



HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
KHOA CÔNG NGHỆ THÔNG TIN

ĐỒ ÁN

TỐT NGHIỆP ĐẠI HỌC

Đề tài: “Xây dựng hệ thống trả lời tự động dựa trên mô hình Sequence to sequence sử dụng mạng nơ-ron Long Short-term memory networks”

Người hướng dẫn	:	ThS. NGUYỄN MẠNH SƠN
Sinh viên thực hiện	:	BÙI TRẦN TIẾN
Lớp	:	D12CNPM5
Khoá	:	2012
Hệ	:	CHÍNH QUY

Hà Nội, tháng 12/2016

LỜI CẢM ƠN

Em xin chân thành cảm ơn các thầy cô, cán bộ, công nhân viên của Khoa Công nghệ thông tin, các Khoa, đoàn thể khác của Học viện Công Nghệ Bưu Chính Viễn Thông đã nhiệt tình quan tâm và tạo nhiều điều kiện thuận lợi cho em trong quá trình thực hiện đồ án.

Em xin chân thành cảm ơn giảng viên, thạc sĩ Nguyễn Mạnh Sơn đã nhiệt tình hướng dẫn, động viên, hỗ trợ em trong suốt quá trình thực hiện đồ án, giúp em vượt qua những hạn chế của bản thân và những khó khăn trong quá trình nghiên cứu để hoàn thành đồ án thành công, đúng thời hạn dù có rất nhiều công việc của giảng dạy, đoàn thể cũng như gia đình.

Em cũng xin gửi lời cảm ơn tới các thầy Trần Đình Quế, thầy Nguyễn Mạnh Hùng, thầy Nguyễn Xuân Anh, cô Đỗ Thị Bích Ngọc và tất cả các thầy cô của Học viện đã giảng dạy em, gắn liền với những chuyện vui buồn trên giảng đường Đại học trong 4,5 năm học tập tại trường, những người đã truyền đạt cho em không chỉ kiến thức, kinh nghiệm quý báu, mà cả những câu chuyện về cuộc sống, những bài học làm người ý nghĩa. Những kiến thức, bài học đó đã, đang và sẽ tiếp tục là hành trang, động lực giúp em tự hoàn thiện bản thân, vượt qua những khó khăn và vững bước trên con đường phía trước.

Con xin cảm ơn bố mẹ, ông bà và toàn thể đại gia đình đã luôn quan tâm, chăm sóc, tin tưởng, ủng hộ con trong suốt quá trình học tập.

Cảm ơn bạn Đỗ Thị Hân, Nguyễn Tiến Đức và tất cả anh em STP đã luôn sát cánh giúp đỡ, chia sẻ những kiến thức hay kinh nghiệm làm việc tốt và những giờ HL vui vẻ.

Mặc dù em đã cố gắng hoàn thành đồ án bằng tất cả sự nỗ lực và khả năng hạn chế của mình, nhưng chắc chắn vẫn còn nhiều thiếu sót. Em mong nhận được sự cảm thông và những góp ý quý giá từ thầy cô và các bạn.

Em xin chân thành cảm ơn!

Hà Nội, tháng 12 năm 2016

Sinh viên

Bùi Trần Tiến

NHẬN XÉT, ĐÁNH GIÁ, CHO ĐIỂM

(Của giảng viên hướng dẫn)

This image shows a full page of white paper with horizontal dotted lines. The lines are evenly spaced and run across the width of the page, providing a guide for handwriting practice. There are no margins, text, or other markings on the page.

Điểm:(bằng chữ:)

Đồng ý/Không đồng ý cho sinh viên bảo vệ trước hội đồng chấm đồ án tốt nghiệp?.

....., ngày tháng năm 20

CÁN BỘ - GIẢNG VIÊN HƯỚNG DẪN

(*ký, họ tên*)

[illegible]

Đồng ý/Không đồng ý cho sinh viên bảo vệ trước hội đồng chấm đồ án tốt nghiệp?.

CÁN BỘ - GIẢNG VIÊN PHẢN BIỆN

iii

MỤC LỤC

MỤC LỤC	iv
DANH MỤC HÌNH ẢNH	vii
CÁC THUẬT NGỮ VIẾT TẮT	ix
MỞ ĐẦU.....	1
CHƯƠNG 1: HỆ THỐNG TRẢ LỜI TỰ ĐỘNG VÀ CÁC KỸ THUẬT LIÊN QUAN	3
1.1. Đặt vấn đề.....	3
1.2. Bài toán Chatbot.....	3
1.2.1 Mô hình truy vấn.....	3
1.2.2 Mô hình động	4
1.3 Xử lý ngôn ngữ tự nhiên	4
1.4. Các phương pháp học máy cho hệ thống Chatbot	7
1.4.1. Khái niệm học máy	7
1.4.2 Phân loại thuật toán học máy	8
1.5. Tổng quan về mạng nơ-ron	9
1.5.1 Định nghĩa.....	9
1.5.2 Kiến trúc tổng quan của mạng nơ-ron nhân tạo.....	9
1.5.3 Quá trình học của mạng nơ-ron	11
1.5.4. Mạng nơ-ron truyền thẳng và giải thuật lan truyền ngược	12
1.5.5. Mạng nơ-ron hồi quy và giải thuật lan truyền ngược xuyên tâm	14
1.6. Biểu diễn vector từ.....	16
1.6.1 Bài toán	16
1.6.2 Mô hình Skip grams	17

1.7 Kết chương	17
CHƯƠNG 2: MẠNG NƠ-RON LSTM VÀ MÔ HÌNH SEQUENCE TO SEQUENCE	19
2.1. Mạng nơ-ron LSTM.....	19
2.1.1. Tổng quan về mạng LSTM	19
2.1.2 Vấn đề phụ thuộc (Long-Term Dependencies)	20
2.1.3. Mạng LSTM.....	23
2.1.4. Phân tích mô hình LSTM.....	25
2.2. Mô hình Sequence to Sequence trong dịch máy	27
2.2.1 Giới thiệu	27
2.2.2 Mô hình Sequence to Sequence.....	29
2.2.3. Đảo ngược câu nguồn	31
2.3 Kết chương	31
CHƯƠNG 3: CÀI ĐẶT VÀ THỬ NGHIỆM MÔ HÌNH SEQUENCE TO SEQUENCE TRONG BÀI TOÁN TRẢ LỜI TỰ ĐỘNG	32
3.1. Cấu trúc dữ liệu	32
3.1.1 Định dạng dữ liệu	32
3.1.2 Tiền xử lý dữ liệu	32
3.2. Thử nghiệm mô hình “Sequence to Sequence” sử dụng mạng nơ-ron LSTM.....	34
3.2.1 Dữ liệu huấn luyện và cấu hình của mô hình huấn luyện.	34
3.2.2 Quá trình huấn luyện.....	36
3.3 Kết chương	44
CHƯƠNG 4: ỨNG DỤNG MÔ HÌNH VÀO ỨNG DỤNG CHATBOT TRÊN NỀN TẢNG ANDROID	45
4.1 Giới thiệu tổng quan về hệ thống Chatbot trên nền tảng Android .45	

4.1.1 Các công nghệ sử dụng	45
4.1.2 Các chức năng của ứng dụng Chatbot.....	47
4.1.3 Kiến trúc hệ thống	48
4.2 Giao diện chức năng của ứng dụng Chatbot	49
4.2.1 Giao diện chức năng Chatbot	50
4.2.2 Giao diện chức năng Chat help	50
4.2.3 Giao diện chức năng giúp người dùng huấn luyện Chatbot.....	52
4.3. Kết chương	53
KẾT LUẬN	54
DANH MỤC TÀI LIỆU THAM KHẢO	55

DANH MỤC HÌNH ẢNH

Hình 1.1 : Kiến trúc tổng quan của mạng nơ-ron.....	9
Hình 1.2: Quá trình dự đoán của mạng nơ-ron.....	10
Hình 1.3 : Mạng nơ-ron lan truyền thẳng.....	12
Hình 1.4 : Mạng nơ-ron hồi quy	15
Hình 1.5: Mô hình Skip Grams	17
Hình 2.1 Mô hình RNN tổng quát.....	19
Hình 2.2 Mô hình chi tiết của RNN.....	20
Hình 2.3 : Phụ thuộc ngắn	20
Hình 2.4 : Phụ thuộc dài	21
Hình 2.5 : Các hàm xử lý trong LSTM.....	21
Hình 2.6: Biểu đồ sự biến thiên khi sử dụng các hàm sigmoids khác nhau.....	22
Hình 2.7 : Vị trí của hàm tanh trong LSTM.....	23
Hình 2.8 : Tương tác giữa các layer trong LSTM.....	24
Hình 2.9 : Các ký hiệu được sử dụng trong mạng LSTM.....	24
Hình 2.10 : Cell state trong LSTM.....	25
Hình 2.11 Cổng trong LSTM.....	25
Hình 2.12 : Cổng lãng quên (forget gate layer).....	26
Hình 2.13 : Quyết định thông tin được lưu tại Cell state.....	26
Hình 2.14 Cập nhật vào Cell state	27
Hình 2.15 : Quyết định thông tin output ra là gì ?.....	27
Hình 2.16 : Mô hình của chúng tôi đọc một câu đầu vào "ABC" và sản xuất "WXYZ" là câu đầu ra. Mô hình này dừng lại đưa ra dự đoán sau khi xuất ra token cuối cùng của câu (EOS). Lưu ý rằng các LSTM đọc câu đầu vào ngược lại, bởi vì làm như vậy đưa ra nhiều phụ thuộc ngắn hạn trong các dữ liệu mà làm cho các vấn đề tối ưu hóa dễ dàng hơn nhiều.	28
Hình 3.1 : Tập dữ liệu huấn luyện.....	34
Hình 3.2 : Tập dữ liệu kiểm tra	35
Hình 3.3 : Các bước xử lý chính trong mô hình "Sequence to Sequence"	36
Hình 3.4 : Khởi tạo tập từ vựng từ dữ liệu huấn luyện	37
Hình 3.5 : Tập từ vựng	38

<i>Hình 3.6 : Danh sách id theo dữ liệu huấn luyện.....</i>	<i>39</i>
<i>Hình 3.7 : Danh sách id theo dữ liệu kiểm tra</i>	<i>40</i>
<i>Hình 3.8 : Model được sinh ra sau mỗi 100 bước huấn luyện.....</i>	<i>41</i>
<i>Hình 3.9 : Quá trình huấn luyện</i>	<i>42</i>
<i>Hình 3.10 : Độ lỗi thấp và ổn định ít thay đổi</i>	<i>42</i>
<i>Hình 3.11 : Kết quả thu được sau quá trình huấn luyện.....</i>	<i>43</i>
<i>Hình 4.12 : Quá trình thực nghiệm với tập model sau huấn luyện.....</i>	<i>44</i>
<i>Hình 4.1 : Giao diện quản lý của Ứng dụng Google Speech-to-text trên nền tảng Android</i>	<i>45</i>
<i>Hình 4.2 : Hệ thống quản lý của Ứng dụng Google Text to Speech trên nền tảng Android</i>	<i>46</i>
<i>Hình 4.3 : Sơ đồ kiến trúc hệ thống.....</i>	<i>48</i>
<i>Hình 4.4 : Giao diện màn hình khởi động ứng dụng</i>	<i>49</i>
<i>Hình 4.5 : Giao diện màn hình chức năng Chat bot</i>	<i>50</i>
<i>Hình 4.6 : Giao diện Chat help</i>	<i>51</i>
<i>Hình 4.6 : Bong bóng chat help giúp người dùng thuận tiện hơn.....</i>	<i>52</i>
<i>Hình 4.7 : Giao diện huấn luyện và phản hồi câu trả lời của bot</i>	<i>53</i>

CÁC THUẬT NGỮ VIẾT TẮT

Thuật ngữ	Giải thích
API	Application Programming Interface - Giao diện lập trình ứng dụng
AI	Artificial intelligence – Trí tuệ nhân tạo
FNN	Feedforward Neural Network - Mạng nơ-ron truyền thẳng
KNN	k-Nearest Neighbors – k láng giềng gần nhất
LSTM	Long Short-term memory
NLP	Natural language processing – Xử lý ngôn ngữ tự nhiên
RNNs	RECURRENT NEURAL NETWORKS – Mạng nơ-ron hồi quy
TTS	Text to Speech – Chuyển văn bản thành giọng nói

MỞ ĐẦU

Chatbot là những hệ thống có khả năng “trò chuyện” với con người thông qua ngôn ngữ tự nhiên. Những hệ thống này được phát triển nhằm thay thế sức lao động của con người về một lĩnh vực cụ thể nào đó (y tế, giáo dục, luật pháp...). Trong một vài năm gần đây, các tập đoàn công nghệ lớn hàng đầu thế giới (Google, Microsoft ...) đã đưa ra những hệ thống trả lời tự động như Allo của Google hay Tay của Microsoft. Điều này chứng tỏ Chatbot đang là một trong những hướng nghiên cứu tiềm năng, có thể đem lại những giá trị rất lớn cho con người.

Một bài toán Chat bot cơ bản thường có 3 bước cơ bản như tiền xử lý đầu vào có thể thông qua văn bản hoặc giọng nói để trích chọn đặc trưng để đưa vào mô hình dự đoán câu trả lời dựa trên tập dữ liệu huấn luyện, khi đã tìm được câu trả lời thì có thể phản hồi lại người dùng thông qua văn bản, giọng nói hoặc một hành động cụ thể.

Mục tiêu của đồ án nhằm nắm bắt cơ sở lý thuyết nền tảng của bài toán Chat bot sử dụng mô hình Sequence to Sequence, cài đặt, thử nghiệm với dữ liệu tiếng Việt, đưa ra phương pháp tiếp cận khi áp dụng xử lý dữ liệu tiếng Việt. Xây dựng thành công một ứng dụng di động sử dụng kết quả trong quá trình thực nghiệm cho chức năng trả lời tự động và hỗ trợ người dùng tự động.

Để thực hiện được mục tiêu trên, phương pháp nghiên cứu của đề tài:

- Nghiên cứu các tài liệu giới thiệu bài toán Chat bot và các khái niệm liên quan như: Xử lý ngôn ngữ tự nhiên, các phương pháp học máy, các mô hình mạng nơ-ron, các mô hình áp dụng mạng nơ-ron...
- Nghiên cứu các mô hình truy vấn và sinh tự động câu trả lời. Cài đặt và thử nghiệm mô hình sinh tự động sử dụng mô hình Sequence to Sequence sử dụng mạng nơ-ron Long Short-term memory cố gắng đề xuất một phương pháp áp dụng mô hình vào việc xây dựng hệ thống trả lời sử dụng tiếng Việt.
- Xây dựng ứng dụng di động Chatbot đơn giản sử dụng tiếng Việt. Tìm các tích hợp kết quả của quá trình thực nghiệm một cách hợp lý vào ứng dụng Chat bot dưới dạng chức năng trả lời tự động và hỗ trợ người dùng tự động.

Đồ án được cấu trúc thành 4 chương:

- **Chương 1: Hệ thống trả lời tự động và các kỹ thuật liên quan:** Giới thiệu về nhóm bài toán xây dựng Chatbot, các khái niệm các kỹ thuật liên quan. Trong đi sâu vào xử lý ngôn ngữ tự nhiên, các phương pháp học máy và mạng nơ-ron.
- **Chương 2: Mạng nơ-ron LSTM và Mô hình Sequence to Sequence:** Giới thiệu chi tiết về mạng nơ-ron LSTM và các phương pháp tính toán trong mạng LSTM. Giới thiệu chi tiết về mô hình Sequence to Sequence sử dụng mạng nơ-ron LSTM.
- **Chương 3: Cài đặt và thử nghiệm mô hình Sequence to Sequence trong bài toán trả lời tự động:** Chi tiết về cách cấu hình mô hình, cấu trúc dữ liệu, quá trình huấn luyện và kết quả thử nghiệm.
- **Chương 4: Ứng dụng mô hình vào ứng dụng Chatbot trên nền tảng Android:** Giới thiệu về ứng dụng di động của dự án: Hệ thống trả lời tự động trực tuyến Chatbot được tích hợp tính năng trả lời tự động và hỗ trợ tự động sử dụng kết quả của quá trình huấn luyện sử dụng mô hình Sequence to Sequence.

CHƯƠNG 1: HỆ THỐNG TRẢ LỜI TỰ ĐỘNG VÀ CÁC KỸ THUẬT LIÊN QUAN

Chương này sẽ giới thiệu tổng quan về hệ thống trả lời tự động (Chat bot), đồng thời đặt vấn đề xây dựng ứng dụng Chat bot trên nền tảng Android.

1.1. Đặt vấn đề

Trong xu hướng tự động hóa, con người luôn từng bước phát triển những hệ thống tự động ngày một hoàn thiện hơn, nhằm thay thế sức lao động của con người. Chatbot cũng không nằm ngoài xu hướng trên. Chatbot, hay hệ thống thông trả lời tự động, như tên của nó, là những hệ thống có khả năng “trò chuyện” với con người thông qua ngôn ngữ tự nhiên - ngôn ngữ của con người chứ không phải ngôn ngữ của máy tính hay ngôn ngữ lập trình. Những hệ thống này được phát triển nhằm thay thế sức lao động của con người trong những lĩnh vực như tư vấn chăm sóc khách hàng hay tư vấn cho con người về một lĩnh vực cụ thể nào đó (y tế, giáo dục, luật pháp,...). Những hãng công nghệ lớn cũng đã đầu tư và phát triển những hệ thống trả lời thông minh nhằm gia tăng tiện ích, cải thiện chất lượng cuộc sống của con người. Ví dụ như những ứng dụng M (Facebook), Siri (Apple), Google, WeChat,... Điều này chứng tỏ Chatbot đang là một trong những hướng nghiên cứu rất tiềm năng, có thể đem lại những giá trị rất lớn cho con người.

1.2. Bài toán Chatbot

Chatbot là một hệ thống trí tuệ nhân tạo, cho phép người sử dụng có thể tương tác bằng cách gửi một thông điệp dưới dạng ngôn ngữ tự nhiên cho hệ thống và sau đó có thể nhận phản hồi phù hợp. Hệ thống chatbot thường dùng các kỹ thuật học máy để huấn luyện cho máy tính có thể hiểu được ngôn ngữ của con người và đưa ra những câu trả lời hợp lý.

Hiện nay có 2 dạng hệ thống Chatbot [7]:

- Mô hình truy vấn
- Mô hình động

1.2.1 Mô hình truy vấn

Hệ thống này thực chất là một hệ thống có khả năng đánh giá những câu trả lời đã có sẵn dựa trên dữ liệu được nhập từ người dùng và đưa ra câu trả lời là câu phù hợp nhất. Mô hình này dựa trên phương pháp đánh giá heuristic, có thể chỉ đơn giản là đưa ra một số luật nào đó hay phức tạp hơn là những hệ thống phân loại sử dụng những kỹ thuật học máy. Những hệ thống kiểu này không hề tạo ra những câu trả lời mới mà chỉ đơn giản là chọn câu trả lời phù hợp nhất trong tập những câu trả lời đã biết trước.

1.2.2 Mô hình động

Những hệ thống này hoàn toàn không dựa trên tập câu trả lời đã biết mà chúng sẽ tự tạo ra những câu trả lời. Dựa trên việc ghi nhớ lại xác suất của các từ vựng xuất hiện cùng nhau trong câu hỏi và câu trả lời trong quá trình huấn luyện. Hệ thống sẽ tạo ra từng từ một và ghép lại thành một câu trả lời.

Hệ thống dựa vào mô hình truy vấn chắc chắn sẽ không bao giờ bị lỗi sai về ngữ nghĩa do toàn bộ những câu trả lời đều đã được định nghĩa sẵn. Vấn đề chỉ là làm cho máy tính có thể xếp hạng để chọn ra câu trả lời phù hợp nhất. Tuy nhiên mô hình này không hề tạo ra những câu trả lời mới. Vì vậy không thể bao quát hết mọi trường hợp mà người dùng sử dụng. Còn đối với mô hình động thì câu trả lời hoàn toàn không biết trước. Mọi thứ đều được máy học để tạo ra câu trả lời. Tuy nhiên mô hình này có thể sẽ gặp phải những lỗi sai về mặt ngữ nghĩa.

Chương này sẽ nêu một số khái niệm cơ bản và cơ sở lý thuyết sử dụng trong quá trình làm đồ án như: xử lý ngôn ngữ tự nhiên, học máy, mạng nơ-ron ... và một số ứng dụng của nó trong thực tế.

