để có thể giảm

sự biến đổi trên đồ thị tính toán của mô hình và giảm kích thước của mô hình bằng Tensorflow.

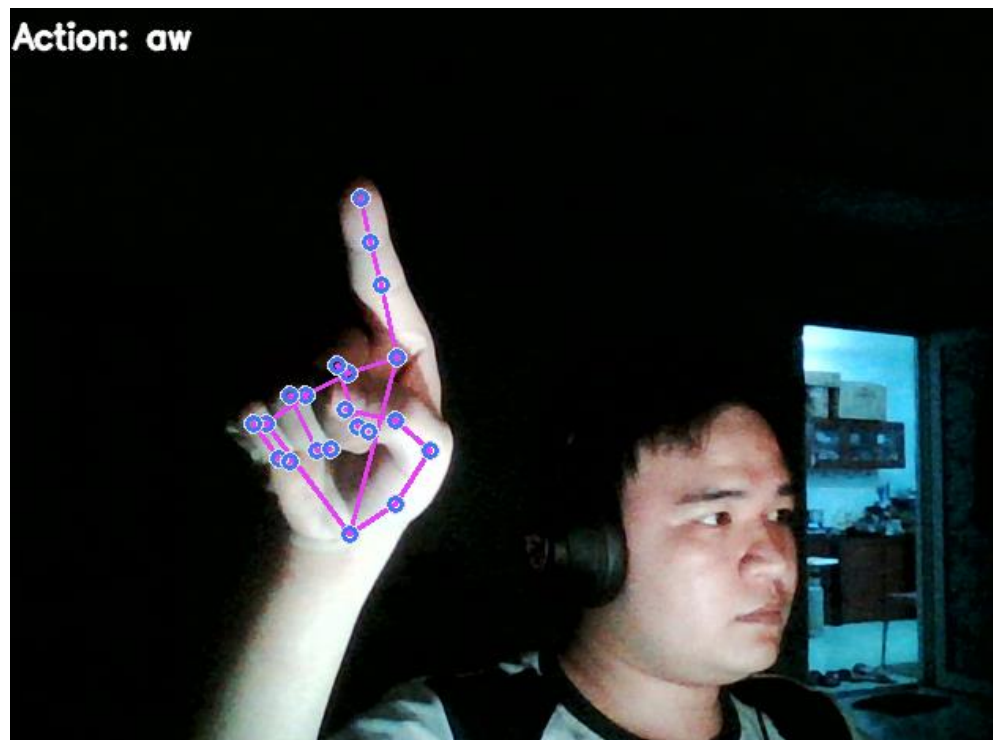
## 2.5. KẾT QUẢ THỰC TẾ



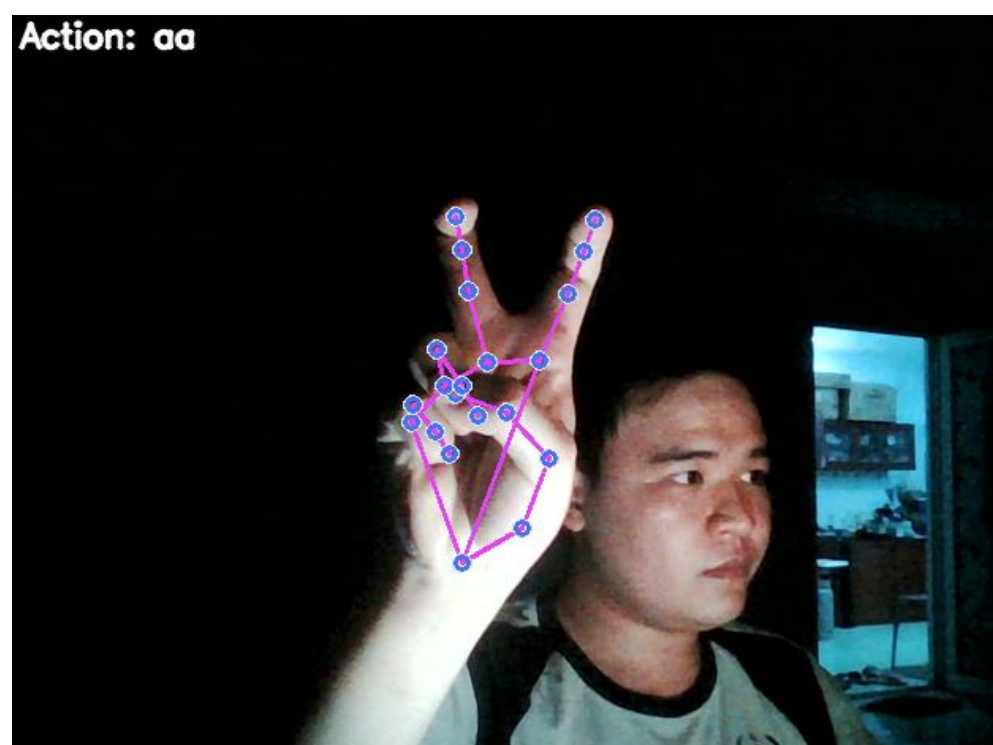
Hình 37: Nhận biết nothing



Hình 38: Nhận biết chữ a

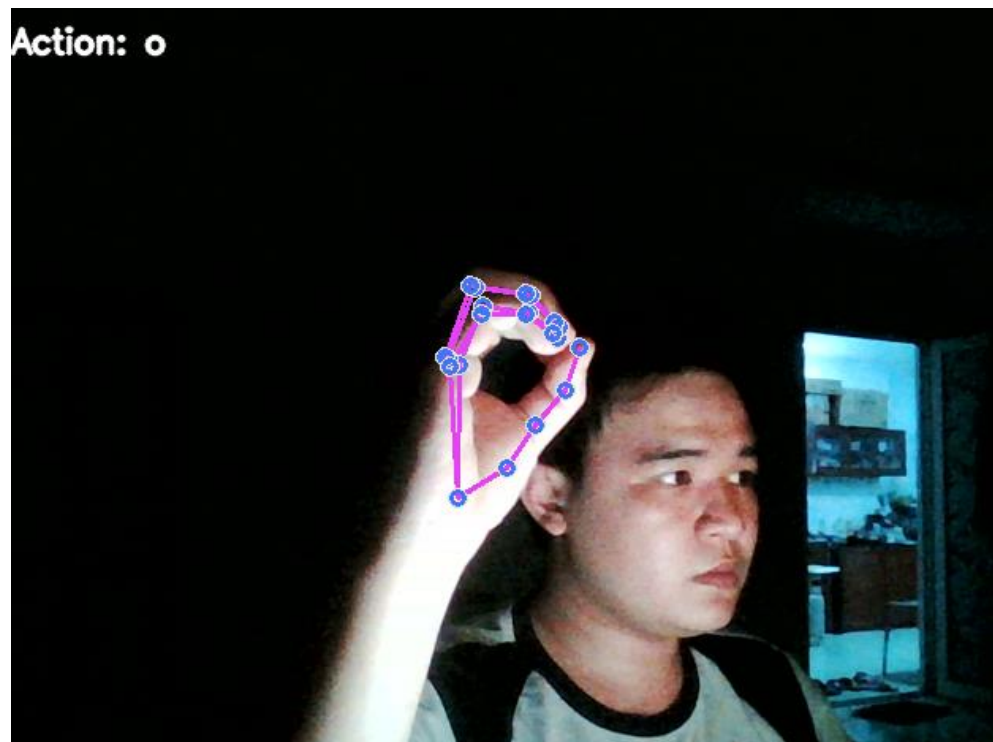


Hình 39: Nhận biết chữ aw

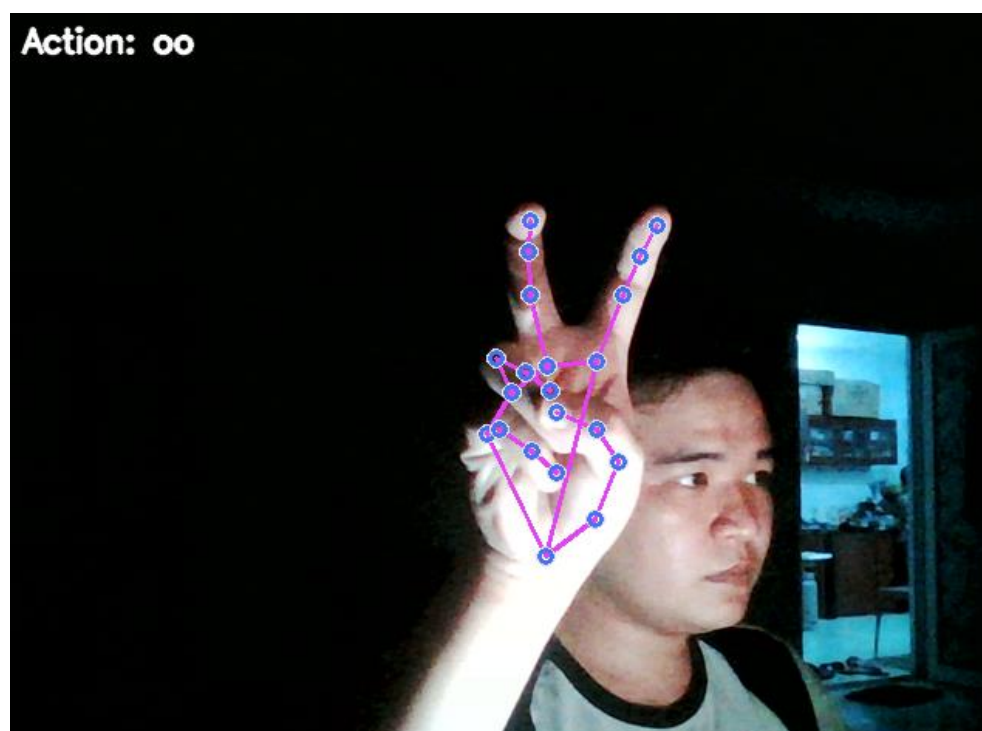


Hình 40: Nhận biết chữ aa

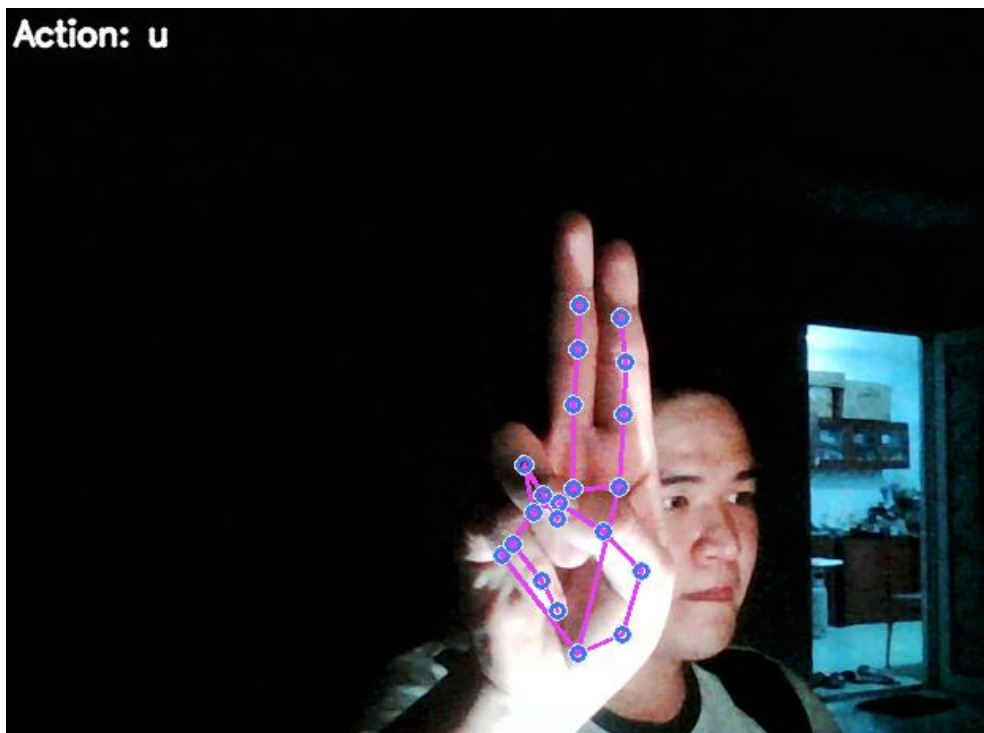




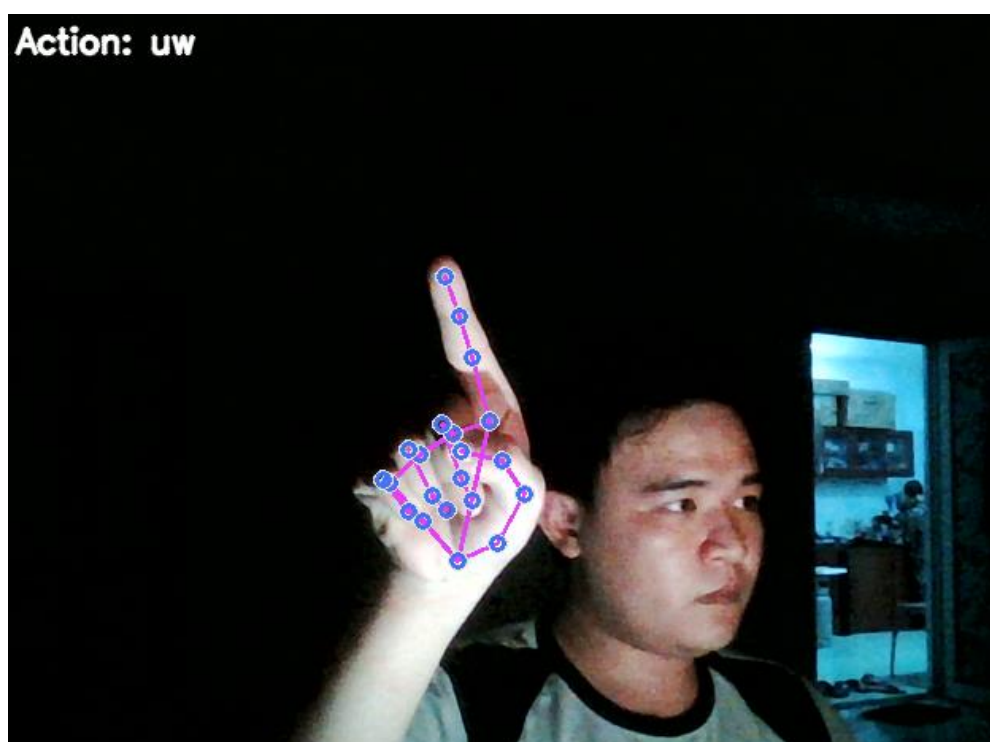
Hình 41: Nhận biết chữ o



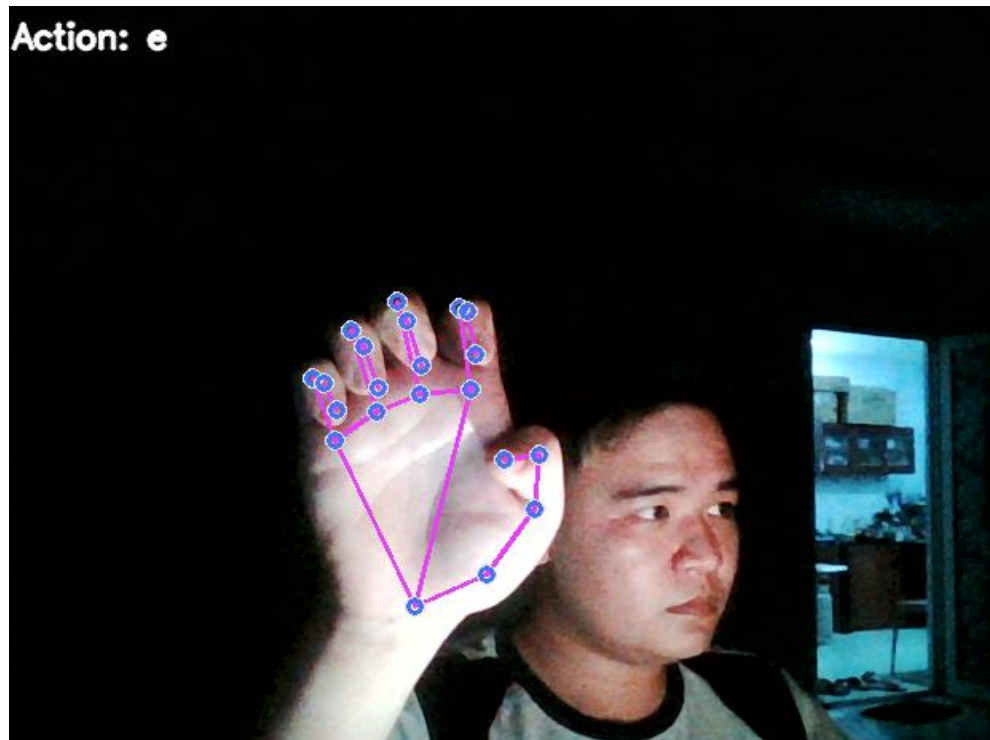
Hình 42: Nhận biết chữ oo



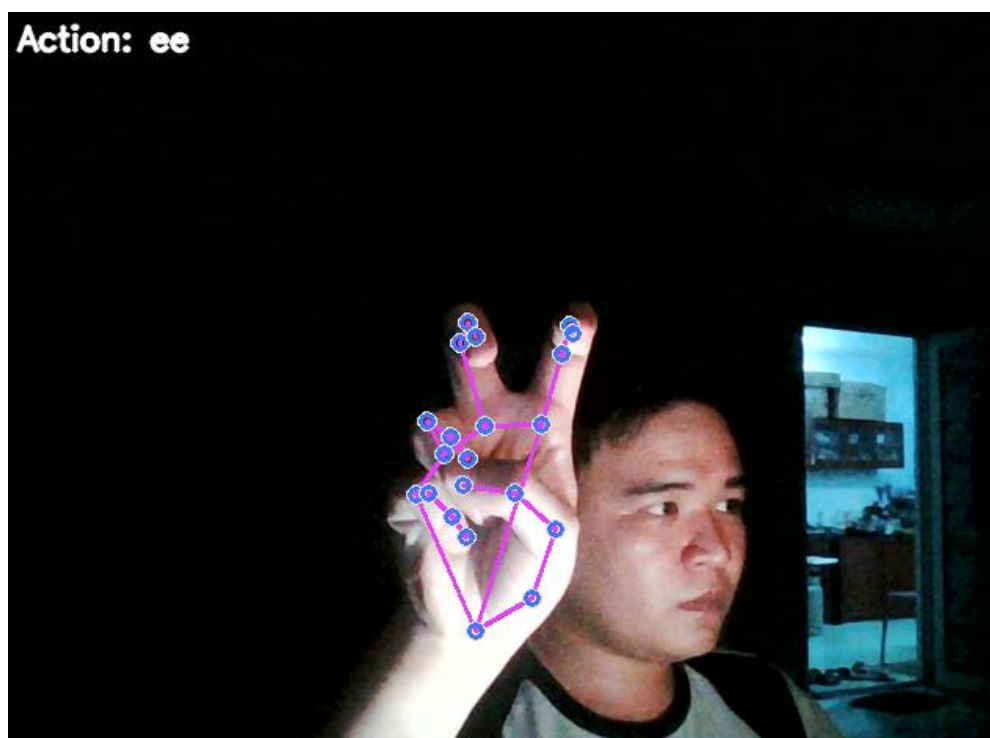
Hình 43: Nhận biết chữ u



