BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC CẦN THƠ
KHOA CÔNG NGHỆ THÔNG TIN & TRUYỀN THÔNG

LUẬN VĂN TỐT NGHIỆP ĐẠI HỌC
NGÀNH CÔNG NGHỆ THÔNG TIN

Đề tài

XÂY DỰNG HỆ THỐNG TRẢ LỜI TỰ ĐỘNG BẰNG PHƯƠNG PHÁP HỌC TĂNG CƯỜNG VÀ TỰ PHÊ BÌNH

Sinh viên: Nguyễn Văn Vĩ
Mã số: B1507343
Khóa: 41

Cần Thơ, 5/2019

BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC CẦN THƠ
KHOA CÔNG NGHỆ THÔNG TIN & TRUYỀN THÔNG
BỘ MÔN CÔNG NGHỆ THÔNG TIN

LUẬN VĂN TỐT NGHIỆP ĐẠI HỌC
NGÀNH CÔNG NGHỆ THÔNG TIN

Đề tài

XÂY DỰNG HỆ THỐNG TRẢ LỜI TỰ ĐỘNG BẰNG PHƯƠNG PHÁP HỌC TĂNG CƯỜNG VÀ TỰ PHÊ BÌNH

Người hướng dẫn
Lâm Nhựt Khang

Sinh viên thực hiện
Họ và tên: Nguyễn Văn Vĩ
Mã số: B1507343
Khóa: 41

Cần Thơ, 5/2019

MỤC LỤC

DANH MỤC ẢNH

DANH MỤC TỪ VIẾT TẮT

Từ viết tắt	Từ chuẩn
AI	Artificial intelligence
ANN	Artificial Neural Network
DL	Deep Learning
DNN	Deep Neural Networks
LSTM	Long Short Term Memory
ML	Machine Learning
MLE	Maximum Likelihood Estimation
NLP	Natural Language Processing
NLU	Natural Language Understanding
NN	Neural Networks
PG	Policy Gradient
RL	Reinforcement Learning
RNN	Recurrent Neural Network
SCST	Self-Critic for Sequence Training
SDG	Stochastic Gradient Descent
seq2seq	Sequence to Sequence
SMT	Statistical Machine Translation

LỜI CẢM ƠN

Đầu tiên, em xin gửi lời cảm ơn chân thành nhất đến gia đình, những người thân và bạn bè đã thường xuyên quan tâm, động viên, chia sẽ kinh nghiệm, cung cấp các tài liệu hữu ích trong thời gian học tập, nghiên cứu cũng như trong suốt quá trình thực hiện luận văn tốt nghiệp.

Em xin gửi lời cảm ơn sâu sắc tới Thầy Cô trường Đại học Cần Thơ đã tận tình giảng dạy và truyền đạt kiến thức trong suốt khóa học vừa qua. Em cũng xin được gửi lời cảm ơn đến Thầy Cô Khoa Công nghệ Thông tin và Truyền thông đã mang lại cho em những kiến thức vô cùng quý giá và bổ ích trong quá trình học tập tại trường.

Đặc biệt xin chân thành cảm ơn TS. Lâm Nhựt Khang, người đã định hướng, giúp đỡ, trực tiếp hướng dẫn và tận tình chỉ bảo em trong suốt quá trình nghiên cứu, xây dựng và hoàn thiện luận văn này.

Sinh viên thực hiện

Nguyễn Văn Vĩ

NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Cần thơ, ngày tháng năm 2019

Giáo viên hướng dẫn

TS. Lâm Nhựt Khang

TÓM TẮT

Phương pháp SCST (Self-Critic for Sequence Training) là một phương pháp mới vốn được áp dụng cho việc mô tả hình ảnh (image captioning). Phương pháp này chưa được bài báo nào áp dụng vào vấn đề NLP. Luận văn áp dụng thành công phương pháp SCST cho vấn đề NLP với mô hình seq2seq (Sequence to Sequence) sử dụng mạng LSTM để xây dựng hệ thống trả lời tự động (chatbot). Tập dữ liệu với 15.000 cặp câu, mỗi cặp câu gồm một câu hỏi, một câu trả lời được được xử lí và tinh chỉnh từ phụ đề phim Subscene[1]. Kết quả đạt được của luận văn với điểm số BLEU là 0,07.

[1] https://subscene.com

ABSTRACT

SCST (Self-Critic for Sequence Training) method is a new method which is applied to image captioning tasks. This method has not been applied to the NLP issue by any article or paper. The thesis successfully applied SCST method to the NLP problem with the seq2seq model (Sequence to Sequence) using the LSTM network to build a chatbot system. The data set contains 15,000 pairs of sentences, each pair of questions consists of one question, one answer is processed and refined from Subscene movie subtitles. The result of the thesis with BLEU score is 0.07.