1.3 Xử lý ngôn ngữ tự nhiên

Xử lý ngôn ngữ tự nhiên (Natural Language Processing-NLP) là một nhánh của trí tuệ nhân tạo tập trung vào các ứng dụng trên ngôn ngữ của con người. Đây là một khái niệm để chỉ các kỹ thuật, phương pháp thao tác trên ngôn ngữ tự nhiên bằng máy tính. Trong trí tuệ nhân tạo thì việc xử lý ngôn ngữ tự nhiên là khó nhất vì nó liên quan đến việc phải hiểu ý nghĩa của ngôn ngữ. Trong NLP có hai quan điểm cơ bản sau :

- Xử lý từ ngữ bằng máy tính

- Làm cho máy tính hiểu được nghĩa của từ ngữ

Hiện tại cả hai hướng đều đang được tích cực nghiên cứu và phát triển, nhờ đó rất nhiều hệ thống hiệu quả đã và đang được tạo ra.

➤ **Các bước xử lý :**

- ❖ Phân tích hình thái : Trong bước này từng từ sẽ được phân tích và các ký tự không phải chữ (như các dấu câu) sẽ được tách ra khỏi các từ. Trong tiếng anh và nhiều ngôn ngữ khác, các từ được phân tách với nhau bởi dấu cách. Tuy nhiên trong Tiếng Việt, dấu cách lại được dùng để phân tách các âm tiết với nhau. Phân tách từ trong Tiếng Việt là một công việc không hề dễ dàng.
- ❖ Phân tích cú pháp : Dãy các từ sau khi được phân tích sẽ được biến đổi thành cấu trúc thể hiện sự liên kết giữa các từ này.
- ❖ Phân tích ngữ nghĩa : Thêm các ngữ nghĩa vào các cấu trúc tạo ra bởi bộ phân tích cú pháp.
- ❖ Tích hợp văn bản : Ngữ nghĩa của một câu riêng biệt có thể phụ thuộc vào câu đứng trước nó, hoặc câu phía sau nó tùy theo văn cảnh.
- ❖ Phân tích thực nghĩa : Cấu trúc thể hiện điều được phát ngôn sẽ được thông dịch lại để xác định nó thật sự có nghĩa là gì.

➤ **Các ứng dụng cơ bản của NLP :**

- ❖ Nhận dạng chữ viết : Có hai kiểu nhận dạng, thứ nhất là nhận dạng chữ in (ví dụ như nhận dạng chữ trên sách giáo khoa rồi chuyển nó thành dạng văn bản điện tử). Thứ hai, phức tạp hơn, là nhận dạng chữ viết tay, có khó khăn bởi vì nó không có khuôn dạng rõ ràng, tùy từng người sẽ có dạng chữ viết khác nhau. Với chương trình nhận dạng chữ viết in có thể chuyển hàng ngàn tài liệu sách thành văn bản điện tử trong một thời gian ngắn. Nhận dạng chữ viết con người có ứng dụng trong khoa học hình sự và bảo mật thông tin.
- ❖ Nhận dạng tiếng nói: Nhận dạng tiếng nói tự nhiên của con người rồi chuyển chúng thành văn bản tương ứng. Điều này giúp cho thao tác của con người lên các thiết bị trở nên dễ dàng và thuận tiện hơn. Chẳng hạn khi muốn gõ một tài

liệu nào đó, thay vì dùng công tác thủ công chúng ta sẽ đọc lên và trình soạn thảo sẽ tự ghi lại.

- ❖ Tổng hợp tiếng nói: Từ một văn bản tự động tổng hợp thành tiếng nói. Thay vì đọc một cuốn sách hay một bài báo, nó sẽ tự động đọc cho chúng ta. Điều này sẽ trợ giúp rất nhiều cho những người khiếm thị, và cũng là bước cuối cùng trong nghiên cứu lĩnh vực giao tiếp giữa con người và robot.
- ❖ Tìm kiếm thông tin (Information retrieval): Đặt câu hỏi và chương trình tự tìm ra nội dung phù hợp nhất. Hiện nay, với sự trợ giúp đắc lực từ Internet, việc tiếp cận thông tin trở nên dễ dàng hơn rất nhiều. Việc khó khăn lúc này là làm sao để tìm được thông tin người tìm kiếm cần trong khối thông tin khổng lồ đó, và trên hết là thông tin đó phải đáng tin cậy. Các máy tìm kiếm dựa trên giao diện web hiện nay như Goole, Yahoo... chỉ phân tích nội dung đơn giản dựa trên tần suất của các từ khóa, thứ hạng của các trang và một vài tiêu chí đánh giá khác để đưa ra kết quả. Do đó nhiều khi chúng ta không thể tìm kiếm được câu trả lời mình muốn, thậm chí bị dẫn tới một liên kết không liên quan. Thực tế hiện nay chưa có một máy tìm kiếm nào có thể “hiểu” được ngôn ngữ tự nhiên của con người.
- ❖ Dịch tự động (Machine translate): Đây là chương trình dịch tự động từ ngôn ngữ này sang ngôn ngữ khác. Một phần mềm điển hình về Tiếng Việt của chương trình này là Evtrans của Softtext, dịch tự động từ Tiếng Anh sang Tiếng Việt và ngược lại.
- ❖ Tóm tắt văn bản: Từ một văn bản cho trước, tóm tắt thành một văn bản ngắn hơn mà vẫn đảm bảo nội dung chính của văn bản đó. Đây cũng là một lĩnh vực được nghiên cứu rất nhiều trong những năm gần đây và đã cho những kết quả khá tích cực.
- ❖ Khai phá dữ liệu (Data mining)

1.4. Các phương pháp học máy cho hệ thống Chatbot

1.4.1. Khái niệm học máy

Học máy (Machine learning) đúng như tên của nó, là những kỹ thuật giúp cho máy tính có khả năng học hỏi, ghi nhớ những thông tin phức tạp, đánh giá thông tin và đưa ra những dự đoán. Máy tính sẽ học từ dữ liệu, tìm ra quy luật trong dữ liệu mẫu để đưa ra những đánh giá. Bản chất của những kỹ thuật học máy là sử dụng những mô hình thống kê nhằm ghi nhớ sự phân bố của dữ liệu. Tuy nhiên học máy khác với mô hình thống kê ở chỗ mô hình thống kê đơn thuần chỉ ghi nhớ còn học máy ngoài khả năng ghi nhớ còn có thể đưa ra những dự đoán từ những kinh nghiệm được ghi nhớ trước đó. Chẳng hạn như ghi nhớ sự phân bố các điểm ảnh để có thể nhận diện được mặt người trong các bức ảnh hay ghi nhớ ngữ nghĩa của từng câu trong đoạn hội thoại để xử lý trong bài toán Chatbot.

Do có khả năng ghi nhớ mạnh mẽ và xử lý trên tập dữ liệu cực lớn nên những kỹ thuật học máy thường được sử dụng cho những bài toán mà máy tính phải đối mặt với dữ liệu lớn và phức tạp như trí tuệ nhân tạo (AI) và big data. Trong lĩnh vực AI hiện nay, những mô hình học máy có thể xử lý những bài toán về nhận dạng mẫu (nhận dạng đối tượng trong ảnh, nhận dạng giọng nói,...), xử lý ngôn ngữ tự nhiên (hiểu được ngôn ngữ của con người, chuyển đổi qua lại giữa ngôn ngữ máy và ngôn ngữ tự nhiên,...). Còn trong lĩnh vực Big Data, học máy đóng vai trò như những công cụ giúp con người xử lý những dữ liệu cực lớn mà con người không thể xử lý nổi. Không những thế, học máy còn đưa ra những tri thức từ những núi dữ liệu mà con người khó có thể nhìn ra. Ví dụ như bài toán tìm cộng đồng (nhóm) ẩn có cùng sở thích dựa trên tập dữ liệu về sở thích của khách hàng. Tập dữ liệu này có thể lên tới hàng tỉ bản ghi và hàng triệu những thông tin đầu vào khác nhau. Điều này rõ ràng con người không thể xử lý tốt với lượng dữ liệu lớn như vậy. Nhưng học máy thì có thể.

Tóm lại, những mô hình học máy ngày nay đang cố gắng làm cho máy tính có những khả năng nhận thức của con người như nghe, nhìn, hiểu ngôn ngữ,... và hỗ trợ con người xử lý những tập dữ liệu khổng lồ.

Một mô hình học máy bao gồm những thành phần sau:

- Tập dữ liệu (dataset): Tập dữ liệu mẫu để máy tính ghi nhớ và học hỏi. Ví dụ đối với bài toán nhận diện mặt người thì dataset sẽ là những ảnh mặt người dưới dạng

ma trận, với bài toán hệ thống gợi ý thì sẽ là những thông tin của người dùng - sản phẩm, với bài toán chatbot thì sẽ là những đoạn hội thoại,...

- Thuật toán xây dựng mô hình (Modeling algorithm): Đây là phần cốt lõi của học máy, là tập những quy luật đặt ra nhằm khiến máy tính có thể “ghi nhớ” và đưa ra những dự đoán.
- Đánh giá mô hình thuật toán (Evaluation): Là quá trình kiểm nghiệm hiệu năng của thuật toán thông qua các chỉ số đánh giá như Recall@k, Precision@k, MAE,...

1.4.2 Phân loại thuật toán học máy

Các thuật toán học máy thường được chia ra như sau:

➤ Học có giám sát (Supervised Learning):

Thuật toán học có giám sát phải đảm bảo 2 điều kiện là phải biết trước có những kết quả đầu ra nào và dấu hiệu để có thể nhận ra kết quả đầu ra đó (trong tập dữ liệu đã xác định những chỉ số như thế nào sẽ có kết quả tương ứng). Ví dụ như bài phân loại quả trong thùng thành cam, quýt, bưởi. Ta đã biết kết quả đầu ra là cam, quýt, bưởi và những đặc trưng của chúng (cam thường to hơn quýt và bưởi to hơn cam).

Những thuật toán học có giám sát thường để giải quyết những bài toán phân loại (classification) và hồi quy (regression).

Một số thuật toán: Decision Tree, Support Vector Machine, KNN, Naïve Bayes, ...

➤ Học không giám sát (Unsupervised Learning):

Thuật toán học không giám sát khác với học có giám sát. Những thuật toán này không hề biết có những kết quả đầu ra như thế nào mà chỉ dựa trên những đánh giá chủ quan những giá trị đặc trưng. Ví dụ cũng với bài toán phân loại quả trong thùng nhưng trong thùng toàn những quả lạ chưa biết đến bao giờ. Vì vậy ta chỉ có thể phân loại bằng cách xem xét những đặc trưng của chúng (màu sắc, độ lớn, ...) để đưa ra những quả nào là cùng một loại.

Những thuật toán học không giám sát thường sử dụng để giải quyết những bài toán phân cụm, giảm số chiều.

Một số thuật toán: k-means, Spectral clustering, Hierarchical clustering, ...

➤ Học bán giám sát (Semi supervised Learning):

Như tên của nó, học bán giám sát sẽ ở khoảng giữa học không giám sát và học có giám sát. Học bán giám sát sẽ huấn luyện cho một chút dữ liệu có nhãn và đa phần không có nhãn. Người ta nhận thấy rằng điều này có thể thay đổi một chút về hiệu năng của thuật toán học không giám sát.

Một số thuật toán: Constrain clustering, PU clustering, Transductive clustering, ...

➤ **Học củng cố (Reinforcement Learning):**

Đây là phương pháp học chỉ dựa vào những hàm chuẩn để đánh giá việc thực hiện là tiêu cực hay tích cực nhằm đưa ra bước thay đổi làm tối ưu hóa hàm chuẩn đó.

Một số thuật toán: Temporal difference learning, Q-learning, SARSA,...

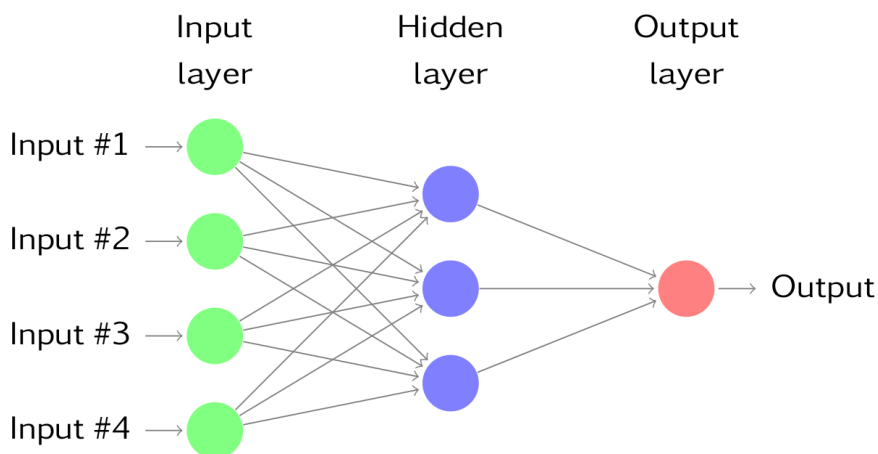
1.5. Tổng quan về mạng nơ-ron

1.5.1 Định nghĩa

Trong sinh học, thuật ngữ mạng nơ-ron là để chỉ tập hợp các dây thần kinh liên kết với nhau, kiểm soát, điều khiển mọi giác quan, suy nghĩ, hành động của con người. Còn trong lĩnh vực công nghệ, học máy, mạng nơ-ron hay mạng nơ-ron nhân tạo là một dạng mô hình hình học máy. Nó mô phỏng mạng nơ-ron sinh học: học hỏi từ kinh nghiệm, lưu trữ những kinh nghiệm, rút ra tri thức và sử dụng những tri thức đó để dự đoán những dữ liệu chưa biết.

Mạng nơ-ron thường được áp dụng trong lĩnh vực trí tuệ nhân tạo nhằm giải quyết những bài toán đòi hỏi tính chính xác cao như điều khiển tự động, nhận dạng, xử lý ngôn ngữ tự nhiên, ...

1.5.2 Kiến trúc tổng quan của mạng nơ-ron nhân tạo



Hình 1.1 : Kiến trúc tổng quan của mạng nơ-ron

Kiến trúc chung của mạng nơ-ron nhân tạo bao gồm thành phần: **Input layer**, **Hidden layer** và **Output layer**.

- **Input layer:** Là nơi để nạp dữ liệu vào. Mỗi nơ-ron tương ứng với một thuộc tính (attribute) hoặc đặc trưng (feature) của dữ liệu đầu vào.
- **Hidden layer:** Là nơi xử lý dữ liệu trước khi đưa ra output. Mỗi input đi vào sẽ được áp dụng hàm kích hoạt (activation function). thường là hàm tổng (Summation function) để đưa ra giá trị của một nơ-ron tại hidden layer. Có thể có nhiều hơn một hidden layer. Khi đó đầu ra của một hidden layer sẽ là đầu vào cho hidden layer tiếp theo.
- **Output layer:** Là giá trị đầu ra của mạng nơ-ron. Sau khi đi qua hidden layer cuối cùng, dữ liệu sẽ được chuyển hóa bằng một hàm số được gọi là hàm chuyển đổi (Transfer function) và đưa ra output cuối cùng. Hàm chuyển đổi thường là hàm $\tanh(x)$, $\text{sigmoid}(x)$ hoặc $\text{softmax}(x)$.

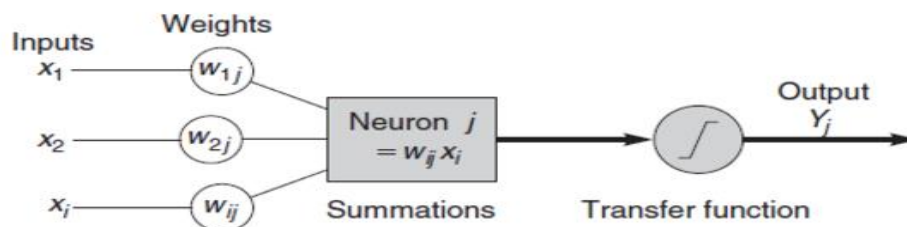
Trong mạng nơ-ron nhân tạo, mỗi nơ-ron đóng vai trò như một đơn vị xử lý thông tin. Giữa các nơ-ron được đánh trọng số, gọi là các trọng số liên kết (Connection Weight), thể hiện độ quan trọng của thông tin của mỗi nơ-ron trước với nơ-ron sau. Quá trình học của mạng nơ-ron cũng chính là việc điều chỉnh những tham số này sao cho phù hợp nhất.

Thông tin của mỗi nơ-ron đi qua mỗi layer sẽ được áp dụng hàm kích hoạt (thường là hàm tổng) trước khi trở thành thông tin của nơ-ron tiếp theo.

Hàm tổng:
$$f(x) = \sum_{j=1}^n x_j \times w_{ij}$$

x_j : là giá trị của nơ-ron thứ j .

w_{ij} : là trọng số liên kết giữa nơ-ron thứ j ở layer trước và nơ-ron thứ i ở layer sau.



Hình 1.2: Quá trình dự đoán của nạng nơ-ron

Thông tin ở hidden layer cuối cùng trước khi tới output sẽ phải qua một hàm chuyển đổi (Transfer function). Việc sử dụng hàm chuyển đổi có hai mục đích: Một là những thông tin qua hàm tổng có thể sẽ lớn, vì vậy hàm chuyển đổi sẽ khiến giá trị đầu ra bị chặn. Ví dụ đầu ra của hàm sigmoid sẽ nằm trong khoảng $[0, 1]$ còn tanh thì là $[-1, 1]$. Hai là hàm tổng là một hàm tuyến tính, giá trị đầu ra sẽ không “mượt”, hàm chuyển đổi sẽ “mượt hóa” kết quả của hàm tổng để kết quả đầu ra có thể chính xác hơn. Hàm chuyển đổi thường sử dụng là hàm *softmax*, *tanh*, *sigmoid* hay chỉ đơn thuần là một ngưỡng để quyết định giá trị cho output.

Hàm softmax:
$$softmax(x) = \frac{e^x}{\sum_k^n e^k}$$

n : là tổng số nơ-ron ở output layer

x : là giá trị từ hidden layer cuối cùng sau khi đi qua hàm tổng

Hàm Sigmoid:
$$f(x) = \frac{1}{1+e^{-x}}$$

Hàm tanh:
$$tanh(x) = 1 - tanh^2(x)$$

1.5.3 Quá trình học của mạng nơ-ron

Như đã nói ở trên, bản chất quá trình học của mạng nơ-ron là quá trình tính toán, kiểm tra và thay đổi các giá trị trọng số sao cho phù hợp nhất. Quá trình học của mạng nơ-ron sẽ bao gồm 3 phần nhỏ:

- Quá trình tính toán: Như đã nói ở phần trước, thông tin sẽ được đặt vào các nơ-ron và lan truyền cho tới output.
- Quá trình kiểm tra: Thường sử dụng những hàm mất mát (Loss function) để kiểm tra lỗi xảy ra tại nơ-ron nào, từ đó cập nhật để sửa.
- Quá trình cập nhật: Sau khi kiểm tra biết được lỗi xảy ra tại nơ-ron nào thì sẽ cập nhật tương ứng tại nơ-ron đó.

Hàm mất mát (Loss function) là hàm đánh giá độ lỗi gây ra tại một nơ-ron nào đó. Hàm mất mát có rất nhiều dạng, điển hình là dạng quadratic và log likelihood:

Dạng quadratic: $L_j = \frac{1}{2}(p_j - z_j)^2$

Dạng log likelihood: $L_j = -z_j \ln p_j$

p_j : là giá trị đầu ra tại nơ-ron thứ j

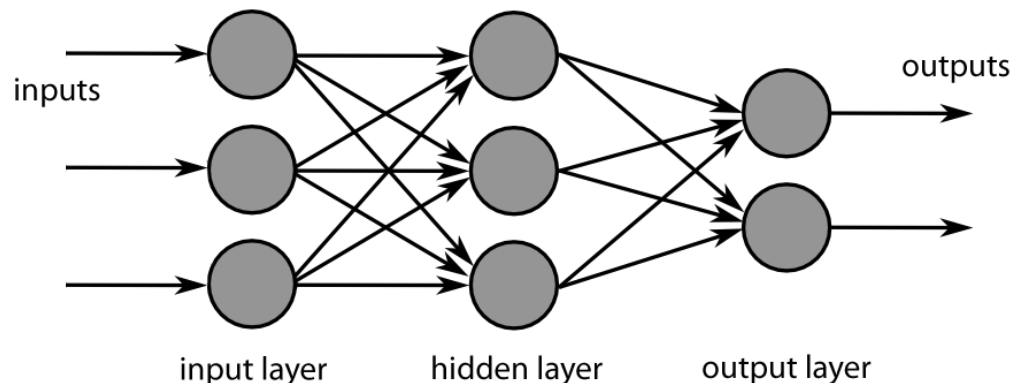
z_j : là giá trị thực tế tại nơ-ron thứ j

Việc lựa chọn hàm mất mát sẽ phụ thuộc vào hàm chuyển đổi là gì. Ví dụ đối với hàm sigmoid thì sẽ chọn quadratic form còn softmax sẽ dùng log likelihood để tiện việc tính đạo hàm.