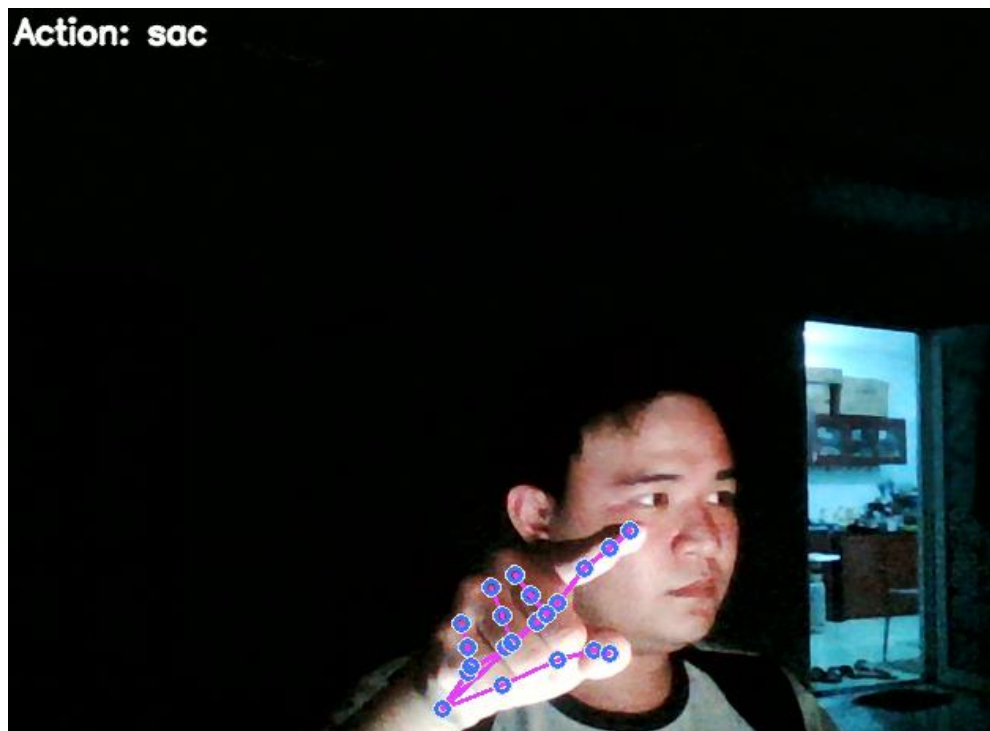
Hình 44: Nhận biết chữ uw



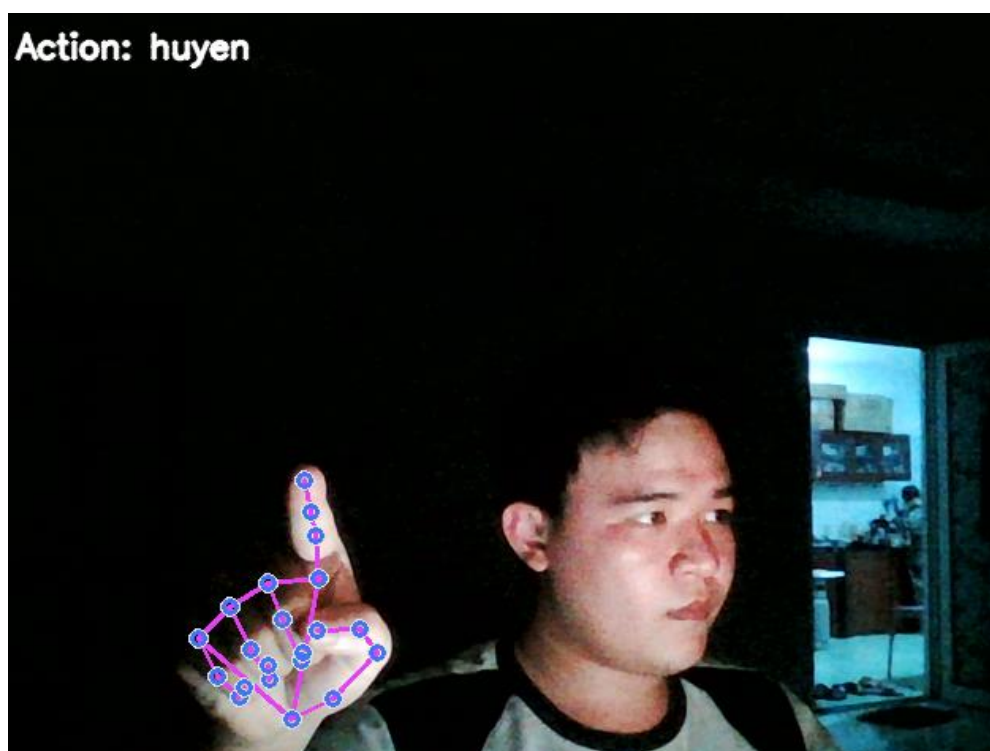
Hình 45: Nhận biết chữ e



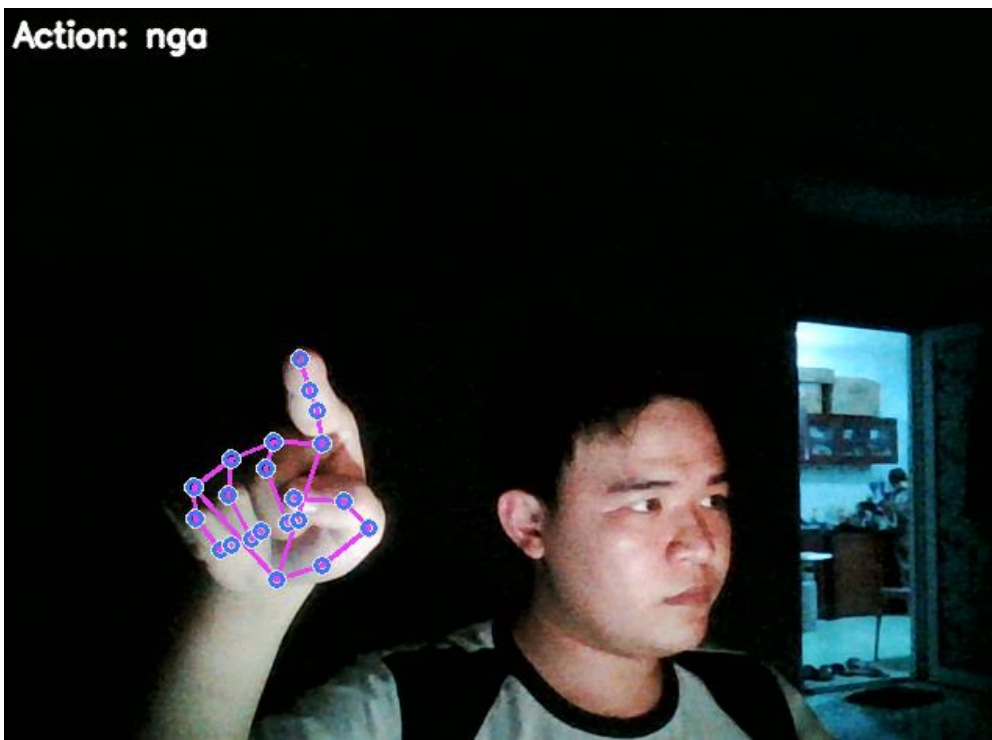
Hình 46: Nhận biết chữ ee



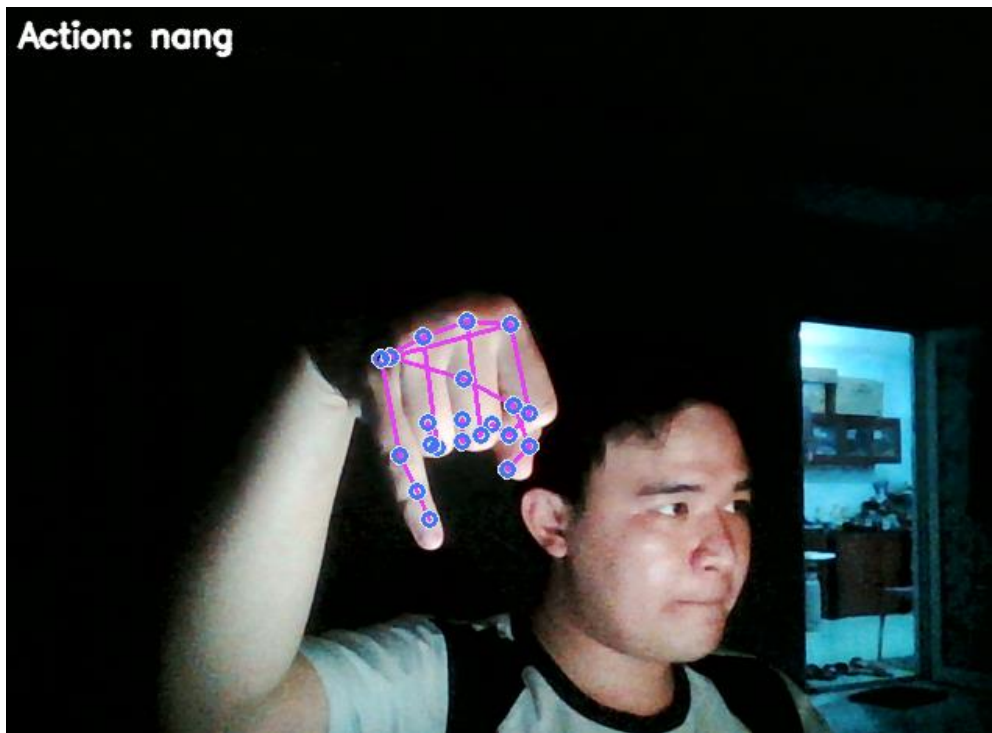
Hình 47: Nhận biết dấu sắc



Hình 48: Nhận biết dấu huyền



Hình 49: Nhận biết dấu nga



Hình 50: Nhận biết dấu nặng

## **PHẦN KẾT LUẬN**

### **1. KẾT QUẢ ĐẠT ĐƯỢC**

Sau khi thực hiện xong đề tài, những kết quả đạt được như sau:

- Tìm hiểu và trình bày được kiến thức của mạng neural nhân tạo, mạng RNN và các biến thể tối ưu hơn của RNN như GRU, LSTM, ...
- Xây dựng được ứng dụng dự đoán thủ ngữ.
- Kết quả đề tài tốt so với mong đợi.