CHƯƠNG 1 – GIỚI THIỆU

1. Tính cấp thiết và lí do chọn đề tài

Những năm gần đây, việc giải đáp thắc mắc của bộ phận chăm sóc khách hàng qua tin nhắn trực tuyến đang được ưa chuộng. Tuy nhiên, việc này còn thực hiện một cách thủ công và gặp nhiều khó khăn như: tốn rất nhiều thời gian và chi phí chi trả cho nhân viên chỉ để trả lời những câu hỏi đơn giản và giống nhau. Chính vì vậy, nhu cầu cấp thiết là cần một hệ thống điều khiển thông minh, tự động để mang lại hiệu quả cao hơn và hệ thống trả lời tự động (chatbot) là một sự lựa chọn hoàn hảo.

Hiện nay, các ứng dụng trò chuyện trực tuyến được mọi người sử dụng đang bắt đầu trở thành một phương tiện ưa thích để giao tiếp với các doanh nghiệp và giải quyết thắc mắc của khách hàng. Ứng dụng nhắn tin nhanh đã trở thành điểm đến hàng đầu cho mọi thương hiệu nhằm tiếp cận người tiêu dùng, bởi vậy không có gì đáng ngạc nhiên khi chatbot ngày càng trở nên phổ biến.

2. Mục tiêu đề tài

Với cơ sở thực tiễn, luận văn này đặt ra mục tiêu nghiên cứu các mô hình sinh văn bản sử dụng mạng học sâu (Deep Neural Networks - DNN), dựa trên mô hình seq2seq sử dụng mạng LSTM (mạng bộ nhớ dài ngắn) kết hợp với học tăng cường (Reinforcement Learning - RL) và phương pháp tự phê bình SCST để huấn luyện trên tập dữ liệu miền mở. Từ đó xây dựng, cài đặt và thử nghiệm một mô hình đối thoại sử dụng mạng nơ-ron để huấn luyện trên tập dữ liệu phụ đề phim gồm 30.000 dòng dữ liệu.

3. Đối tượng và phạm vi nghiên cứu

Đối tượng nghiên cứu của luận văn là mô hình seq2seq sử dụng mạng LSTM kết hợp với học tăng cường RL và phương pháp tự phê bình SCST.

Phạm vi nghiên cứu của luận văn là ngôn ngữ tiếng Việt.

4. Cơ sở nghiên cứu khoa học

Mô hình Seq2Seq LSTM [12] là một loại của sinh câu trả lời bằng nơ-ron bằng cách tối đa hóa khả năng tạo ra phản hồi được đưa ra bởi lượt trước đó. Cách tiếp cận này cho phép kết hợp các bối cảnh phong phú khi ánh xạ giữa các lượt đối thoại liên tiếp [13]. Trong bài báo này các tác giả sử dụng tập dữ liệu WMT-14 [2] để hoàn thành tác vụ dịch từ tiếng Anh sang tiếng Pháp. Kết quả của bài báo với điểm số BLEU là 34,8 dựa trên tập test.

Ritter và các cộng sự [14] đặt vấn đề tạo phản hồi như là vấn đề dịch máy thống kê (statistical machine translation - SMT). Trong bài báo này, các tác giả sử dụng tập dữ liệu khoảng 1,3 tỉ cuộc hội thoại được lấy từ twitter với điểm đánh giá BLEU là 1,59. Sordoni và cộng sự [13] đã cải thiện hệ thống Ritter. Bằng cách nối lại các đầu ra của hệ thống hội thoại dựa trên các cụm từ với một mô hình thần kinh kết hợp bối cảnh trước đó. Trong bài báo này, các tác giả sử dụng tập dữ liệu 127 triệu đoạn hội thoại từ Twitter FireHose[3] với điểm số BLEU là 3,21.

Những tiến bộ gần đây trong các mô hình SEQ2SEQ truyền cảm hứng cho các nỗ lực [7] để xây dựng các hệ thống đàm thoại đầu cuối đầu tiên. Nó áp dụng bộ mã hóa để ánh xạ một thông điệp tới một vectơ phân tán đại diện cho ngữ nghĩa của nó và tạo ra phản hồi từ vectơ thông điệp. Các tác giả của bài báo này sử dụng tập dữ liệu OpenSubtitles[4] với 88 triệu câu. Bài báo này sử dụng đánh giá bằng con người với 4 người đánh giá và kết quả đánh giá là 97/200 (97 câu trả lời được cho là hợp lí trên 200 câu trả lời).