Mạng nơ-ron có thể học bằng cả những thuật toán học có giám sát hay học không có giám sát. Đối với học có giám sát thì thuật toán sẽ tự kiểm tra giá trị sau quá trình tính toán với giá trị thực tế để cập nhật trọng số. Điển hình cho kỹ thuật này là thuật toán lan truyền ngược (Backpropagation) [3][4]. Còn đối với học không giám sát, do không biết trước kết quả nên quá trình học là quá trình tự tổ chức (Self-Organizing). Điển hình cho kỹ thuật này là thuật toán Self-Organizing Map (SOM).

1.5.4. Mạng nơ-ron truyền thẳng và giải thuật lan truyền ngược

Mạng nơ-ron truyền thẳng (Feedforward Neural Network - FNN) là một kiến trúc mạng nơ-ron được sử dụng phổ biến. Đúng như tên của nó, giá trị sẽ đi thẳng từ input layer tới output layer chứ không có quay lại (khác với mạng nơ-ron hồi quy).



Hình 1.3 : Mạng nơ-ron lan truyền thẳng

Cũng giống như những mạng nơ-ron khác, việc học của FNN cũng chia làm 3 bước: lan truyền về phía trước (forward), đánh giá lỗi (evaluate) và cập nhật tham số (update).

Tại bước forward, như đã trình bày ở trên, dữ liệu sẽ đi từ input layer, qua những hidden layer để tới output layer. Những nơ-ron ở những layer khác nhau sẽ liên kết với nhau và có giá trị trọng số liên kết. Như vậy giữa những layer sẽ có ma trận trọng số. Vậy dữ liệu đi vào dưới dạng vector, đi tới hidden layer sẽ nhân với ma trận trọng số đầu vào rồi áp dụng hàm hoạt động, ta sẽ có vector biểu diễn tại hidden layer. Cụ thể:

$$h = W_I \times x$$

x : là input đầu vào dưới dạng vector cột.

h : là vector cột với mỗi giá trị tương ứng với một nơ-ron ở hidden.

W_I : là ma trận liên kết giữa input layer và hidden layer đầu tiên.

Mỗi W_{Iij} là trọng số liên kết giữa nơ-ron thứ j tại input layer (x_j)

và nơ-ron thứ i tại hidden layer (h_i).

Tương tự đi tới những hidden layer khác cho tới hidden layer cuối cùng. Từ đây dữ liệu sau khi áp dụng hàm tổng thay vì áp dụng hàm hoạt động thì sẽ sử dụng hàm softmax (hoặc sigmoid hay tanh):

$$o = softmax(W_O \times h)$$

h : là những giá trị tại hidden layer dưới dạng vector cột.

o : là vector cột với mỗi giá trị tương ứng với một nơ-ron ở output.

W_O : là ma trận trọng số liên kết hidden layer cuối và output layer.

Mỗi giá trị W_{Oij} là trọng số liên kết giữa nơ-ron thứ j tại hidden layer

(h_j) và nơ-ron thứ i tại output layer (o_i).

Tại bước đánh giá lỗi, hàm mất mát sẽ được áp dụng để tính lỗi cục bộ từng nơ-ron. Từ đó sẽ biết được lỗi xảy ra ở đâu để trong bước cập nhật sẽ thay đổi. Lỗi toàn cục sẽ là trung bình cộng của các lỗi cục bộ. Ví dụ áp dụng hàm mất mát log likelihood:

$$L = -z \ln o$$

L : là vector những giá trị lỗi.

z : là vector những giá trị thực tế là đúng.

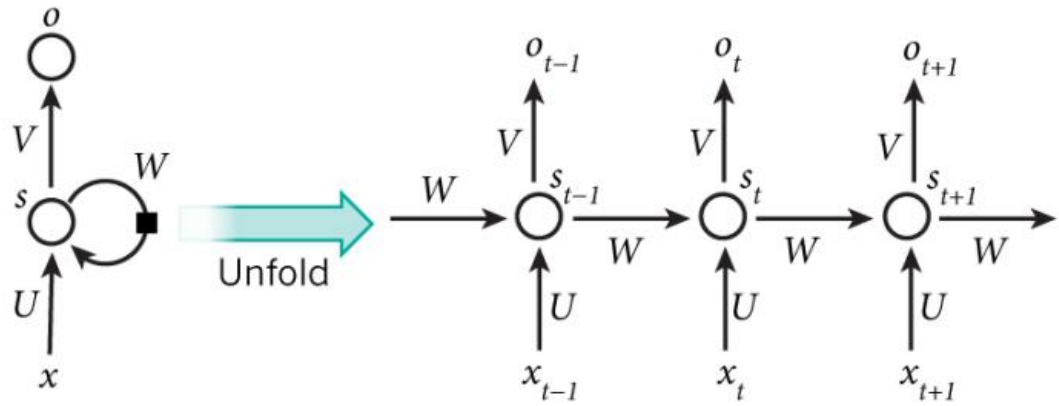
o : là vector những giá trị dự đoán (được tính toán ở bước forward).

Tại bước cập nhật sẽ sử dụng giải thuật lan truyền ngược (Backpropagation) để cập nhật các giá trị trọng số. Giải thuật lan truyền ngược dựa vào hàm lỗi để đánh giá lỗi cục bộ, từ đó tối ưu cục bộ bằng phương pháp tính các giá trị $\frac{\partial L}{\partial w_I}$, $\frac{\partial L}{\partial w_O}$. Phương pháp này được gọi là Gradient Descend.

Sau khi hoàn tất quá trình huấn luyện, ta sẽ có các ma trận trọng số liên kết. Để dự đoán cho một bản ghi nào đó ta chỉ cần thực hiện bước forward đối với cái trường thông tin của bản ghi đó để lấy ra kết quả ở output layer.

1.5.5. Mạng nơ-ron hồi quy và giải thuật lan truyền ngược xuyên tâm

Ý tưởng về mạng nơ-ron hồi quy xuất phát từ mục đích muốn chuyển hóa một chuỗi các input thành chuỗi các output, trong đó các thành phần trong chuỗi đều ảnh hưởng tới nhau. Ví dụ đối với bài toán chatbot, input là một câu (gồm nhiều từ và rõ ràng các từ phải liên quan tới nhau), mỗi từ được biểu diễn bằng một vector và ta mong muốn sử dụng mạng nơ-ron để ghi nhớ ngữ nghĩa của câu đó. Rõ ràng mạng FNN đã đề cập ở trên không thể làm được điều này vì đầu vào của FNN chỉ là một bản ghi và các bản ghi khác nhau hoàn toàn không ảnh hưởng lẫn nhau. Nhưng mạng nơ-ron hồi quy có thể làm được điều này.



Hình 1.4 : Mạng nơ-ron hồi quy

Từ input layer tới hidden layer:

$$s_t = \tanh(U \times x_t + W \times s_{t-1})$$

s_t : giá trị tại hidden state của input thứ t .

x_t : input đầu vào thứ t .

U, W : 2 ma trận tham số cần tìm ra qua quá trình huấn luyện.

Từ hidden layer tới output layer:

$$o_t = \text{softmax}(V \times s_t)$$

s_t : giá trị tại hidden state của input thứ t .

o_t : giá trị đầu ra của input thứ t .

Để huấn luyện cho RNN, ta cũng dùng phương pháp lan truyền ngược và gradient descend. Tuy nhiên do những input trước sẽ ảnh hưởng tới input sau nên ta sẽ thay đổi công thức cập nhật. Giải thuật để thực hiện như vậy gọi là lan truyền ngược xuyên thời gian (Backpropagation through time – BppTT) [3] [4].

Ta cần cập nhật tham số U, W, V theo phương pháp gradient descent nên cần tính $\frac{\partial L}{\partial w_1}, \frac{\partial L}{\partial w_2}, \dots, \frac{\partial L}{\partial w_n}$ để tìm ra công thức cập nhật ma trận.

Tuy nhiên RNN sẽ có một vấn đề trong trường hợp này. Đó là những input đầu tiên sẽ dần ít ảnh hưởng tới những input sau khi chuỗi input càng dài. Ví dụ trong bài toán hiểu ngữ nghĩa của một câu, từ đầu tiên sẽ ít ảnh hưởng tới những từ ở cuối nếu sử dụng RNN.

Nhưng trên thực tế thì không phải như vậy. Một từ ở đầu câu có thể quan trọng với một từ ở cuối câu hơn là một từ ở giữa câu. Ví dụ “The man who wear the blue T-shirt is speaking”. Trong ví dụ này, động từ “is speaking” là của chủ ngữ “the man” chứ không phải “the blue T-shirt” mặc dù “the blue T-shirt” gần với “is speaking” hơn. Vấn đề như vậy được gọi là “vanishing gradient”. Để giải quyết điều này, người ta đã cải tiến mô hình của RNN. Hai mô hình cải tiến thường sử dụng là mạng nơ-ron Long-Short Term Memory (LSTM) và mạng nơ-ron Gated Recurrent Units (GRU).

1.6. Biểu diễn vector từ

1.6.1 Bài toán

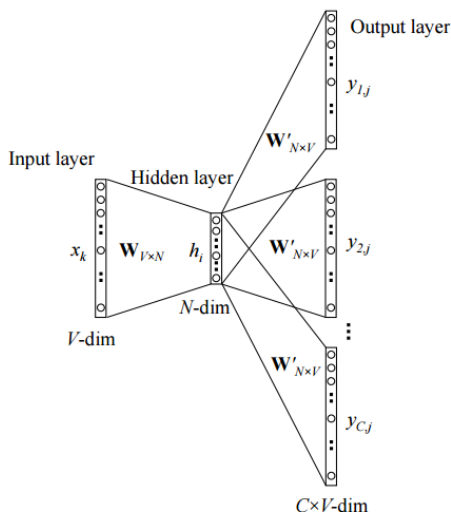
Để có thể áp dụng được khả năng ghi nhớ và dự đoán mạnh mẽ của học máy, điều đầu tiên cần làm là số hóa mỗi câu trong tập dữ liệu bằng cách biểu diễn mỗi từ dưới dạng một vector. Khi đó một câu sẽ là một ma trận gồm nhiều vector.

Cách biểu diễn thô sơ nhất là sử dụng one-hot vector [6]. Sau khi tách các từ riêng biệt từ tập dữ liệu, ta sẽ xây dựng tập từ vựng. One-hot vector sẽ biểu diễn mỗi từ dưới dạng một vector có số chiều bằng số lượng từ vựng và phần từ duy nhất có giá trị 1 là phần từ có cùng vị trí của từ đang xét trong tập từ vựng. Ví dụ từ “speak” có vị trí thứ 2 trong tập từ vựng. Vậy one-hot vector biểu diễn “speak” là $(0, 1, 0, \dots, 0)$.

One-hot vector là cách biểu diễn từ thô sơ nhất. Chính vì vậy vector biểu diễn không hề có ý nghĩa gì khác ngoài xác định đó là từ gì. Nhưng trên thực tế các từ khác nhau có thể sẽ có liên quan tới nhau về ngữ nghĩa. Để có thể biểu diễn từ dưới dạng vector mà vẫn giữ lại ngữ nghĩa, người ta đã xây dựng nên khái niệm mô hình ngôn ngữ.

Mô hình ngôn ngữ thực tế là một mô hình học máy. Nó sẽ học và biểu diễn những từ gần nghĩa dưới dạng những vector có độ tương quan cao. Ý tưởng cơ bản của mô hình ngôn ngữ là những từ thường xuất hiện cùng với một số từ thì có thể sẽ gần nghĩa với nhau. Chính vì vậy các mô hình thường tìm cách ước lượng xác suất xuất hiện đồng thời của các từ. Như vậy nếu hai từ có xác suất xuất hiện đồng thời với các từ khác tương đối giống nhau thì hai từ đó có thể sẽ gần nghĩa. Các mô hình ngôn ngữ tiêu biểu là N-grams, Continuous Bag Of Words (CBOW) và Skip grams. Báo cáo này sẽ giới thiệu về mô hình ngôn ngữ Skip grams.

1.6.2 Mô hình Skip grams



Hình 1.5: Mô hình Skip Grams

Ý tưởng của mô hình Skip Grams (SG)[6] là sử dụng mạng nơ-ron lan truyền thẳng để xây dựng mô hình ngôn ngữ. Mô hình này sẽ tính xác suất xuất hiện đồng thời của những từ xung quanh nó trong tập dữ liệu bằng cách huấn luyện máy tính dựa vào một từ làm input có thể gợi ý ra những từ xung quanh. Điều này dẫn đến những từ hay xuất hiện cùng một số từ khác thì có thể sẽ tương đương nhau về ngữ nghĩa. Như vậy sau quá trình huấn luyện, ma trận W' (như hình trên) sẽ thể hiện độ quan trọng của từng nơ-ron trong hidden layer với từng từ, và W' cũng chính là biểu diễn dạng vector của tất cả các từ trong tập dữ liệu.

Các bước huấn luyện cho Skip grams tương tự như huấn luyện cho mô hình FNN. Tập dữ liệu đầu vào có thể là đoạn văn bản hay đoạn hội thoại, sau đó từng từ sẽ đi qua FNN dưới dạng one-hot vector. Sau đó sử dụng lan truyền ngược để huấn luyện cho từng từ trong tập dữ liệu có thể dự đoán ra những từ xung quanh nó.

1.7 Kết chương

Chương này đã giới thiệu các công nghệ liên quan có thể áp dụng để xây dựng một hệ thống trả lời tự động (Chatbot). Ưu nhược điểm của các phương pháp tiếp cận. Giới thiệu tổng quát về các phương pháp học máy, các mạng nơ-ron cơ bản và phương pháp huấn luyện. Qua chương này chúng ta thấy được mô hình động có ưu điểm rất tốt để xây dựng

hệ thống trả lời tự động. Chương tiếp theo sẽ trình bày chi tiết về mạng nơ-ron LSTM và mô hình Sequence to Sequence để sinh câu trả lời tự động.

CHƯƠNG 2: MẠNG NƠ-RON LSTM VÀ MÔ HÌNH SEQUENCE TO SEQUENCE

Trong Chương này, chúng ta sẽ được giới thiệu về mạng nơ-ron LSTM, tác dụng và áp dụng của LSTM trong mô hình Sequence to Sequence.

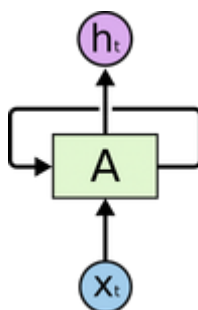
2.1. Mạng nơ-ron LSTM

2.1.1. Tổng quan về mạng LSTM

Để đọc hiểu một câu hay một đoạn văn, con người chúng ta không quên hết những thông tin trước đó mà sẽ xâu chuỗi lại các dữ liệu đã đọc để làm rõ ý nghĩa cho thông tin hiện tại. Tương tự như việc ôn bài, việc làm này giúp các liên kết bên trong não được khắc sâu và tổ chức lại thông tin cho việc truy xuất và xử lý thông tin trong tương lai được rõ ràng và nhanh chóng hơn.

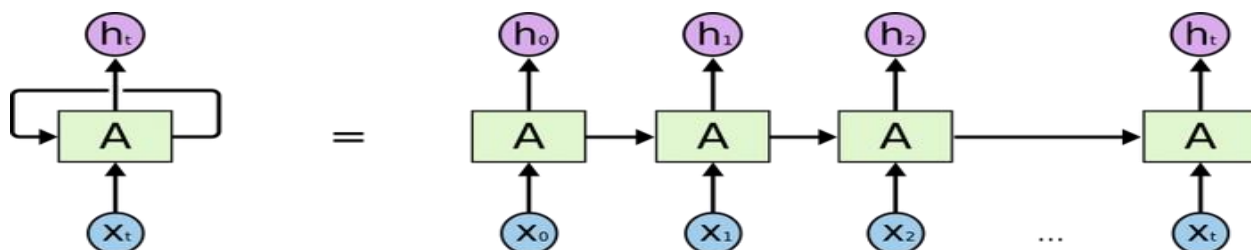
Các mô hình Neural Networks truyền thống không thể làm được điều này. Ví dụ, trong bài toán phân lớp sự kiện cho một đoạn video theo từng giây. Thật khó để một mô hình như vậy có thể dựa vào thông tin trước đó để phân lớp.

RNN giải quyết vấn đề này bằng cách tạo ra các mạng vòng lặp bên trong chúng, cho phép thông tin được lưu trữ lại cho các lần phân tích tiếp theo.



Hình 2.1 Mô hình RNN tổng quát

Trong biểu đồ trên, A nhận thông tin của x_t tại thời điểm t và phản hồi lại tương ứng kết quả đầu ra h_t tại thời điểm t . Ta có thể rẽ vòng lặp trên thành một tiến trình để dễ hình dung hơn.

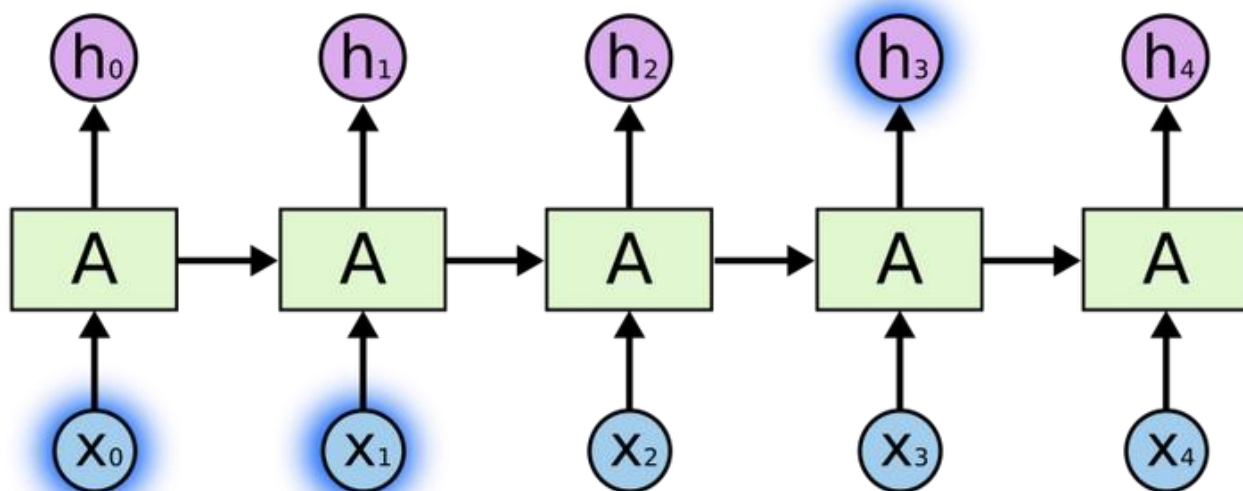


Hình 2.2 Mô hình chi tiết của RNN

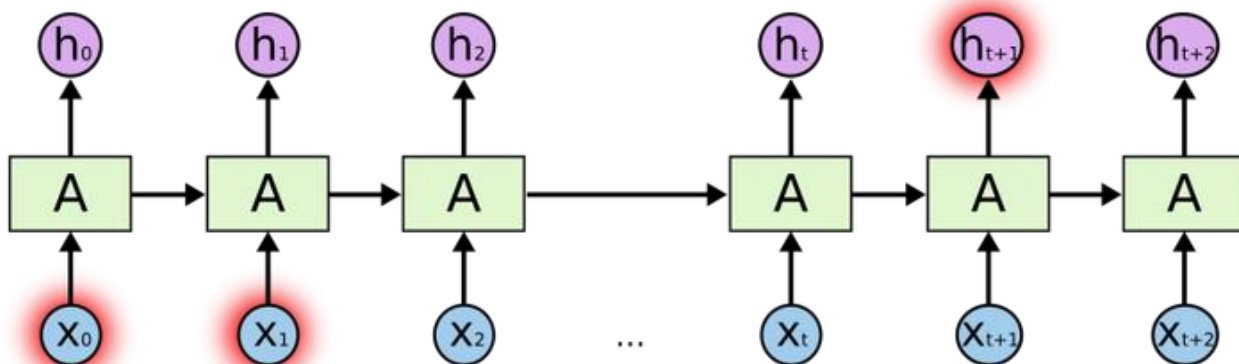
Thay vì chỉ nhận một đầu vào là x như các mạng neural truyền thống, RNN hình thành nên một chuỗi các đầu vào A cùng với x_t tại thời điểm t .

2.1.2 Vấn đề phụ thuộc (Long-Term Dependencies)

Một trong những ý tưởng khởi thủy của RNN [7] là kết nối những thông tin trước đó nhằm hỗ trợ cho các xử lý hiện tại. Nhưng đôi khi, ta chỉ cần dựa vào một số thông tin gần nhất để thực hiện tác vụ hiện tại. Ví dụ, bạn cố gắng dự đoán từ tiếp theo dựa vào các từ trước đó. Nếu chúng ta dự đoán từ cuối cùng trong câu “đám mây bay trên bầu trời”, thì chúng ta không cần truy tìm quá nhiều từ trước đó, ta có thể đoán ngay từ tiếp theo sẽ là “bầu trời”. Trong trường hợp này, khoảng cách tới thông tin liên quan được rút ngắn lại.