- Tối ưu mô hình khá tốt khi sử dụng trên máy tính không có GPU.

### **2. HẠN CHẾ**

Một số hạn chế còn tồn tại bao gồm:

- Một số kiến thức về thuật toán cũng như phân tích cụ thể còn chưa bao quát.
- Mô hình huấn luyện có độ chính xác gần tuyệt đối do chưa có bộ dữ liệu đầy đủ.
- Việc triển khai mô hình dự đoán chỉ ở mức độ nghiên cứu, chưa đưa ra một sản phẩm hoàn chỉnh để người dùng sử dụng.

### **3. HƯỚNG PHÁT TRIỂN**

Do kiến thức cũng như thời gian còn khá hạn chế, đề tài chưa có thể thành một sản phẩm đầu ra hoàn chỉnh có tính ứng dụng cao và chưa có giao diện để phục vụ người dùng. Vì thế, định hướng phát triển sau này sẽ tiếp tục quá trình nghiên cứu và xây dựng để trở thành một phần mềm hoàn chỉnh. Bên cạnh đó, việc thu thập dữ liệu thủ ngữ vẫn được tiếp tục nhằm bổ sung đầy đủ 29 chữ cái tiếng Việt cũng như dữ liệu thủ ngữ biểu thị thay vì ký tự.

Trong tương lai, tác giả sẽ tiếp tục phát triển đề tài theo đúng hướng đang mong muốn.

## DANH MỤC TÀI LIỆU THAM KHẢO

- [1] A. H. Vo, V. H. Pham, and B. T. Nguyen, “Deep learning for Vietnamese Sign Language recognition in video sequence,” *Int J Mach Learn Comput*, vol. 9, no. 4, pp. 440–445, Aug. 2019, doi: 10.18178/IJMLC.2019.9.4.823.
- [2] T. D. Bui and L. T. Nguyen, “Recognizing postures in vietnamese sign language with MEMS accelerometers,” *IEEE Sens J*, vol. 7, no. 5, pp. 707–712, May 2007, doi: 10.1109/JSEN.2007.894132.