Serban và cộng sự [15] đề xuất một mô hình thần kinh phân cấp (generative hierarchical neural network) nắm bắt các phụ thuộc trong lịch sử hội thoại mở rộng. Các tác giả sử dụng tập dữ liệu Movie-Tripples[5] với 245 nghìn đoạn thội thoại (736 nghìn câu) với đánh giá error rate[6] là 66%.

[2] https://nlp.stanford.edu/projects/nmt/
[3] https://developer.twitter.com/en/docs/tweets/compliance/overview
[4] https://www.opensubtitles.org
[5] https://breakend.github.io/DialogDatasets/
[6] https://en.wikipedia.org/wiki/Error_rate

Li và cộng sự [16] đề xuất thông tin lẫn nhau (mutual information) giữa thông điệp và phản hồi như là một chức năng mục tiêu thay thế nhằm giảm tỷ lệ phản hồi chung do các hệ thống seq2seq tạo ra. Các tác giả sử dụng tập dữ liệu OpenSubtitles với khoảng 80 triệu cặp câu (câu hỏi, câu trả lời). Kết quả đánh giá điểm BLEU của bài báo là 1,28.

Luận văn của Đào Văn Chiến[7] sử dụng mô hình Sequence to sequence và mạng LSTM cho tiếng việt đã đạt được một số thành công đáng mong đợi. Bài luận văn sử dụng tập dữ liệu tiếng Việt với 25.000 câu với kết quả đánh giá BLEU là 0,06.

Ranzato đã đề xuất một phương pháp đào tạo dựa trên học tăng cường [10]. Phương pháp này được phát triển dựa trên thuật toán REINFORCE [19], trực tiếp tối ưu hóa số liệu kiểm tra không phân biệt (như BLEU [20], CIDEr [21], METEOR [22], v.v.) và đạt được những cải tiến đầy hứa hẹn. Trong bài báo này Ranzato đã sử dụng tập dữ liệu với 80 nghìn ảnh (với 5 câu mô tả cho mỗi ảnh) để hoàn thành tác vụ mô tả hình ảnh (image captioning). Kết quả đạt được với điểm số BLEU là 29,16.

Phương pháp Actor-critic [23] thường được áp dụng cho việc giảm phương sai cao của gradient. Nó bao gồm đào tạo một mạng giá trị bổ sung để dự đoán phần thưởng dự kiến. Mặt khác, Rennie và cộng sự [24] đã thiết kế một phương pháp tự phê bình (Self-Critic for Sequence Training - SCST), sử dụng đầu ra của thuật toán suy luận thời gian thử nghiệm (test-time inference algorithm [33] [34] [35]) làm cơ sở để chuẩn hóa các phần thưởng, dẫn đến tăng hiệu suất hơn nữa. Trong bài báo này các tác giả sử dụng tập dữ liệu với hơn 113 nghìn ảnh (với 5 mô tả cho mỗi ảnh). Kết quả đánh giá BLEU của bài báo là 31,9.

Trong bài luận văn này, chúng tôi sử dụng phương pháp SCST kết hợp với mô hình SEQ2SEQ và LSTM để tạo ra hệ thống trả lời tự động trên ngôn ngữ Tiếng Việt.

5. Phương pháp nghiên cứu

[7] Xây dựng mô hình trả lời tự động dựa trên mô hình Sequence to sequence với LSTM, Đào Văn Chiến, 2018

Để giải quyết các vấn đề mà bài toán đặt ra, cần phải nghiên cứu lý thuyết và tiến hành thực nghiệm trên phương pháp đã lựa chọn. Các phương pháp nghiên cứu bao gồm:

- Sử dụng Internet để tham khảo, tìm hiểu các ký thuật và tìm nguồn tài liệu thô cho nghiên cứu.
- Tìm hiểu các nghiên cứu liên quan đến đề tài, lên ý tưởng giải quyết vấn đề.
- Tìm hiểu những công cụ, thư viện lập trình phù hợp để áp dụng vào việc nghiên cứu và thử nghiệm các phương pháp.
- Viết báo cáo kết quả.

6. Bố cục luận văn

Luận văn bao gồm các chương sau:

- Chương 1 – Giới thiệu: giới thiệu khái quát về đề tài
- Chương 2 – Cơ sở lí thuyết: trình bày các lí thuyết được áp dụng trong luận văn
- Chương 3 – Cài đặt giải pháp: trình bày cách thức cài đặt giải pháp
- Chương 4 – Thực nghiệm và đánh giá kết quả: trình bày kết quả của đề tài
- Chương 5 – Kết luận: khái quát kết quả đạt được và hướng phát triển

CHƯƠNG 2 – CƠ SỞ LÍ THUẾT

Chương này trình bày tổng quan về chatbot cùng với các kiến thức lí thuyết về các mô hình thuật và thuật toán được áp dụng.