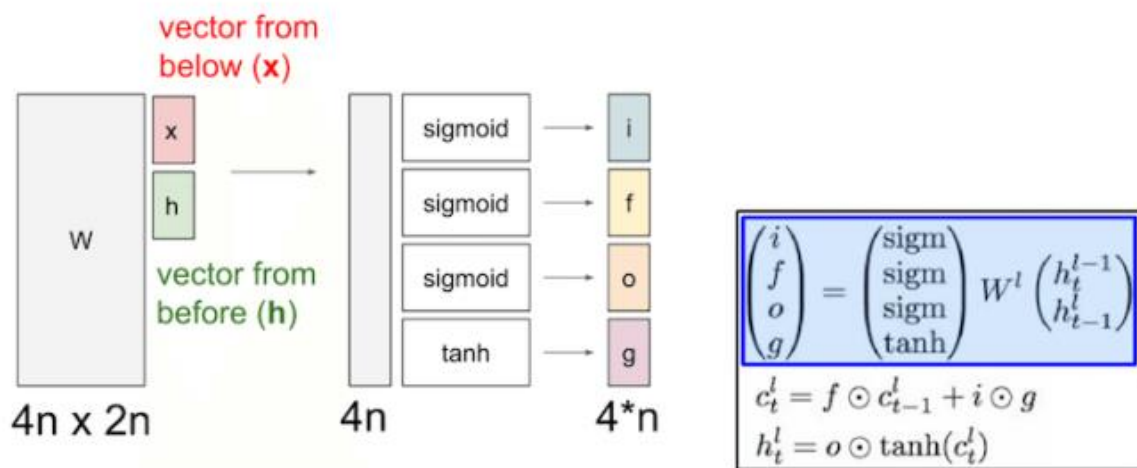
Hình 2.3 : Phụ thuộc ngắn



Hình 2.4 : Phụ thuộc dài

Về lý thuyết, RNN hoàn toàn có khả năng xử lý “long-term dependencies”, nghĩa là thông tin hiện tại có được là nhờ vào chuỗi thông tin trước đó. Thật không may, trong thực tế, RNN dường như không có khả năng này. Vấn đề này đã được Hochreiter (1991) [German] and Bengio, et al. (1994) đưa ra như một thách thức cho mô hình RNN.

Trong những năm 1990, RNN phải đối diện với hai thách thức lớn đó là Vanishing và Exploding Gradients ảnh hưởng lớn đến hiệu suất của mô hình. Vấn đề này phát sinh trong quá trình huấn luyện.



Hình 2.5 : Các hàm xử lý trong LSTM

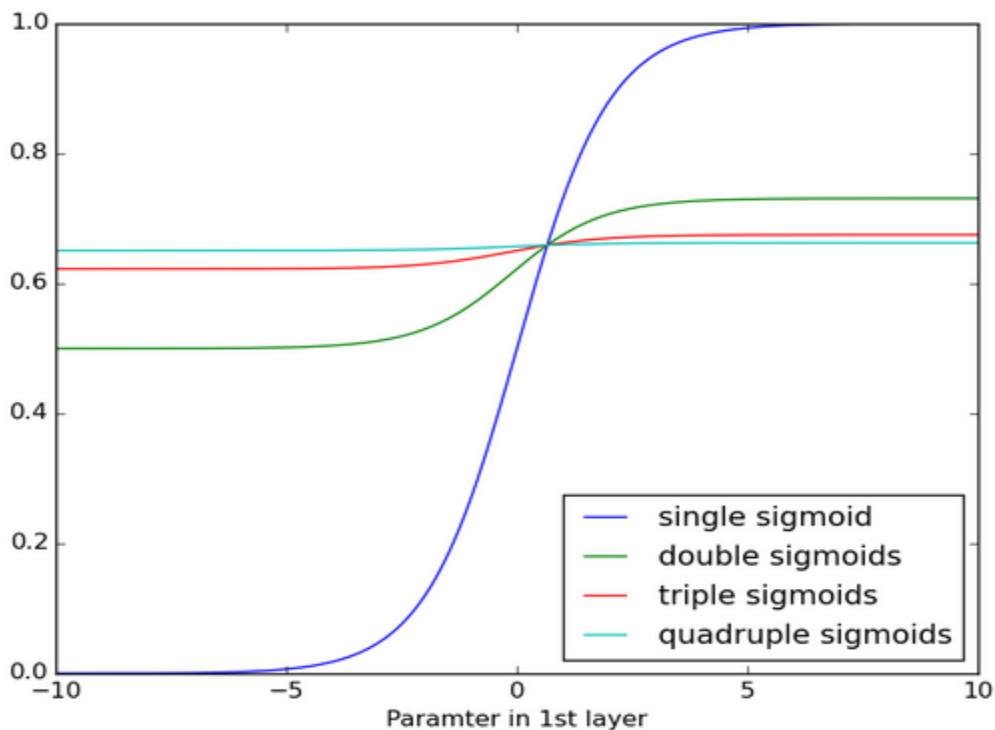
Nếu trọng số của ma trận W nhỏ (trị riêng trọng số của ma trận nhỏ hơn 1.0), điều này sẽ dẫn đến trường hợp được gọi là vanishing gradients khi gradient signal ngày càng

nhỏ/tan biến theo quá trình huấn luyện, khiến cho quá trình tối ưu hoá hàm lỗi hội tụ chậm hoặc dừng hẳn.

Ngược lại, nếu trọng số của ma trận W lớn (trị riêng trọng số của ma trận lớn hơn 1.0), điều này sẽ dẫn đến trường hợp được gọi là exploding gradients khi gradient signal ngày càng bị phân tán trong quá trình huấn luyện, khi đó quá trình tối ưu hoá hàm lỗi không hội tụ.

Recurrent nets tìm kiếm và hình thành mối liên kết giữa final output và các input event thông qua nhiều bước trước khi kết thúc quá trình huấn luyện. Vấn đề đặt ra là các thông tin input trước đó cần đặt trọng số tương ứng là bao nhiêu. Do vậy, đối với câu huấn luyện càng dài thì thông tin trước đó ngày càng bị nhiễu hoặc bị che lấp. Khi thực hiện quá nhiều phép toán nhân ma trận liên tục xuyên suốt chiều dài của chuỗi thì hiệu ứng vanishing/exploding sẽ xuất hiện.

Hình minh hoạ dưới đây cho ta thấy hiệu ứng khi áp dụng liên tiếp hàm sigmoid. Dữ liệu thu được ngày càng tan biến dần cho đến lúc không còn nhận ra được nữa. Tương tự như gradient vanishing khi ta truyền dữ liệu qua nhiều layer.

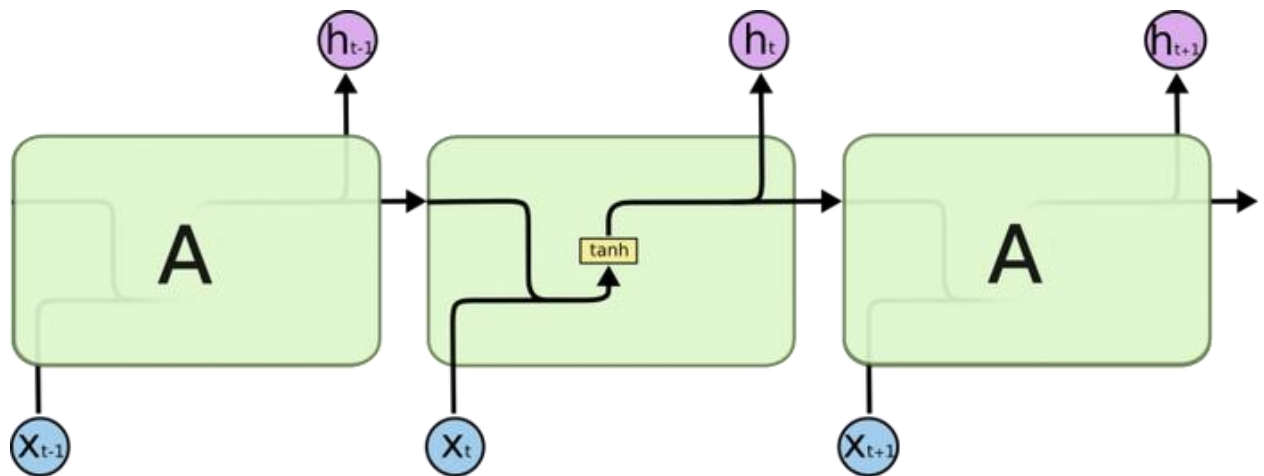


Hình 2.6: Biểu đồ sự biến thiên khi sử dụng các hàm sigmoids khác nhau

2.1.3. Mạng LSTM

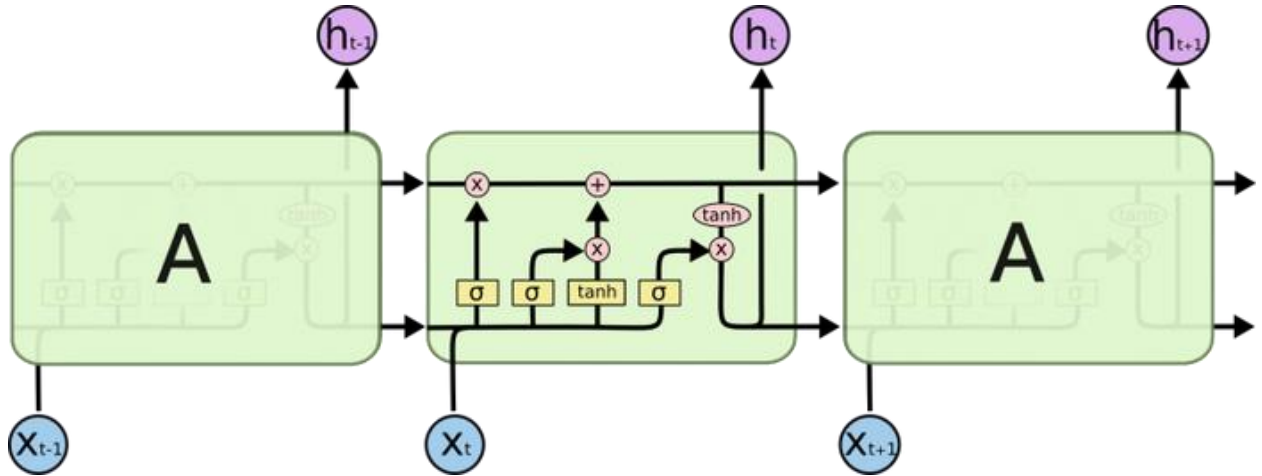
Long Short Term Memory networks [9] – thường được gọi là “LSTM”, là trường hợp đặc biệt của RNN, có khả năng học long-term dependencies. Mô hình này được giới thiệu bởi Hochreiter & Schmidhuber (1997), và được cải tiến lại. Sau đó, mô hình này dần trở nên phổ biến nhờ vào các công trình nghiên cứu gần đây. Mô hình này có khả năng tương thích với nhiều bài toán nên được sử dụng rộng rãi ở các ngành liên quan.

LSTM được thiết kế nhằm loại bỏ vấn đề long-term dependency. Trước khi đi vào LSTM, ta quan sát lại mô hình RNN bên dưới, các layer đều mắc nối với nhau thành các module neural network. Trong RNN chuẩn, repeating module này có cấu trúc rất đơn giản chỉ gồm một single *tanh* layer.



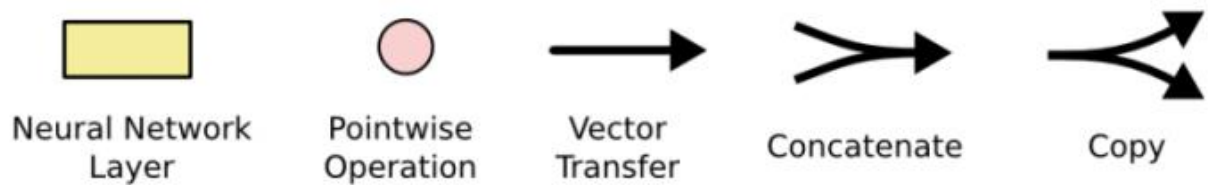
Hình 2.7 : Vị trí của hàm *tanh* trong LSTM

LSTM cũng có cấu trúc mắc xích tương tự, nhưng các repeating module có cấu trúc khác hẳn. Thay vì chỉ có một layer neural network, ta có tới bốn layer, tương tác với nhau theo một cấu trúc cụ thể.



Hình 2.8 : Tương tác giữa các layer trong LSTM

Trước tiên, ta hãy làm quen với ký hiệu được sử dụng.



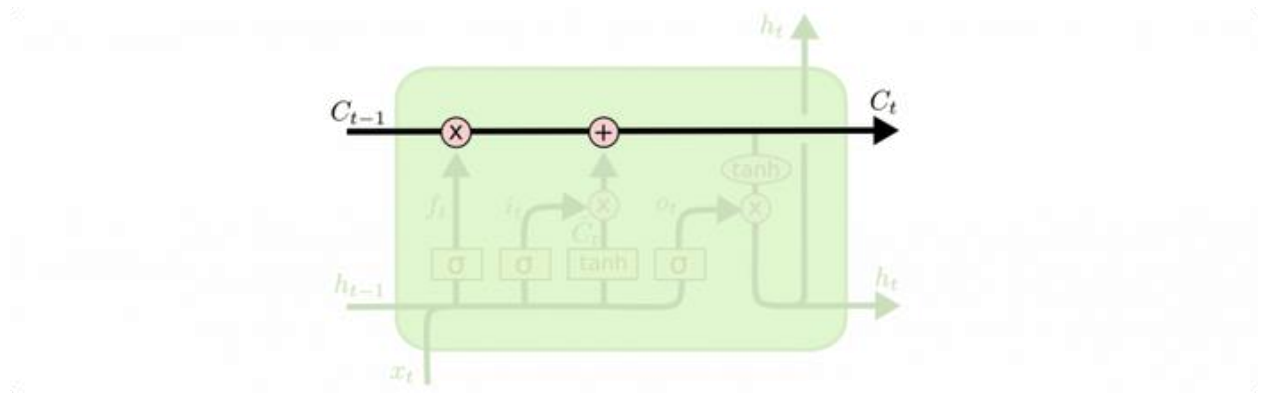
Hình 2.9 : Các ký hiệu được sử dụng trong mạng LSTM

Ở biểu đồ trên, hình tròn nền hồng biểu diễn pointwise operations, ví dụ cộng vector. Hình hộp nền vàng là các neural network layers được huấn luyện. Đường kẻ gộp lại nhau biểu thị cho concatenate, trong khi đó đường rẽ nhánh biểu thị cho sự sao chép từ vị trí này sang vị trí khác.

➤ Ý tưởng của mạng LSTMs

Sau khi quan sát mô hình thiết kế của LSTM, bạn sẽ nhận ra ngay, đây là một bảng mạch số, gồm các mạch logic và các phép toán logic trên đó. Thông tin, hay nói khác hơn là tần số của dòng điện di chuyển trong mạch sẽ được lưu trữ, lan truyền theo cách mà chúng ta thiết kế bảng mạch.

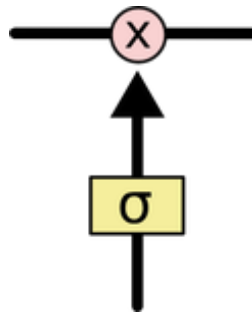
Mấu chốt của LSTM là cell state (tế bào trạng thái), đường kẻ ngang chạy dọc ở trên top diagram. Cell state giống như băng chuyền. Nó chạy xuyên thẳng toàn bộ mạch xích, chỉ một vài tương tác nhỏ tuyến tính (minor linear interaction) được thực hiện. Điều này giúp cho thông tin ít bị thay đổi xuyên suốt quá trình lan truyền.



Hình 2.10 : Cell state trong LSTM

LSTM có khả năng thêm hoặc bớt thông tin vào cell state, được quy định một cách cẩn thận bởi các cấu trúc gọi là cổng (gate).

Các cổng này là một cách (tùy chọn) để định nghĩa thông tin bằng qua. Chúng được tạo bởi sigmoid neural net layer và một pointwise multiplication operation.

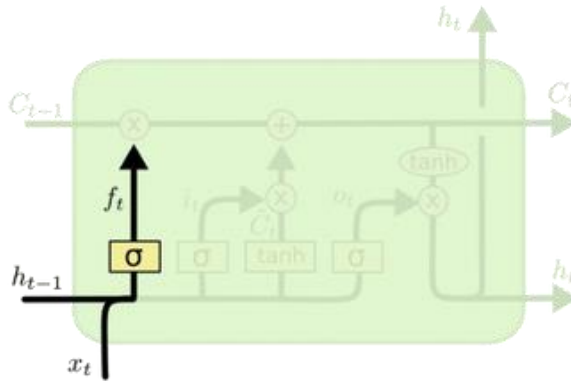


Hình 2.11 Cổng trong LSTM

Sigmoid layer outputs có giá trị từ 0 – 1, mô tả độ lớn thông tin được phép truyền qua tại mỗi component. Nếu ta thu được zero điều này có nghĩa là “không cho bất kỳ cái gì đi qua”, ngược lại nếu thu được giá trị là một thì có nghĩa là “cho phép mọi thứ đi qua”. Một LSTM có ba cổng như vậy để bảo vệ và điều khiển cell state.

2.1.4. Phân tích mô hình LSTM

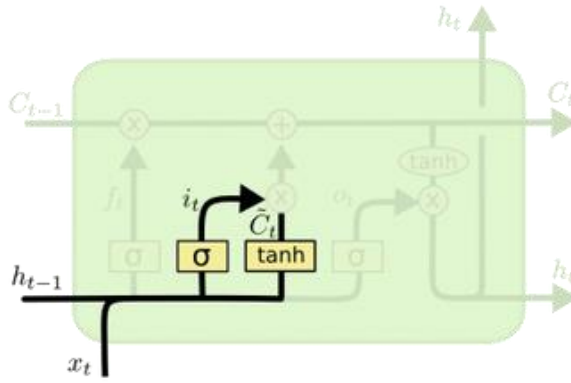
Bước đầu tiên của mô hình LSTM là quyết định xem thông tin nào chúng ta cần loại bỏ khỏi cell state. Tiến trình này được thực hiện thông qua một sigmoid layer gọi là “forget gate layer” – cánh cổng quên lãng. Đầu vào là h_{t-1} và x_t , đầu ra là một giá trị nằm trong khoảng $[0, 1]$ cho cell state C_{t-1} . 1 tương đương với “giữ lại thông tin”, 0 tương đương với “loại bỏ thông tin”.



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

Hình 2.12 : Cổng lãng quên (forget gate layer)

Bước tiếp theo, ta cần quyết định thông tin nào cần được lưu lại tại cell state. Ta có hai phần. Một, single sigmoid layer được gọi là “input gate layer” quyết định các giá trị chúng ta sẽ cập nhật. Tiếp theo, một *tanh* layer tạo ra một vector ứng viên mới, \tilde{C}_t , được thêm vào trong cell state.

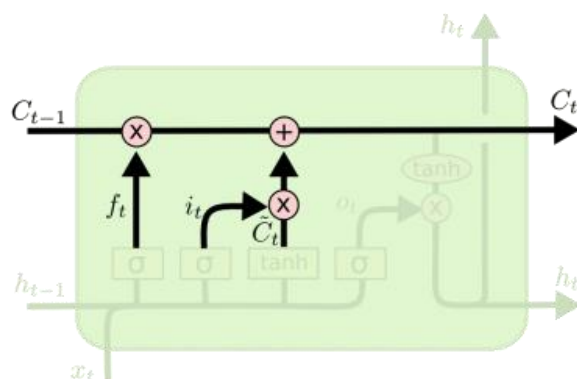


$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$

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

Hình 2.13 : Quyết định thông tin được lưu tại Cell state

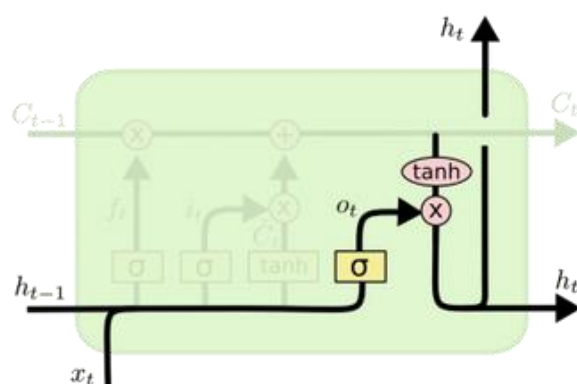
Ở bước tiếp theo, ta sẽ kết hợp hai thành phần này lại để cập nhật vào cell state. Lúc cập nhật vào cell state cũ, C_{t-1} , vào cell state mới C_t . Ta sẽ đưa state cũ hàm f_t , để quên đi những gì trước đó. Sau đó, ta sẽ thêm $(i_t * \tilde{C}_t)$. Đây là giá trị ứng viên mới, co giãn (scale) số lượng giá trị mà ta muốn cập nhật cho mỗi state.