- [3] “3-D-printed robotic arm translates words into sign language.” Accessed: Dec. 29, 2023. [Online]. Available: <https://nypost.com/2018/03/19/3d-printed-robotic-arm-translates-words-into-sign-language/>
- [4] “Convolutional Neural Networks: Architectures, Types & Examples.” Accessed: Dec. 29, 2023. [Online]. Available: <https://www.v7labs.com/blog/convolutional-neural-networks-guide>
- [5] “Các hàm kích hoạt (activation function) trong neural network.” Accessed: Dec. 29, 2023. [Online]. Available: <https://aicurious.io/blog/2019-09-23-cac-ham-kich-hoat-activation-function-trong-neural-networks>
- [6] Q. Yang, “Research on Voltage Presetting Decision of Electrolytic Cell Based on RNN,” *Lecture Notes in Electrical Engineering*, vol. 1059 LNEE, pp. 648–656, 2023, doi: 10.1007/978-981-99-3951-0\_71/COVER.
- [7] “8.7. Lan truyền Ngược qua Thời gian — Đắm mình vào Học Sâu 0.14.4 documentation.” Accessed: Dec. 29, 2023. [Online]. Available: [https://d2l.aivivn.com/chapter\\_recurrent-neural-networks/bptt\\_vn.html](https://d2l.aivivn.com/chapter_recurrent-neural-networks/bptt_vn.html)
- [8] “(PDF) Smart Wallet.” Accessed: Dec. 29, 2023. [Online]. Available: [https://www.researchgate.net/publication/353271013\\_Smart\\_Wallet](https://www.researchgate.net/publication/353271013_Smart_Wallet)
- [9] “(PDF) TradeNIC: Trading-specific SmartNIC Design for Low Latency and High Throughput Algorithmic Trading.” Accessed: Dec. 29, 2023. [Online]. Available: [https://www.researchgate.net/publication/365048610\\_TradeNIC\\_Trading-specific\\_SmartNIC\\_Design\\_for\\_Low\\_Latency\\_and\\_High\\_Throughput\\_Algorithmic\\_Trading](https://www.researchgate.net/publication/365048610_TradeNIC_Trading-specific_SmartNIC_Design_for_Low_Latency_and_High_Throughput_Algorithmic_Trading)
- [10] “Why is BiLSTM better than LSTM ?. Know the underlying functionality | by Sourasish Nath | Medium.” Accessed: Dec. 29, 2023. [Online]. Available: <https://medium.com/@souro400.nath/why-is-bilstm-better-than-lstm-a7eb0090c1e4>