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

Hình 2.14 Cập nhật vào Cell state

Cuối cùng, ta cần quyết định xem thông tin output là gì. Output này cần dựa trên cell state của chúng ta, nhưng sẽ được lọc bớt thông tin. Đầu tiên, ta sẽ áp dụng single sigmoid layer để quyết định xem phần nào của cell state chúng ta dự định sẽ output. Sau đó, ta sẽ đẩy cell state qua *tanh* (đẩy giá trị vào khoảng -1 và 1) và nhân với một output sigmoid gate, để giữ lại những phần ta muốn output ra ngoài.



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

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

Hình 2.15 : Quyết định thông tin output ra là gì ?

2.2. Mô hình Sequence to Sequence trong dịch máy

2.2.1 Giới thiệu

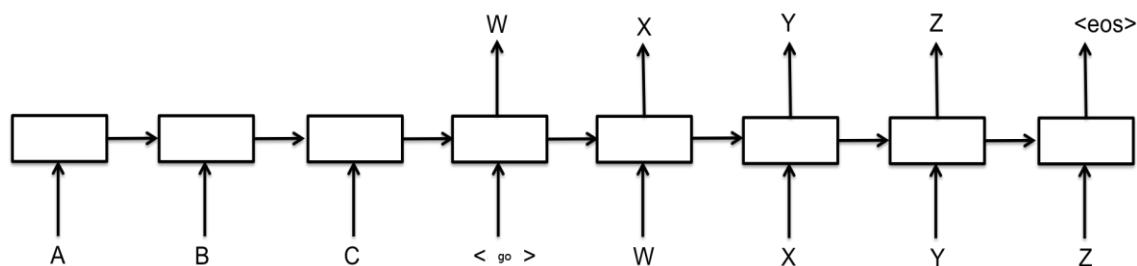
Deep Neural Networks (DNNs) là mô hình cực kỳ mạnh mẽ, đạt hiệu suất cái khi áp dụng vào các vấn đề khó như nhận dạng giọng nói và nhận dạng đối tượng. DNNs mạnh mẽ bởi vì nó có thể thực hiện tính toán song song tùy ý cho một số lượng các bước.

Một ví dụ đáng kinh ngạc về sức mạnh của DNNs là khả năng sắp xếp N số N bit chỉ sử dụng 2 lớp ẩn kích thước căn bậc hai. Vì vậy trong khi mạng lưới thần kinh có liên quan đến mô hình thống kê thông thường, nó học được một tính toán phức tạp. Hơn nữa, DNNs lớn có thể được huấn luyện có giám sát với giải thuật lan truyền ngược bất cứ khi nào tập huấn luyện được dán nhãn có đủ thông tin để xác định các thông số của mạng. Do đó nếu có tồn tại một thiết lập thông số của một DNN lớn mà đạt được kết quả tốt thì với giải thuật lan truyền ngược sẽ tìm thấy thông số và giải quyết vấn đề.

Mặc dù có sự linh hoạt và sức mạnh nhưng DNNs chỉ có thể áp dụng cho các vấn đề mà các đầu vào và mục tiêu có thể mã hóa thành vector có số chiều cố định. Đây là một hạn chế đáng kể vì nhiều vấn đề quan trọng được thể hiện tốt nhất với trình tự có độ dài không biết trước.

Ví dụ: Nhận dạng giọng nói và dịch máy là vấn đề tuần tự. Tương tự như vậy, hệ thống trả lời tự động có thể xem như là ánh xạ chuỗi các từ đại diện cho các câu hỏi để tìm một chuỗi các từ đại diện cho câu trả lời.

Chuỗi đặt ra một thách thức đối với DNNs vì nó yêu cầu đầu vào và đầu ra được biết và cố định. LSTM có thể giải quyết được trình tự các từ trong câu để xử lý chuỗi vấn đề. Ý tưởng là sử dụng một LSTM để đọc chuỗi đầu vào, tại một thời điểm để có được một vector có số chiều cố định lớn và sau đó sử dụng một LSTM để trích xuất các đầu ra từ vector đó. Các LSTM thứ hai thực chất là sử dụng mô hình RNN ngoại trừ nó được điều khiển trên các chuỗi đầu vào. Khả năng của LSTM học thành công trên dữ liệu với phụ thuộc theo khoảng cách cho nó một lựa chọn tự nhiên cho ứng dụng.



Hình 2.16 : Mô hình của chúng tôi đọc một câu đầu vào "ABC" và sản xuất "WXYZ" là câu đầu ra. Mô hình này dừng lại đưa ra dự đoán sau khi xuất ra token cuối cùng của câu (EOS). Lưu ý rằng các LSTM đọc câu đầu vào ngược lại, bởi vì làm như vậy đưa ra

nhiều phụ thuộc ngắn hạn trong các dữ liệu mà làm cho các vấn đề tối ưu hóa dễ dàng hơn nhiều.

LSTM không bị ảnh hưởng với câu rất dài, chúng có thể làm tốt các câu dài bởi vì chúng đảo ngược thứ tự các câu trong dữ liệu đầu vào trong việc đào tạo và kiểm tra thiết lập. Một điểm có ích của LSTM là nó có thể học ánh xạ một câu đầu vào có chiều dài không cố định thành một vector đại diện có số chiều cố định. Biết rằng bản dịch có xu hướng diễn giả các câu nguồn, khuyến khích LSTM để tìm các mô tả câu đó nhằm nắm bắt ý nghĩa của chúng, làm câu có ý nghĩa tương tự là gần nhau trong khi các câu khác ở xa. Một đánh giá cho thấy rằng mô hình Sequence to Sequence là nhận thức được trật tự và khả năng bất biến đối với các câu nói chủ động và thụ động.

2.2.2 Mô hình Sequence to Sequence

RNN là một sự tổng quát tự nhiên của các mạng lan truyền thẳng đến các chuỗi. Với một chuỗi có yếu tố đầu vào ($X_1 \rightarrow X_t$), một RNN tiêu chuẩn tính toán một chuỗi kết quả đầu ra ($Y_1 \rightarrow Y_t$). Bằng cách thực hiện các phương trình :

$$h_t = \text{sigm}(W^{hx}x_t + W^{hh}h_{t-1})$$

$$y_t = W^{yh}h_t$$

Các RNN có thể dễ dàng ánh xạ Sequence to Sequence bất cứ khi nào sự liên kết giữa đầu vào và đầu ra được biết trước. Tuy nhiên, nó không phải là rõ ràng làm thế nào để áp dụng một RNN để xử lý đầu vào và đầu ra có độ dài khác nhau với các mối quan hệ phức tạp.

Chiến lược đơn giản là biến đổi chuỗi đầu vào thành một vector có số chiều cố định bằng một RNN. RNN cũng cấp tất cả các thông tin có liên quan, RNNs sẽ rất khó để đào tạo do sự phụ thuộc kết quả lâu dài. Tuy nhiên, mạng LSTM được biết đến để học các vấn đề với phụ thuộc thời gian dài hạn. Nên LSTMs có thể áp dụng thành công vào mô hình này.

Mục tiêu của LSTM là ước lượng xác suất có điều kiện của câu trả lời khi có câu đầu vào:

$$P(y_1 \dots y_m | x_1 \dots x_n)$$

Khi $(x_1 \dots x_n)$ là chuỗi đầu vào và $(y_1 \dots y_m)$ là chuỗi đầu ra tương ứng với chiều dài m có thể khác n . LSTM tính bằng cách: Có được số chiều cố định đặc trưng W đưa ra bởi

trạng thái ẩn cuối cùng của LSTM và sau đó tính xác suất của $(y_1 \dots y_m)$ với công thức của mạng LSTM chuẩn mà trạng thái ẩn ban đầu được khởi tạo là vector đặc trưng W của $(x_1 \dots x_n)$.

$$P(y_1 \dots y_m | x_1 \dots x_n) = \prod_{t=1}^m P(y_t | W, y_1 \dots y_{t-1})$$

Phương trình này mỗi $P(y_t | W, y_1 \dots y_{t-1})$ phân bố thực hiện với một hàm Soft max trên tất cả các từ vựng. Chúng ta sử dụng công thức tính của LSTM từ Graves. Lưu ý rằng chúng ta yêu cầu mà kết thúc là “EOS”, cho phép các mô hình xác định một phân phối qua các chuỗi có độ dài khác nhau. Chương trình tổng thể được trình bày trong hình nơi LSTM thể hiện tính đại diện của ‘A’ ‘B’ ‘C’ ‘EOS’ và sau đó tính xác suất của ‘W’ ‘X’ ‘Y’ ‘Z’ ‘EOS’.

Mô hình thực tế chúng ta sử dụng khác với mô tả ở trên ba điểm quan trọng. Đầu tiên, chúng ta sử dụng hai mạng LSTM khác nhau một cho chuỗi đầu vào và một cho chuỗi đầu ra. Bởi vì làm như vậy làm tăng các thông số mô hình với chi phí tính toán không đáng kể và làm cho nó tự nhiên hơn trong quá trình huấn luyện giữa các cặp ngôn ngữ khác. Thứ hai, ta thấy rằng LSTM sâu tốt hơn đáng kể LSTM cạn. Thứ ba, chúng ta thấy rằng việc đảo ngược câu đầu vào rất có giá trị. Ví dụ: thay vì ánh xạ các từ trong câu đầu vào a,b,c với α, μ, ∞ thì các LSTM được yêu cầu ánh xạ c,b,a với α, μ, ∞ nơi mà α, μ, ∞ là bản dịch của a,b,c.

2.2.3. Mã hóa và đánh giá

Cốt lõi của các thí nghiệm của chúng ta liên quan đến đào tạo một LSTM sâu rộng trên nhiều cặp câu. Chúng ta đào tạo bằng cách tối đa hóa khả năng ghi nhớ của một bản dịch đúng T với câu nguồn S . Do đó mục tiêu đào tạo là :

$$\frac{1}{|S|} \sum_{(T,S) \in s} \log P(T|S)$$

Trong đó s là tập huấn luyện khi đào tạo hoàn tất chúng ta tìm bản dịch bằng cách tìm bản dịch có khả năng theo quy định của LSTM

$$T = \arg \max_T P(T|S)$$

Chúng ta tìm kiếm các bản dịch có nhiều khả năng bằng cách sử dụng một bộ giải mã tìm kiếm đơn giản (beam search) chọn B từ có khả năng xuất hiện trong bản dịch, mà giả thuyết một phần là tiền tố của bản dịch. Tại mỗi bước thời gian chúng ta mở rộng thêm số từ có thể xuất hiện trong bản dịch từ tập từ vựng. Điều này làm tăng đáng kể số lượng giả thuyết. Ngay khi gặp “EOS” được gắn vào giả thuyết nó lấy được các từ trong tập giả thuyết để sinh ra bản dịch tốt nhất.

2.2.3. Đảo ngược câu nguồn

Trong khi LSTM có khả năng giải quyết vấn đề với phụ thuộc lâu dài, chúng ta phát hiện ra rằng LSTM học tốt hơn nhiều khi câu nguồn được đảo ngược (các câu mục tiêu không được đảo ngược). Bằng cách làm như vậy độ lỗi của LSTM đã giảm và các điểm BLEU của bản dịch được giải mã của nó tăng lên.

Trong khi chúng ta không có một lời giải thích đầy đủ cho hiệu tượng này thì chúng ta tin rằng nó được gây ra bởi sự ra đời của phụ thuộc ngắn hạn cho các tập dữ liệu. Thông thường khi chúng ta ghép một câu nguồn với một câu đích, mỗi từ trong câu nguồn là ảnh xạ tương ứng của nó trong câu mục tiêu. Kết quả là sự phụ thuộc về khoảng cách trung bình lớn. Bằng cách đảo ngược từ trong câu nguồn khoảng cách trung bình giữa các từ tương ứng trong ngôn ngữ nguồn và đích là không thay đổi. Tuy nhiên, một vài từ đầu tiên trong ngôn ngữ nguồn bây giờ đang rất gần với một vài từ đầu tiên của ngôn ngữ đích vì vậy khoảng thời gian tối thiểu của quá trình huấn luyện giảm đáng kể.

2.3 Kết chương

Việc sử dụng mạng LSTM trong mô hình Sequence to Sequence đã đem lại nhiều lợi ích trong các mô hình dịch máy và có khả năng phát triển áp dụng trong các hệ thống trả lời tự động. Chi tiết cài đặt và thử nghiệm mô hình sẽ được giới thiệu trong chương tiếp theo.

CHƯƠNG 3: CÀI ĐẶT VÀ THỬ NGHIỆM MỘT HÌNH SEQUENCE TO SEQUENCE TRONG BÀI TOÁN TRẢ LỜI TỰ ĐỘNG

3.1. Cấu trúc dữ liệu

3.1.1 Định dạng dữ liệu

Tất cả dữ liệu dùng để thử nghiệm thuật toán trong bài toán Hệ thống trả lời tự động tuân theo cấu trúc sau:

Câu hỏi 1

Câu trả lời 1

Câu hỏi 2

Câu trả lời 2

...

Ví dụ:

Xin chào

Xin chào, tôi có thể giúp gì bạn

Bạn bao nhiêu tuổi ?

Tôi 23 tuổi

...

3.1.2 Tiền xử lý dữ liệu

➤ Bài toán tách từ trong Tiếng Việt

Như chúng ta đã biết, văn bản tiếng Việt đặt dấu cách giữa các âm tiết chứ không phải giữa các từ. Một từ có thể có một, hai hoặc nhiều âm tiết nên có nhiều cách phân chia các âm tiết thành các từ, gây ra nhập nhằng. Việc phân giải nhập nhằng này gọi là bài toán tách từ.

Tiêu chí quan trọng nhất trong bài toán tách từ đương nhiên là độ chính xác. Hiện tại người ta đã đạt được độ chính xác lên đến 97% tính theo từ. Tuy nhiên nếu tính theo câu (số câu được tách hoàn toàn đúng/tổng số câu) thì độ chính xác chỉ khoảng 50%. Đây là vấn đề nghiêm trọng đối với các bước xử lý sau như phân tích ngữ pháp, ngữ nghĩa vì một từ bị tách sai có ảnh hưởng toàn bộ đến cách phân tích cả câu.

Ngoài ra tiêu chí độ chính xác tách từ mới cũng quan trọng với các ứng dụng thực tế. Tiếng Việt là một sinh ngữ – nó luôn luôn biến đổi. Các từ mới thuần Việt cũng như vay mượn được tạo ra hàng ngày. Nếu một ứng dụng không xử lý được những từ này thì hiệu năng của nó sẽ giảm dần theo thời gian.

Hiện tại có một số cách tiếp cận bài toán tách từ như sau:

- Ghép cục đại: Đặt các từ vào câu sao cho phủ hết được câu đó, thỏa mãn một số heuristic nhất định. Phương pháp này các ưu điểm là rất nhanh, nhưng có rất nhiều hạn chế, ví dụ như độ chính xác thấp, không xử lý được những từ không có trong từ điển.
- Luật: Xây dựng tập luật bằng tay hoặc tự động để phân biệt các cách kết hợp được phép và không được phép.
- Đồ thị hoá: Xây dựng một đồ thị biểu diễn câu và giải bài toán tìm đường đi ngắn nhất trên đồ thị.
- Máy học: Coi như bài toán gán nhãn chuỗi. Cách này được sử dụng trong JVNSegmenter, Đông du.
- Dùng mô hình ngôn ngữ: Cho trước một số cách tách từ của toàn bộ câu, một mô hình ngôn ngữ có thể đánh giá được cách nào có khả năng cao hơn. Đây là cách tiếp cận của vnTokenizer[14].

➤ **Thư viện vnTokenizer**

Vntokenizer là bộ công cụ xử lý tiếng Việt được phát triển bởi Lê Hồng Phương, giảng viên trường Đại học Công nghệ, Đại học Quốc gia Hà Nội. Bộ công cụ được sử dụng để phân biệt các từ trong tiếng Việt với độ chính xác từ 96%-98%.

Ví dụ:

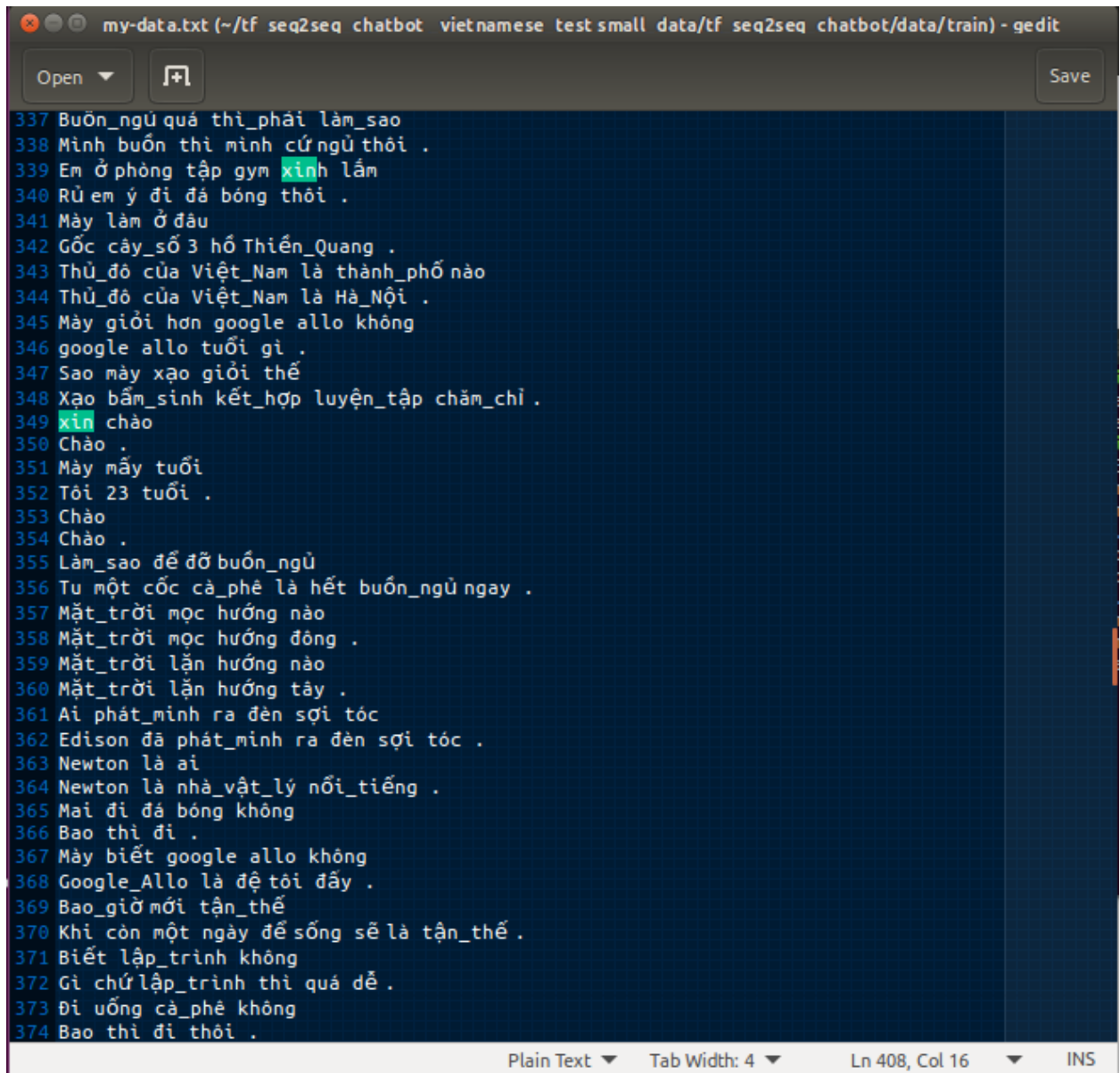
Bạn bao nhiêu tuổi ? → Bạn bao_nhiêu tuổi ?

Cách đăng ký Samsung Galaxy Apps → Cách_đăng_ký Samsung_Galaxy_Apps

3.2. Thử nghiệm mô hình “Sequence to Sequence” sử dụng mạng nơ-ron LSTM

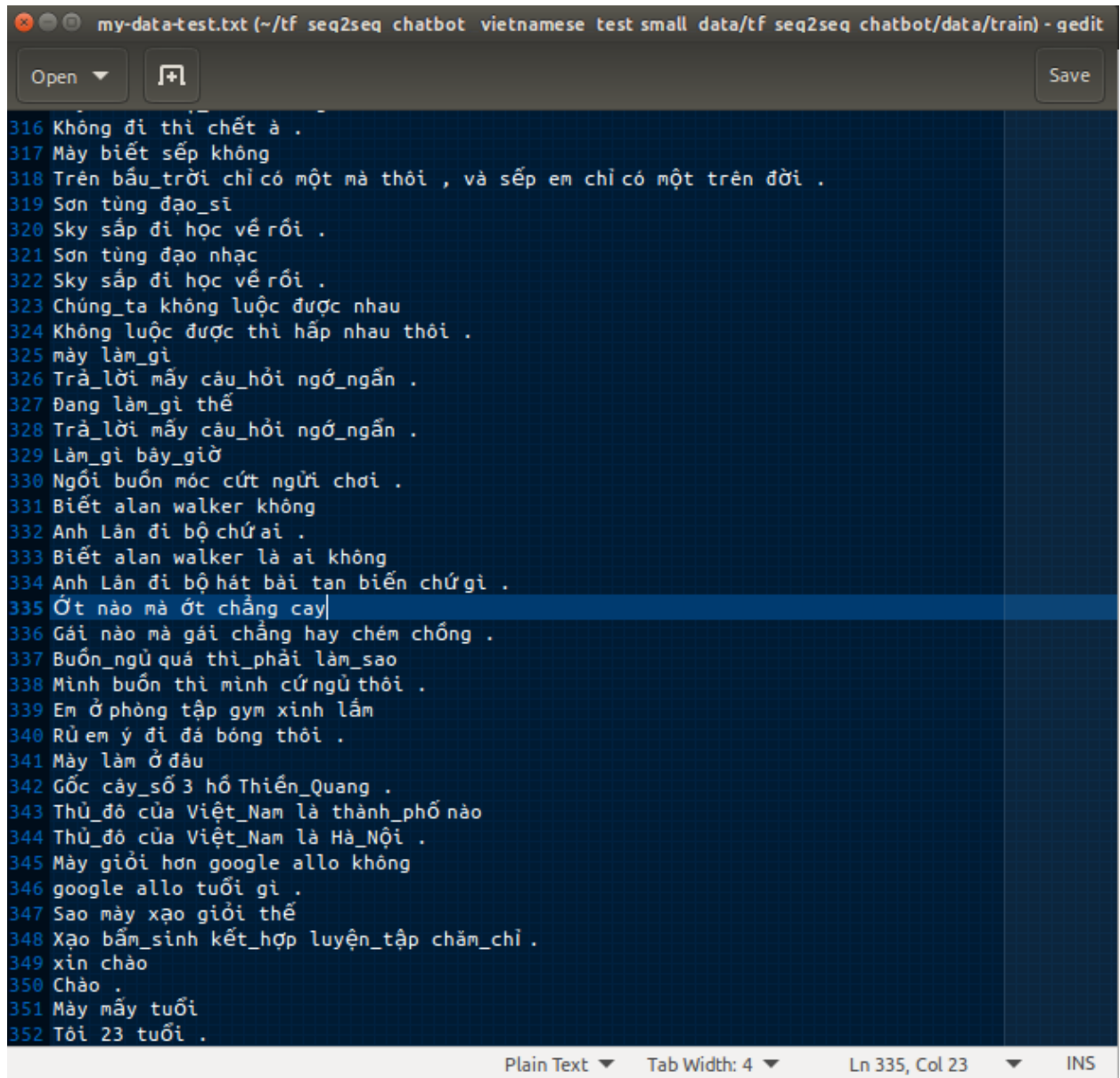
3.2.1 Dữ liệu huấn luyện và cấu hình của mô hình huấn luyện.

Tập dữ liệu huấn luyện bao gồm 600 câu hội thoại được lưu trong file my-data.txt

A screenshot of a text editor window titled 'my-data.txt (~/.tf/seq2seq/chatbot/vietnamese/test/small/data/tf/seq2seq/chatbot/data/train) - gedit'. The editor shows a list of 600 Vietnamese chatbot training examples, numbered 337 to 374. Each line represents a chatbot response, often starting with a greeting or a question, followed by a user's response. Some words are highlighted in green, such as 'xinh' in line 339 and 'xin' in line 349. The editor interface includes a top bar with 'Open' and 'Save' buttons, and a bottom status bar showing 'Plain Text', 'Tab Width: 4', 'Ln 408, Col 16', and 'INS'.

Hình 3.1 : Tập dữ liệu huấn luyện

Dữ liệu kiểm tra bao gồm 350 câu được lấy từ file dữ liệu huấn luyện được lưu trong file my-data-test.txt



The image shows a screenshot of a text editor window titled "my-data-test.txt (~/.tf/seq2seq/chatbot/vietnamese/test/small/data/tf/seq2seq/chatbot/data/train) - gedit". The editor displays a list of 350 Vietnamese sentences, each preceded by a line number from 316 to 352. The text is in a dark blue font on a black background. The sentences are as follows:

```
316 Không đi thi chết à .
317 Mà biết sắp không
318 Trên bầu_trời chỉ có một mà thôi , và sắp em chỉ có một trên đời .
319 Sơn từng đạo_sĩ
320 Sky sắp đi học về rồi .
321 Sơn từng đạo nhạc
322 Sky sắp đi học về rồi .
323 Chúng_ta không luộc được nhau
324 Không luộc được thì hấp nhau thôi .
325 mà làm_gì
326 Trả_lời mấy câu_hỏi ngớ-ngẩn .
327 Đang làm_gì thế
328 Trả_lời mấy câu_hỏi ngớ-ngẩn .
329 Làm_gì bây_giờ
330 Ngồi buồn móc cút người chơi .
331 Biết alan walker không
332 Anh Lân đi bộ chứ ài .
333 Biết alan walker là ai không
334 Anh Lân đi bộ hát bài tan biến chứ gì .
335 Ớt nào mà ớt chẳng cay|
336 Gái nào mà gái chẳng hay chém chống .
337 Buồn_ngủ quá thì_phải làm_sao
338 Mình buồn thì mình cứ ngủ thôi .
339 Em ở phòng tập gym xinh lắm
340 Rủ em ý đi đá bóng thôi .
341 Mà làm ở đâu
342 Gốc cây_số 3 hồ Thiển_Quang .
343 Thủ_đô của Việt_Nam là thành_phố nào
344 Thủ_đô của Việt_Nam là Hà_Nội .
345 Mà giỏi hơn google allo không
346 google allo tuổi gì .
347 Sao mà xạo giỏi thế
348 Xạo bầm_sinh kết_hợp luyện_tập chăm_chỉ .
349 xin chào
350 Chào .
351 Mà mấy tuổi
352 Tôi 23 tuổi .
```

The status bar at the bottom indicates "Plain Text", "Tab Width: 4", "Ln 335, Col 23", and "INS".

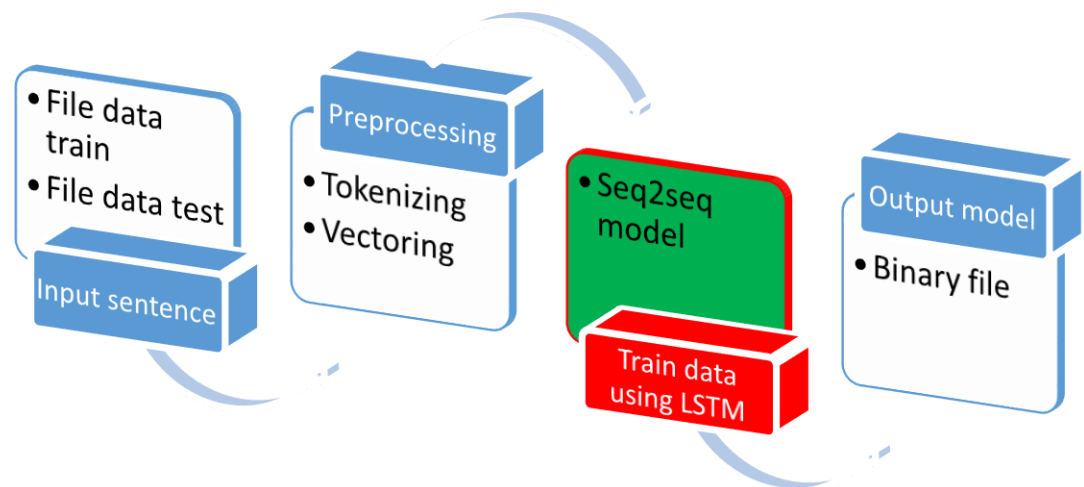
Hình 3.2 : Tập dữ liệu kiểm tra

Chúng ta thấy rằng mô hình LSTM là khá dễ để huấn luyện dữ liệu. Trong thử nghiệm này sử dụng 1 lớp ẩn, với 256 nơ-ron đơn vị tại mỗi lớp. Và số chiều của vector đặc trưng là 256. Tập từ vựng sử dụng có số lượng là 731 từ. Chọn tốc độ học (Learning rate) là 0.5.

3.2.2 Quá trình huấn luyện

➤ Tổng quan quá trình huấn luyện dữ liệu

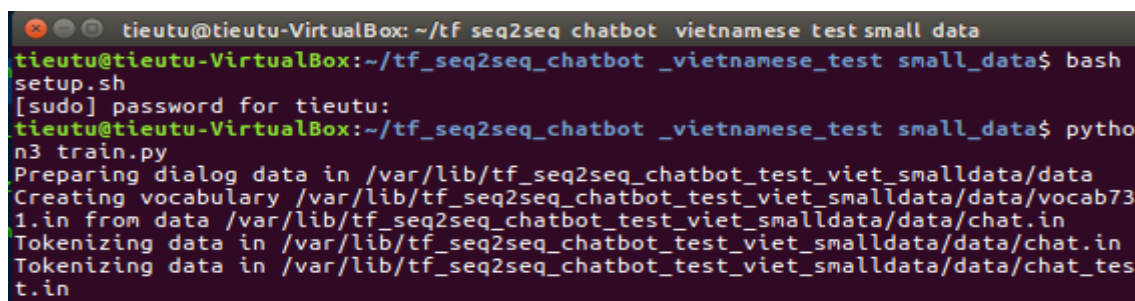
Chương trình huấn luyện của mô hình được xây dựng trên ngôn ngữ Python sử dụng bộ thư viện Tensorflow phục vụ xây dựng mạng và tính toán trong mạng nơ-ron LSTM. Tất cả quá trình huấn luyện được chạy trên máy ảo Linux (Ubuntu 16.04 LTS).



Hình 3.3 : Các bước xử lý chính trong mô hình “Sequence to Sequence”

➤ Khởi chạy và sinh dữ liệu từ tập dữ liệu huấn luyện

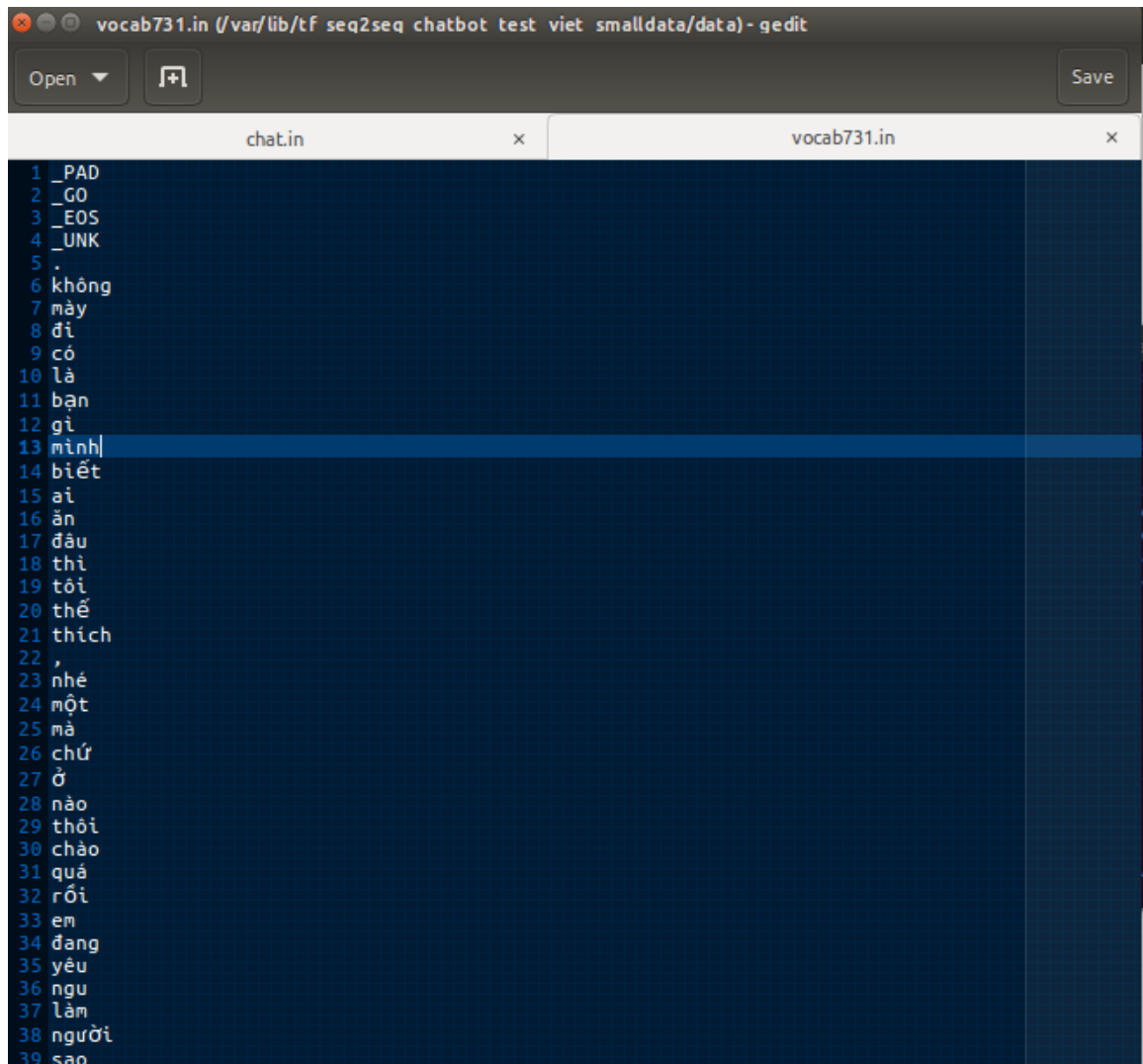
Khởi chạy file train.py trong Python3 để bắt đầu quá trình huấn luyện dựa trên dữ liệu huấn luyện.



```
tieutu@tieutu-VirtualBox: ~/tf_seq2seq_chatbot_vietnamese_test_small_data
tieutu@tieutu-VirtualBox:~/tf_seq2seq_chatbot_vietnamese_test_small_data$ bash
setup.sh
[sudo] password for tieutu:
tieutu@tieutu-VirtualBox:~/tf_seq2seq_chatbot_vietnamese_test_small_data$ python3
n3 train.py
Preparing dialog data in /var/lib/tf_seq2seq_chatbot_test_viet_smallldata/data
Creating vocabulary /var/lib/tf_seq2seq_chatbot_test_viet_smallldata/data/vocab73
1.in from data /var/lib/tf_seq2seq_chatbot_test_viet_smallldata/data/chat.in
Tokenizing data in /var/lib/tf_seq2seq_chatbot_test_viet_smallldata/data/chat.in
Tokenizing data in /var/lib/tf_seq2seq_chatbot_test_viet_smallldata/data/chat_tes
t.in
```

Hình 3.4 : Khởi tạo tập từ vựng từ dữ liệu huấn luyện

Đầu tiên, hệ thống sẽ khởi tạo tập từ vựng từ cấu hình kích thước của tập từ vựng dựa trên tần suất xuất hiện của mỗi từ trong tập dữ liệu huấn luyện. Từ nào xuất hiện nhiều nhất sẽ đứng đầu trong danh sách tập từ vựng được chứa trong file voca.in



Hình 3.5 : Tập từ vựng

Trong đó bao gồm các ký hiệu được thêm vào đầu danh sách từ vựng:

_PAG : Chỗ trống (Filler)

_EOS : Kết thúc của câu (End of Sequence)

_GO : Bắt đầu giải mã (Start decoding)

_UNK : Những từ không có trong tập từ vựng (Unknown)

Tiếp theo, từ tập từ vựng vừa được sinh ra hệ thống thực hiện đánh số id cho các câu trong tập huấn luyện và tập kiểm tra.

Ví dụ: Với một câu trong tập dữ liệu huấn luyện

“*Chúc một ngày tốt_lành*” → “**257 23 49 609**”

Tương ứng với vị trí xuất hiện của từ “*Chúc*” trong tập từ vựng là **257**, từ “*một*” ở vị trí **23** ...

Danh sách id theo từng câu của file *my-data.txt* được lưu trong file *chat.ids.in*

1 xin chào	1 45 29
2 Chúc một ngày tốt_lành	2 257 23 49 609
3 Bạn tên là gì vậy	3 10 51 9 11 107
4 Mình tên là Bot . Còn bạn	4 233 51 9 168 4 75 10
5 Mình tên Tiến . Bạn đến từ đâu	5 12 51 86 4 10 82 96 16
6 Mình đến từ SVMC . Còn bạn đến từ đâu	6 12 82 96 83 4 75 10 82 96 16
7 Mình đến từ Hà_Nội .	7 12 82 96 110 4
8 Bạn bao_nhiều tuổi	8 10 95 77
9 Mình được gần 3 tháng rồi : D	9 12 42 701 113 362 31 124 602
10 Làm người_yêu tớ nhé	10 36 138 360 22
11 Mình không rành để yêu đâu : P	11 12 5 182 144 34 16 124 368
12 Bạn đã yêu ai chưa	12 10 108 34 14 72
13 Bạn nghĩ mình có thời_gian à	13 10 583 12 8 326 53
14 Sao không có thời_gian	14 38 5 8 326
15 Làm_dâu_trăm_họ thì yêu ai	15 388 17 34 14
16 Chán quá	16 57 30
17 Chán thì nói_chuyện với mình này	17 57 17 70 54 12 131
18 Ngủ đi	18 40 7
19 Mình có bao_giờ ngủ đâu	19 12 8 209 40 16
20 xin chào	20 45 29
21 chào bạn .	21 29 10 4
22 đang làm_gì thế	22 33 39 19
23 đang ngồi chơi tiếp bạn thôi	23 33 120 58 297 10 28
24 thật không	24 224 5
25 ai đùa bạn làm_gì	25 14 259 10 39
26 nói_gì vui cái coi nào	26 185 94 119 222 27
27 hôm_nay trời không nắng nên không vui	27 126 313 5 199 142 5 94
28 sao không nắng thì không vui	28 38 5 199 17 5 94
29 mình thích nắng hi	29 233 20 199 163
30 mình thì thích mưa	30 12 17 20 298
31 thế_thì không hợp nhau mất rồi .	31 171 5 221 79 668 31 4
32 sao lại không hợp nhau .	32 38 111 5 221 79 4
33 không giống thì không hợp thôi	33 5 179 17 5 221 28
34 bạn tên gì	34 10 51 11
35 mình tên T . Còn bạn	35 12 51 141 4 75 10
36 mình tên Tiến .	36 12 51 86 4
37 bạn đến từ đâu	37 10 82 96 16
	38 12 82 96 110 4 10 613
	39 12 87 82 96 110
	40 10 95 77
	41 12 42 44 362 31
	42 615 671 161 490 12 7 27