- [11] “python - How to implement a hierarchical model of LSTM units in Keras? - Stack Overflow.” Accessed: Dec. 29, 2023. [Online]. Available: <https://stackoverflow.com/questions/52317452/how-to-implement-a-hierarchical-model-of-lstm-units-in-keras>
- [12] “10.2. Gated Recurrent Units (GRU) — Dive into Deep Learning 1.0.3 documentation.” Accessed: Dec. 29, 2023. [Online]. Available: [https://d2l.ai/chapter\\_recurrent-modern/gru.html](https://d2l.ai/chapter_recurrent-modern/gru.html)
- [13] “layout: forward target: [https://developers.google.com/mediapipe/solutions/vision/hand\\_landmarker](https://developers.google.com/mediapipe/solutions/vision/hand_landmarker) title: Hands parent: MediaPipe Legacy Solutions nav\_order: 4 — MediaPipe v0.7.5 documentation.” Accessed: Dec. 29, 2023. [Online]. Available: <https://mediapipe.readthedocs.io/en/latest/solutions/hands.html>
- [14] “Hand landmarks detection guide | MediaPipe | Google for Developers.” Accessed: Dec. 29, 2023. [Online]. Available: [https://developers.google.com/mediapipe/solutions/vision/hand\\_landmarker](https://developers.google.com/mediapipe/solutions/vision/hand_landmarker)
- [15] “Hand tracking landmarks - Z value range · Issue #742 · google/mediapipe · GitHub.” Accessed: Dec. 29, 2023. [Online]. Available: <https://github.com/google/mediapipe/issues/742>
- [16] D. P. Kingma and J. L. Ba, “Adam: A Method for Stochastic Optimization,” *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, Dec. 2014, Accessed: Dec. 29, 2023. [Online]. Available: <https://arxiv.org/abs/1412.6980v9>
- [17] L. Xu *et al.*, “In-Database Machine Learning with CorgiPile: Stochastic Gradient Descent without Full Data Shuffle,” *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 1286–1300, Jun. 2022, doi: 10.1145/3514221.3526150.
- [18] “12.10. Adam — Dive into Deep Learning 1.0.3 documentation.” Accessed: Dec. 29, 2023. [Online]. Available: [https://d2l.ai/chapter\\_optimization/adam.html](https://d2l.ai/chapter_optimization/adam.html)
- [19] “Lượng tử hóa sau đào tạo | TensorFlow Lite.” Accessed: Dec. 29, 2023. [Online]. Available: [https://www.tensorflow.org/lite/performance/post\\_training\\_quantization?hl=vi](https://www.tensorflow.org/lite/performance/post_training_quantization?hl=vi)
- [20] “Pruning in Keras example | TensorFlow Model Optimization.” Accessed: Dec. 29, 2023. [Online]. Available:



[https://www.tensorflow.org/model\\_optimization/guide/pruning/pruning\\_with\\_keras](https://www.tensorflow.org/model_optimization/guide/pruning/pruning_with_keras)

- [21] “Weight clustering comprehensive guide | TensorFlow Model Optimization.” Accessed: Dec. 29, 2023. [Online]. Available: [https://www.tensorflow.org/model\\_optimization/guide/clustering/clustering\\_comprehensive\\_guide](https://www.tensorflow.org/model_optimization/guide/clustering/clustering_comprehensive_guide)
- [22] “Weight clustering in Keras example | TensorFlow Model Optimization.” Accessed: Dec. 29, 2023. [Online]. Available: [https://www.tensorflow.org/model\\_optimization/guide/clustering/clustering\\_example](https://www.tensorflow.org/model_optimization/guide/clustering/clustering_example)

-