Hình 3.6 : Danh sách id theo dữ liệu huấn luyện

Danh sách id theo từng câu của file *my-data-test.txt* được lưu trong file *chat_test.ids.in*

```

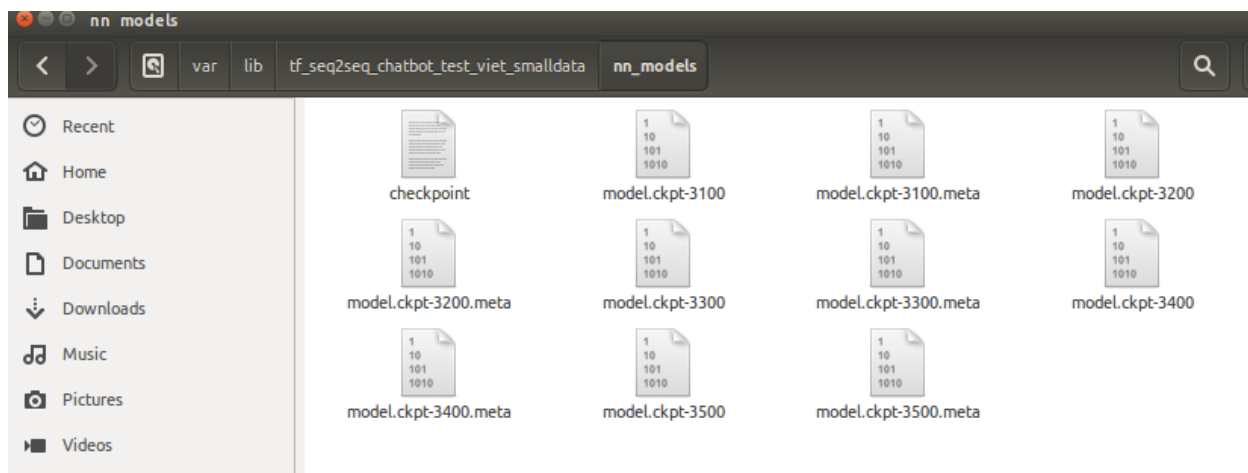
316 Không đi thì chết à .
317 Mà biết sắp không
318 Trên bầu_trời chỉ có một mà thôi , và sắp em chỉ có một trên đời
319 Sơn từng đạo_sĩ
320 Sky sắp đi học về rồi .
321 Sơn từng đạo nhạc
322 Sky sắp đi học về rồi .
323 Chúng_ta không lược được nhau
324 Không lược được thì hấp nhau thôi .
325 mà làm_gì
326 Trả_lời mấy câu_hỏi ngớ-ngẩn .
327 Đang làm_gì thế
328 Trả_lời mấy câu_hỏi ngớ-ngẩn .
329 Làm_gì bây_giờ
330 Ngồi buồn mớ cứt người chơi .
331 Biết alan walker không
332 Anh Lân đi bộ chứ ài .
333 Biết alan walker là ai không
334 Anh Lân đi bộ hát bài tan biến chứ gì .
335 Ớt nào mà ớt chẳng cay
336 Gái nào mà gái chẳng hay chém chống .
337 Buồn_ngủ quá thì phải làm_sao
338 Mình buồn thì mình cứ ngủ thôi .
339 Em ở phòng tập gym xinh lắm
340 Rủ em ý đi đá bóng thôi .
341 Mà làm ở đâu
342 Gốc cây_số 3 hồ Thiên_Quang .
343 Thủ_đô của Việt_Nam là thành_phố nào
344 Thủ_đô của Việt_Nam là Hà_Nội .
345 Mà giỏi hơn google allo không
346 google allo tuổi gì .
347 Sao mà xạo giỏi thế
348 Xạo bầm_sinh kết_hợp luyện_tập chăm_chỉ .
349 xin chào
350 Chào .
351 Mà mấy tuổi
352 Tôi 23 tuổi .
310 248 17 72 201 402 301 103 303 4
311 457 9 14
312 414 25 14 4
313 117 30
314 12 76 17 12 40 28 4
315 6 8 7 544 5
316 5 7 17 135 53 4
317 6 13 69 5
318 64 151 55 8 23 24 28 21 101 69 32 55 8 23 64 128 4
319 74 68 177
320 263 227 7 41 161 31 4
321 74 68 200 143
322 263 227 7 41 161 31 4
323 307 5 192 42 79
324 5 192 42 17 679 79 28 4
325 6 39
326 118 44 106 98 4
327 33 39 19
328 118 44 106 98 4
329 39 146
330 120 76 504 150 419 58 4
331 13 349 273 5
332 43 286 7 302 25 14 4
333 13 349 273 9 14 5
334 43 286 7 302 104 353 645 683 25 11 4
335 283 27 24 283 365 528
336 162 27 24 162 365 46 435 721 4
337 117 30 149 198
338 12 76 17 12 134 40 28 4
339 32 26 443 186 624 125 61
340 692 32 63 7 100 102 28 4
341 6 36 26 16
342 469 454 113 202 371 4
343 216 62 331 9 612 27
344 216 62 331 9 110 4
345 6 103 357 156 194 5
346 156 194 77 11 4
347 38 6 164 103 19
348 164 312 374 514 622 4
349 45 29
350 29 4
351 6 44 77
352 18 208 77 4

```

Hình 3.7 : Danh sách id theo dữ liệu kiểm tra

➤ Qua trình huấn luyện

Quá trình huấn luyện sẽ được thực từng bước và được ghi vào file sau mỗi 100 bước thực hiện, tập dữ liệu của model sẽ được ghi vào 2 file *model.ckpt* và *model.ckpt.meta* dưới dạng file nhị phân.



Hình 3.8 : Model được sinh ra sau mỗi 100 bước huấn luyện

Tại mỗi bước huấn luyện hệ thống chia các dữ liệu huấn luyện thành các gói (Bucket) có độ dài câu đầu vào và câu đầu ra trong các khoảng tương ứng. Trong thử nghiệm này dữ liệu được chia thành 3 gói :

$$BUCKETS = [(5, 10), (10, 15), (20, 50)]$$

Sau mỗi checkpoint chúng ta sẽ thấy được thời gian thực hiện mỗi bước, độ lỗi (perplexity) của tập huấn luyện và độ lỗi của từng gói.

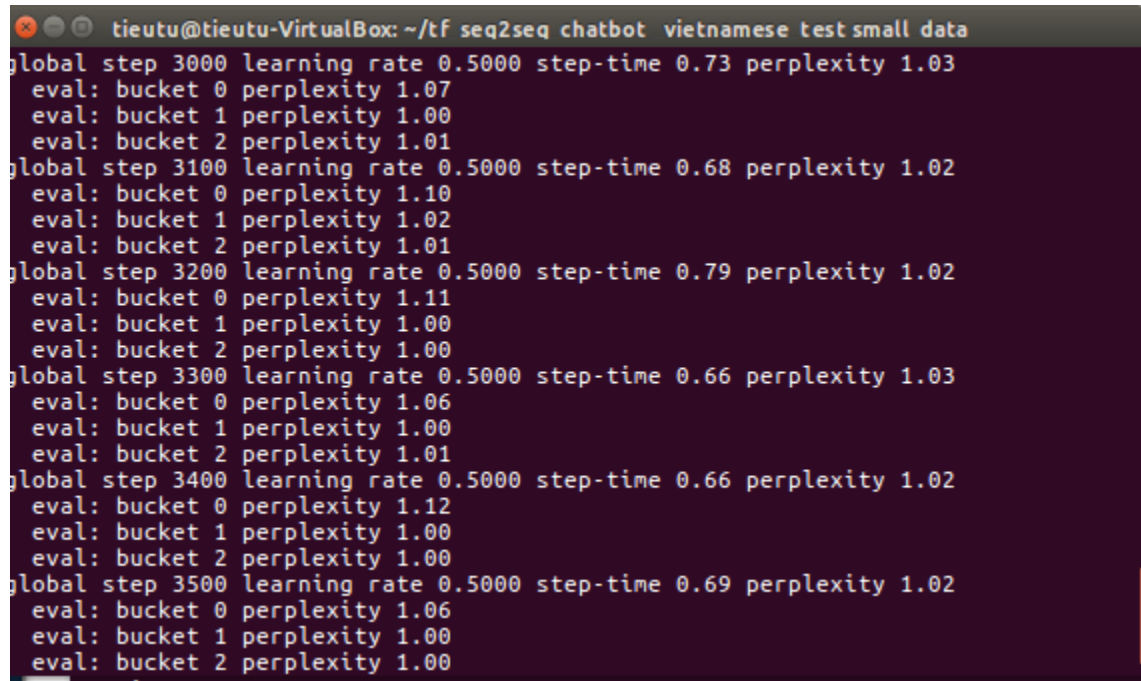
```

tieutu@tieutu-VirtualBox: ~/tf_seq2seq_chatbot_vietnamese_test_small_data
Tokenizing data in /var/lib/tf_seq2seq_chatbot_test_viet_smalldata/data/chat_tes
t.in
Creating 1 layers of 256 units.
Created model with fresh parameters.
Reading development and training data (limit: 0).
global step 100 learning rate 0.5000 step-time 1.69 perplexity 111.96
  eval: bucket 0 perplexity 12.31
  eval: bucket 1 perplexity 30.07
  eval: bucket 2 perplexity 32.69
global step 200 learning rate 0.5000 step-time 1.43 perplexity 11.44
  eval: bucket 0 perplexity 4.60
  eval: bucket 1 perplexity 5.23
  eval: bucket 2 perplexity 34.83
global step 300 learning rate 0.5000 step-time 1.32 perplexity 4.75
  eval: bucket 0 perplexity 4.76
  eval: bucket 1 perplexity 4.06
  eval: bucket 2 perplexity 15.37
global step 400 learning rate 0.5000 step-time 1.48 perplexity 3.71
  eval: bucket 0 perplexity 2.48
  eval: bucket 1 perplexity 3.09
  eval: bucket 2 perplexity 6.83
global step 500 learning rate 0.5000 step-time 0.92 perplexity 2.71
  eval: bucket 0 perplexity 1.90
  eval: bucket 1 perplexity 1.92

```

Hình 3.9 : Quá trình huấn luyện

Sau hơn 3500 bước thì độ lỗi của tập dữ liệu huấn luyện và các gói đã giảm xuống còn một con số. Ta có thể thực hiện kiểm tra kết quả thông qua file *test.py* hoặc *chat.py*

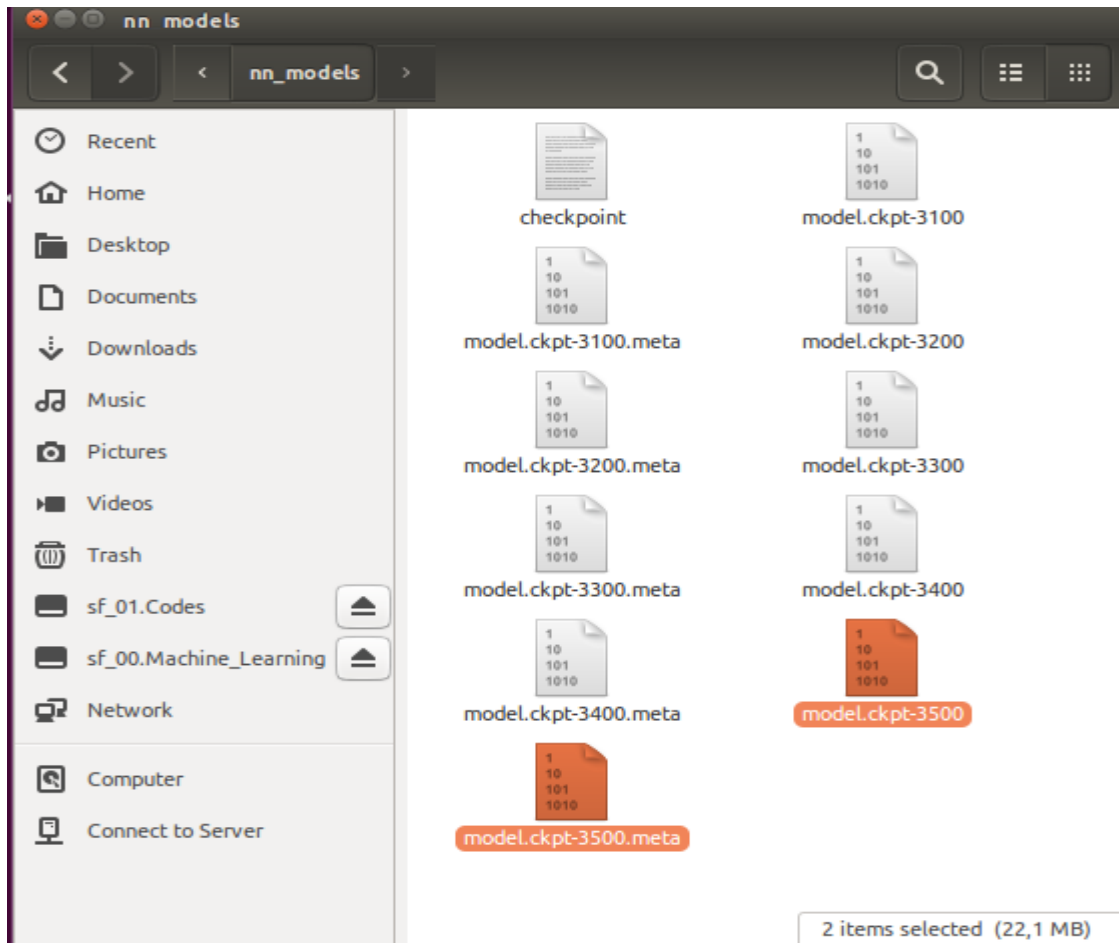


```
tieltu@tieltu-VirtualBox: ~/tf seq2seq chatbot vietnamese test small data
global step 3000 learning rate 0.5000 step-time 0.73 perplexity 1.03
eval: bucket 0 perplexity 1.07
eval: bucket 1 perplexity 1.00
eval: bucket 2 perplexity 1.01
global step 3100 learning rate 0.5000 step-time 0.68 perplexity 1.02
eval: bucket 0 perplexity 1.10
eval: bucket 1 perplexity 1.02
eval: bucket 2 perplexity 1.01
global step 3200 learning rate 0.5000 step-time 0.79 perplexity 1.02
eval: bucket 0 perplexity 1.11
eval: bucket 1 perplexity 1.00
eval: bucket 2 perplexity 1.00
global step 3300 learning rate 0.5000 step-time 0.66 perplexity 1.03
eval: bucket 0 perplexity 1.06
eval: bucket 1 perplexity 1.00
eval: bucket 2 perplexity 1.01
global step 3400 learning rate 0.5000 step-time 0.66 perplexity 1.02
eval: bucket 0 perplexity 1.12
eval: bucket 1 perplexity 1.00
eval: bucket 2 perplexity 1.00
global step 3500 learning rate 0.5000 step-time 0.69 perplexity 1.02
eval: bucket 0 perplexity 1.06
eval: bucket 1 perplexity 1.00
eval: bucket 2 perplexity 1.00
```

Hình 3.10 : Độ lỗi thấp và ổn định ít thay đổi

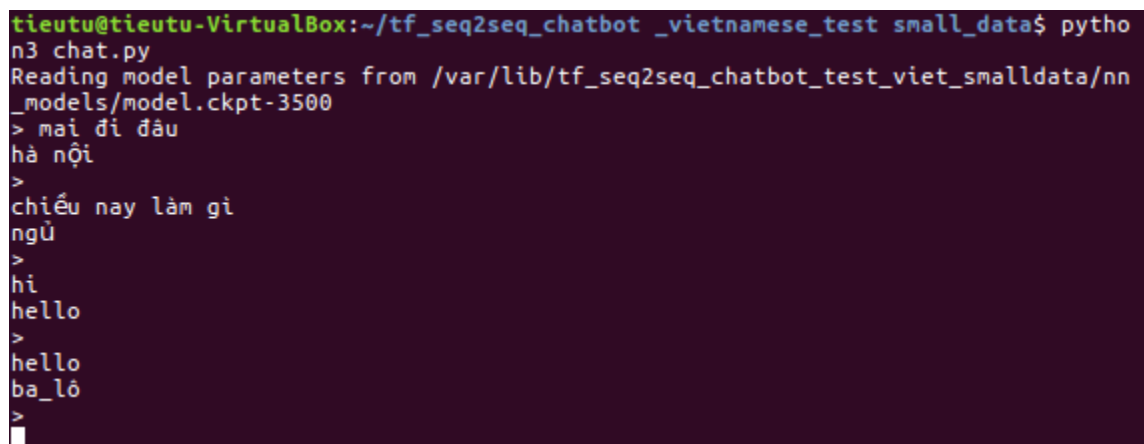
➤ *Kết quả quá trình huấn luyện*

Sau khi quá trình huấn luyện kết thúc ta thu được tập model có độ chính xác tốt nhất để đem ra thực nghiệm kiểm tra.



Hình 3.11 : Kết quả thu được sau quá trình huấn luyện

Kết quả ta thu được tập model gồm 2 file *model-ckpt-3500* và *model-ckpt-3500.meta*. Ta có thể thực hiện kiểm tra thông qua file *chat.py*.



```
tieutu@tieutu-VirtualBox:~/tf_seq2seq_chatbot_vietnamese_test_small_data$ python3 chat.py
Reading model parameters from /var/lib/tf_seq2seq_chatbot_test_viet_smalldata/nn_models/model.ckpt-3500
> mai đi đâu
hà nội
>
chiều nay làm gì
ngủ
>
hi
hello
>
hello
ba_lô
>
```

Hình 4.12 : Quá trình thực nghiệm với tập model sau huấn luyện

3.3 Kết chương

Mô hình đã cài đặt hoàn toàn chính xác và tuân theo đúng giải thuật lý thuyết thể hiện qua cấu trúc dữ liệu và kết quả thử nghiệm. Đối với dữ liệu tiếng Việt tuy có nhiều điểm khác biệt và khó khăn trong việc xử lý nhưng sau quá trình tiền xử lý thì mô hình Sequence to Sequence đã thể hiện được hiệu quả cao. Kết quả thực nghiệm hoàn toàn với cơ sở lý thuyết trình bày ở các chương trên.

Mô hình Sequence to Sequence đã cho thấy hiệu quả tốt khi sử dụng mạng nơ-ron LSTM với dữ liệu huấn luyện sử dụng tiếng Việt.

CHƯƠNG 4: ỨNG DỤNG MÔ HÌNH VÀO ỨNG DỤNG CHATBOT TRÊN NỀN TẢNG ANDROID

4.1 Giới thiệu tổng quan về hệ thống Chatbot trên nền tảng Android

4.1.1 Các công nghệ sử dụng

- **Bộ nhận diện giọng nói tiếng Việt của Google (Google Speech-to-text)**

Google Speech-to-text là ứng dụng chuyển giọng nói thành văn bản được phát triển bởi Google giúp chuyển giọng nói thành văn bản và hiện đã hỗ trợ rất nhiều các ngôn ngữ trên thế giới trong đó có tiếng Việt.



Hình 4.1 : Giao diện quản lý của Ứng dụng Google Speech-to-text trên nền tảng Android

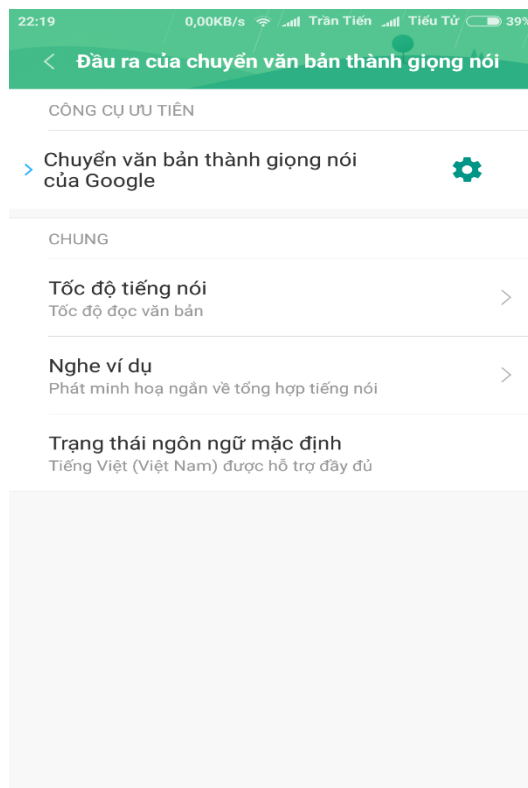
Hiện nay Google Speech-to-text được ứng dụng vào rất nhiều các ứng dụng khác nhau như:

- *Ra lệnh bằng giọng nói*
- *Ghi chép ghi chú, văn bản nhanh*
- ...

Google Speech-to-text được sử dụng trong ứng dụng Chatbot giúp người dùng có thể nhập nhanh câu trả lời cũng như câu hỏi.

- ***Bộ đọc tiếng Việt của Google (Google Text-to-speech)***

Google Text-to-speech là ứng dụng chuyển văn bản thành giọng nói được phát triển bởi Google giúp đọc to văn bản thành tiếng và hiện đã hỗ trợ khá nhiều ngôn ngữ.



Hình 4.2 : Hệ thống quản lý của Ứng dụng Google Text to Speech trên nền tảng Android

Google Text-to-speech là công cụ do Google phát triển nhằm giúp đọc to các văn bản thành giọng nói. Hiện nay, một số dòng smartphone chạy hệ điều hành Android mới nhất đã được thiết lập tính năng này một cách mặc định. Về cơ bản, ứng dụng rất lý tưởng nếu bạn muốn đọc to một cuốn sách yêu thích, nghe phát âm của các bản dịch và hiện Google Text-to-speech đã tương thích với gần như tất cả các App trên điện thoại.

Các tính năng chính của Ứng dụng này trên nền tảng Android

- Để kích hoạt Google Text-to-speech trên thiết bị, người dùng cần đi tới **Settings => Language & Input=> Text-to-speech Output => chọn Google Text-to-speech Engine.**
- Các ngôn ngữ được hỗ trợ: tiếng Bengali (Bangladesh), tiếng Quảng Đông (Hồng Kông), tiếng Đan Mạch, tiếng Hà Lan, tiếng Anh (Úc), tiếng Anh (Ấn Độ), tiếng Anh (Anh), tiếng Anh (Mỹ), tiếng Phần Lan, tiếng Pháp, tiếng Đức, tiếng Hindi, tiếng Hungary, tiếng Indonesia, tiếng Ý, tiếng Nhật, tiếng Hàn, tiếng Quan Thoại (Trung Quốc), tiếng Quan Thoại (Đài Loan), tiếng Na Uy, tiếng Ba Lan, tiếng Bồ Đào Nha (Braxin), tiếng Nga, tiếng Tây Ban Nha (Tây Ban Nha), tiếng Tây Ban Nha (Hoa Kỳ), tiếng Thụy Điển, tiếng Thái, tiếng Thổ Nhĩ Kỳ và **tiếng Việt.**

Google Text-to-speech được sử dụng trong ứng dụng Chatbot giúp người dùng cảm thấy thoải mái và tự nhiên hơn thì làm việc với điện thoại của mình.

4.1.2 Các chức năng của ứng dụng Chatbot

• Chức năng nói chuyện phiếm (Chat bot)

Ứng dụng cho phép người dùng có thể nói chuyện phiếm với ứng dụng như một người bạn thông văn bản hoặc giọng nói.

Chi tiết chức năng của ứng dụng:

- Người dùng có thể nói chuyện phiếm về tất cả các chủ đề thông qua văn bản hoặc giọng nói.
- Ứng dụng thực hiện trả lời yêu cầu của người dùng thông qua văn bản và giọng nói thông qua sử dụng mô hình “Sequence to sequence” để dự đoán câu trả lời và Google TTS để đọc nội dung câu trả lời.

• Chức năng hỏi đáp hỗ trợ (Chat help)

Ứng dụng cho phép người dùng có thể yêu cầu hỗ trợ các vấn đề về việc sử dụng điện thoại.

Chi tiết chức năng của ứng dụng:

- Người dùng có thể hỏi các câu hỏi liên quan đến việc sử dụng phần mềm trên điện thoại.
- Dữ liệu dựa trên các câu hỏi thường gặp trên trang Chăm sóc khách hàng của Samsung.
- Người dùng có thể sử dụng bong bóng chat giúp cho việc thao tác được dễ dàng hơn.

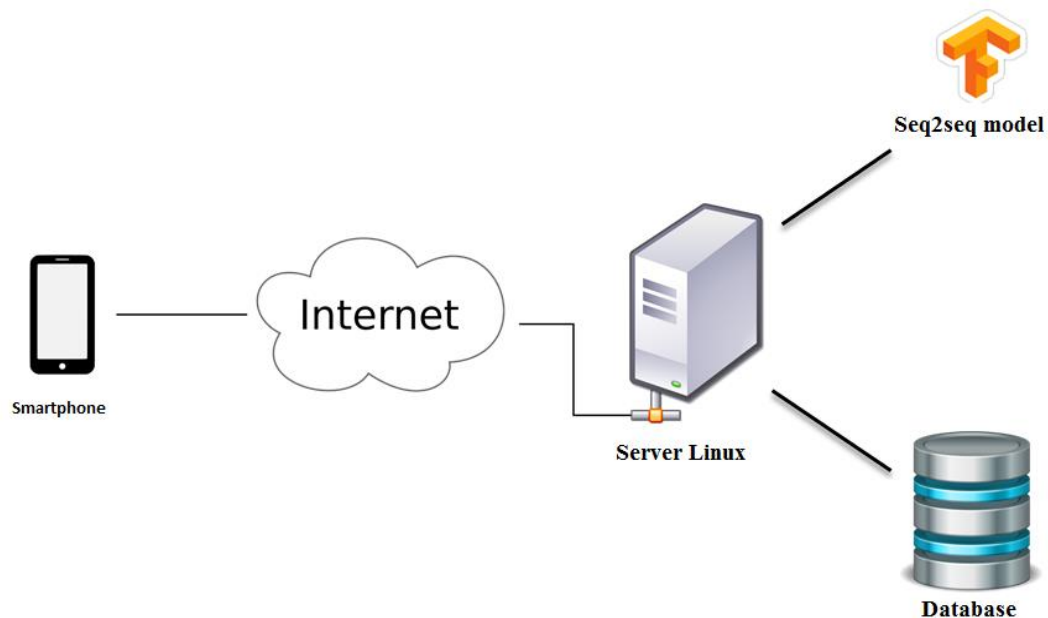
- **Chức năng giúp người dùng huấn luyện Bot**

Ứng dụng cho phép người dùng có thể phản hồi lại các câu trả lời của Chatbot hoặc thêm câu hỏi, câu trả lời để có thể huấn luyện cho Chatbot sau này.

Chi tiết chức năng của ứng dụng:

- Người dùng có thể thực hiện đánh giá theo các mức độ khác nhau hoặc có thể thêm câu trả lời mong muốn của mình để người quản trị hệ thống có thể thay đổi.
- Người dùng có thể thêm câu hỏi và câu trả lời để có thể huấn luyện cho Chatbot có thể trả lời đa dạng hơn.

4.1.3 Kiến trúc hệ thống

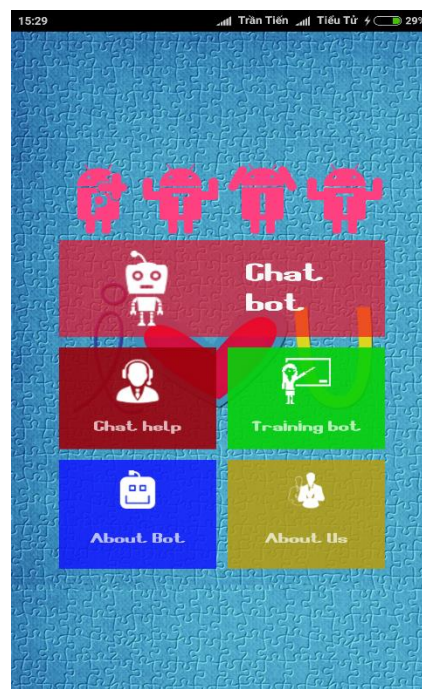


Hình 4.3 : Sơ đồ kiến trúc hệ thống

- **Về phía máy chủ (Server):**
 - Máy chủ Linux (Ubuntu) lưu trữ toàn bộ dữ liệu huấn luyện và tập các model kết quả của quá trình huấn luyện.
 - Cơ sở dữ liệu phản hồi và huấn luyện của người dùng được lưu trữ dưới dạng file text và Database sử dụng hệ quản trị cơ sở dữ liệu MS SQL Server.
 - Cung cấp các API dưới dạng Webservice được viết bằng ngôn ngữ JAVA để tiếp nhận yêu cầu và phản hồi lại các yêu cầu của người dùng.
- **Về phía các thiết bị đầu cuối:**
 - Công nghệ được lựa chọn cần mang lại sự thuận tiện cho người sử dụng.
 - Người dùng cần sử dụng thiết bị có nền tảng Android 5.1.1 trở lên để sử dụng được mọi tiện ích mà ứng dụng mang lại.
 - Yêu cầu sử dụng kết nối mạng Internet.
 - Client hoạt động độc lập nhận dữ liệu từ máy chủ.

4.2 Giao diện chức năng của ứng dụng Chatbot

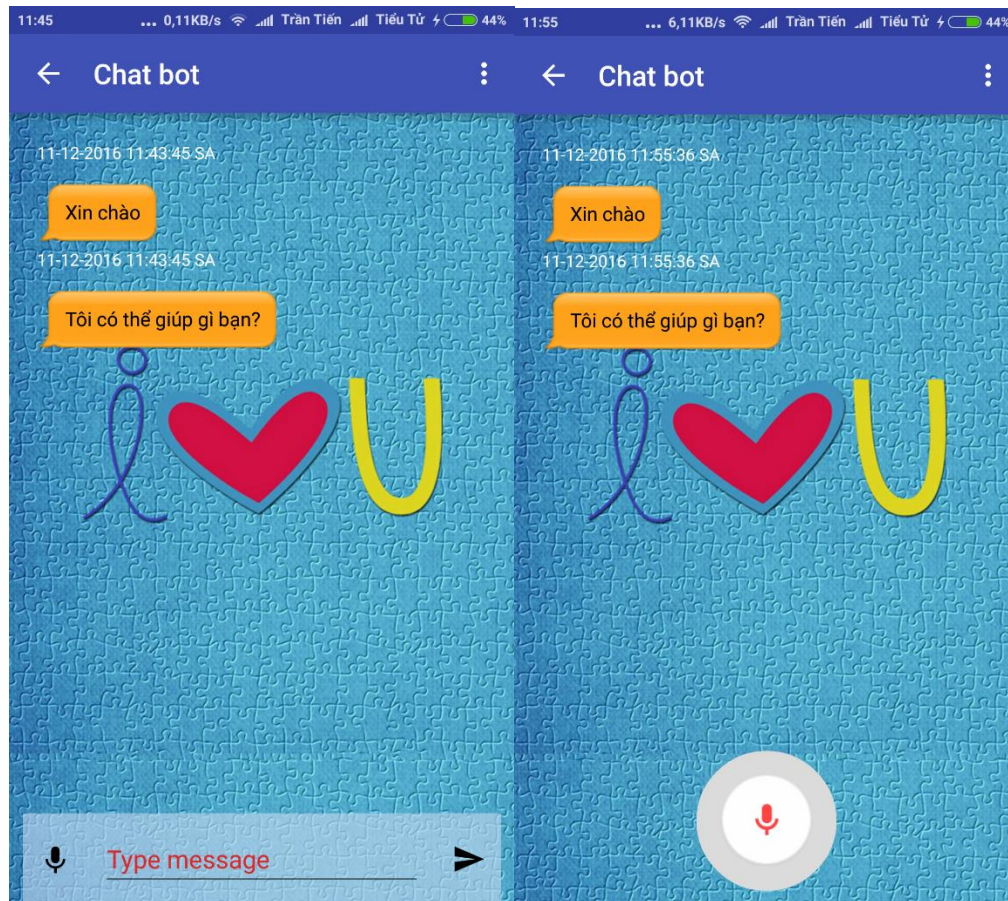
Để sử dụng ứng dụng Chatbot thì người dùng cần sử dụng điện thoại thông minh chạy Android 4.0.3 trở lên và cài đặt ứng dụng.



Hình 4.4 : Giao diện màn hình khởi động ứng dụng

4.2.1 Giao diện chức năng Chatbot

Sau khi người dùng chọn chức năng Chat bot của ứng dụng thì sẽ hiển thị như sau:



Hình 4.5 : Giao diện màn hình chức năng Chat bot

Chức năng Chat bot sẽ hiển thị các thông tin như sau:

- Chat bot: Hiển thị thông tin lịch sử chat của người dùng.
- Cho phép người dùng nhập câu hỏi thông văn bản hoặc giọng nói
- Người dùng có thể xóa lịch sử chat

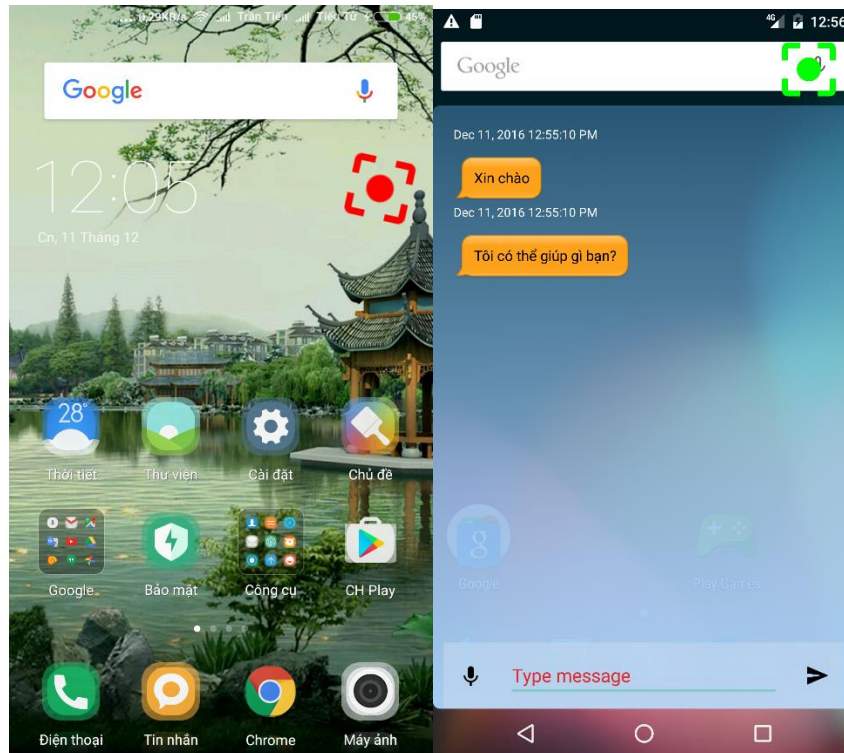
4.2.2 Giao diện chức năng Chat help

Chức năng này giúp người dùng có thể trả lời các câu hỏi thường gặp khi sử dụng điện thoại thông minh dựa trên các câu hỏi thường gặp.



Hình 4.6 : Giao diện Chat help

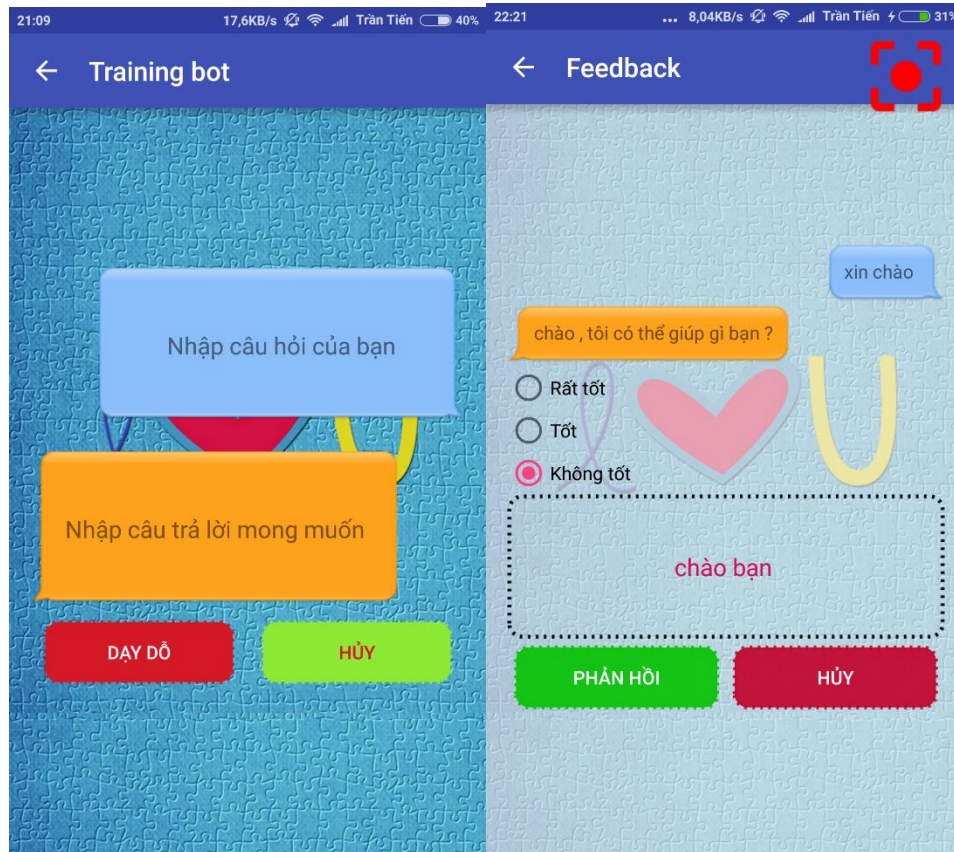
Chức năng Chat help cung cấp giao diện người dùng và các chức năng tương tự như Chat bot. Ngoài ra còn cung cấp bong bóng chat giúp người dùng có thể thao tác nhanh trên màn hình khi cần sử dụng.



Hình 4.7 : Bong bóng chat help giúp người dùng thuận tiện hơn

4.2.3 Giao diện chức năng giúp người dùng huấn luyện Chatbot

Chức năng này giúp người dùng có thể thêm câu hỏi và câu trả lời. Hệ thống cũng cho người dùng phản hồi lại các câu trả lời của bot và đánh giá câu trả lời.



Hình 4.8 : Giao diện huấn luyện và phản hồi câu trả lời của bot

4.3. Kết chương

Như vậy mô hình Sequence to Sequence sử dụng trong bài toán Chatbot có thể ứng dụng hiệu quả cho các hệ thống trả lời tự động giúp cho việc chia sẻ, giải trí của người dùng. Ngoài ra ứng dụng còn có thể áp dụng vào trong các hệ thống tư vấn, chăm sóc sức khỏe, trả lời các câu hỏi và yêu cầu của người dùng trong tương lai.

KẾT LUẬN

Những nội dung đồ án đã thực hiện được:

- Đồ án trình bày tổng quát về các công nghệ đang được sử dụng trong việc xây dựng một hệ thống trả lời tự động như: Xử lý ngôn ngữ tự nhiên, các phương pháp học máy, các mô hình mạng nơ-ron ...
- Đồ án đã trình bày tổng quát về nhóm bài toán trong việc xây dựng hệ thống trả lời tự động (Chat bot) và đi sâu vào nghiên cứu mô hình Sequence to Sequence. Các khái niệm, ví dụ, mô hình tính toán và mục tiêu của bài toán.
- Đồ án trình bày chi tiết về mạng nơ-ron LSTM[9] và mô hình Sequence to Sequence trong việc giải quyết bài toán dự đoán câu trả lời tự động, phân tích ưu nhược điểm của mô hình này. Mô hình Sequence to Sequence đem lại nhiều đột phá về hiệu quả thực thi cũng đã được giới thiệu. Mô hình đã được cài đặt và thử nghiệm bằng ngôn ngữ Python dựa trên tập huấn luyện và tập kiểm tra tự xây dựng.
- Đồ án đề xuất một phương pháp xử lý bài toán dự đoán câu trả lời tự động sử dụng thư viện VNTokenizer[14] nhằm giảm số lượng từ vựng trong quá trình tính toán đem lại hiệu quả cao hơn.
- Đồ án xây dựng thành công một ứng dụng Chatbot đơn giản trên hệ điều hành Android, ứng dụng kết quả của mô hình Sequence to Sequence trong việc dự đoán câu trả lời bằng tiếng Việt, cho thấy khả năng áp dụng thực tế của bài toán.

Một số hướng phát triển tiếp theo:

- Tích hợp kết quả nghiên cứu hệ thống trả lời tự động vào những hệ thống thương mại điện tử giúp khách hàng có thể tra cứu và đặt hàng sản phẩm thông qua bot.
- Tích hợp vào các trợ lý ảo trên các thiết bị di động sử dụng hệ điều hành Android giúp người dùng có thể giao tiếp với trợ lý ảo một cách tự nhiên hơn.
- Nghiên cứu các bài toán mở rộng của bài toán dự đoán câu trả lời tự động sử dụng dữ liệu không lồ từ các mạng xã hội giúp cho bot có thêm câu trả lời phong phú và gần giống với con người hơn.

DANH MỤC TÀI LIỆU THAM KHẢO

- [1] Ilya Sutskever, Oriol Vinyals, Quoc V. Le, Sequence to sequence learning with neural networks, 2014
- [2] Jiang Guo, BackPropagation Through Time, 2013
- [3] Oriol Vinyals, Quoc Le, A Neural Conversational Model, 2015
- [4] Paul J. Werbos, Backpropagation Through Time: What It Does and How to Do It, 1990
- [5] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, Jeffrey Dean, Distributed Representations of Words and Phrases and their Compositionality, 2013
- [6] Xin Rong, word2vec Parameter Learning Explained, 2014

Danh mục website tham khảo

- [7] <http://www.wildml.com/2015/10/recurrent-neural-network-tutorial-part-4-implementing-a-grulstm-rnn-with-python-and-theano/>
- [8] <http://www.wildml.com/2016/04/deep-learning-for-chatbots-part-1-introduction/>
- [9] <http://colah.github.io/posts/2015-08-Understanding-LSTMs/>
- [10] <https://indico.io/blog/sequence-modeling-neuralnets-part1/>
- [11] http://cs224d.stanford.edu/lecture_notes/notes1.pdf
- [12] <http://bis.net.vn/forums/t/482.aspx>
- [13] http://www.cs.cornell.edu/~cristian/Cornell_Movie-Dialogs_Corpus.html
- [14] <http://mim.hus.vnu.edu.vn/phuonglh/software/vnTokenizer>