1. Tổng quan về chatbot

1.1 Định nghĩa

Chatbot là một chương trình máy tính hoặc trí thông minh nhân tạo thực hiện một cuộc trò chuyện thông qua các phương pháp thính giác hoặc văn bản. Các chương trình như vậy thường được thiết kế để mô phỏng một cách thuyết phục cách con người cư xử như một đối tác đàm thoại, qua đó vượt qua bài kiểm tra.

Chatbot thường được sử dụng trong các hệ thống hội thoại cho các mục đích thực tế khác nhau bao gồm dịch vụ khách hàng hoặc thu thập thông tin.

1.2 Lịch sử chatbot

Năm 1966, Joseph Weizenbaum xuất bản chương trình ELIZA[8], được coi là một trong những chương trình Chatbots đầu tiên trên thế giới. ELIZA đã đạt được những thành tích đáng kể và được coi là thành tựu đỉnh cao về trí thông minh nhân tạo vào thời điểm đó. Bằng cách nhận ra các từ và cụm từ chính từ đầu vào (Input) của người dùng và đưa những câu trả lời tương ứng bằng cách sử dụng các tập lệnh viết sẵn. Một trong những kịch bản này có tên là DOCTOR, cho phép ELIZA đảm nhận vai trò như một nhà tâm lý học hay một bác sỹ tâm thần. Chatbot ELIZA có thể được cải thiện và nâng cấp từng bước bằng cách chỉnh sửa các kịch bản Chatbot của ELIZA. Khái niệm kịch bản Chatbots cũng được hình thành từ thời điểm này.

Năm 1995, ALICE[9] được xây dựng trên cùng một kỹ thuật được sử dụng để tạo nên ELIZA. ALICE ban đầu được sáng tạo bởi Richard Wallace, ra đời vào ngày 23 tháng 11 năm 1995. Chương trình được viết lại bằng ngôn ngữ Java vào năm 1998. ALICEBOT sử dụng một lược đồ XML có tên AIML (Artificial Intelligence Markup

[8] https://en.wikipedia.org/wiki/ELIZA
[9] https://en.wikipedia.org/wiki/Artificial_Linguistic_Internet_Computer_Entity

Language- Ngôn ngữ đánh dấu trí thông minh nhân tạo) để xác định các quy tắc trò chuyện heuristic. Tuy nhiên, nó lại không thể vượt qua Các phép thử Turing.

Những năm 2010-2016, nửa đầu thập kỷ này chúng ta chứng kiến sự bùng nổ của các trợ lý cá nhân ảo: Siri[10] (2010), Google Now[11] (2012), Alexa[12] (2015), Cortana[13] (2015) và Google Assistant[14] (2016). Với khả năng phân tích và xử lý ngôn ngữ tự nhiên, các trợ lý này kết nối với các dịch vụ web để trả lời các câu hỏi và đáp ứng các yêu cầu của người dùng.

Gần đây, Google Home và Amazon Echo đã bắt đầu trở thành các tính năng phổ biến trong các ngôi nhà của người Mỹ. Google Assistant và Alexa là cơ sở để cho các thiết bị thông minh trong ngôi nhà tương tác với người dùng. Các thiết bị thông minh Smart Devices kết nối với nhau tạo thành hệ thống Smart Home, cho phép người dùng ra lệnh và điều khiển bằng giọng nói của họ.

Từ năm 2016 đến nay, sau hội nghị F8 năm 2016, Facebook – mạng xã hội lớn nhất thế giới giới thiệu Messenger Platform. Một nền tảng thân thiện hơn và cho phép bất kỳ ai cũng có thể tạo cho mình một Chatbot. Ngay sau đó, các ứng dụng chat khác như LINE, WhatsApp, Telegram hay Twitter cũng đưa ra các hỗ trợ hoặc các API cho phép người dùng tạo các Chatbot trên ứng dụng nhắn tin.

Nhưng WeChat của Trung Quốc mới chính là người đi tiên phong trong lĩnh vực này khi cho ra mắt Xiaoice – chatbot khá hoàn thiện từ năm 2013. Trong cuộc đua của các nhà phát triển chatbots, Facebook đang nắm giữ thị phần toàn cầu lớn nhất vì có đến hơn 1 tỷ người sử dụng ứng dụng Messenger hàng tháng. Riêng ở thị trường Trung Quốc, WeChat lại là ứng dụng chat số 1 mà không ứng dụng chat nào có thể cạnh tranh nổi.

1.3. Phân loại

1.3.1 Phân loại theo hướng tiếp cận

[10] https://en.wikipedia.org/wiki/Siri
[11] https://en.wikipedia.org/wiki/Google_Now
[12] https://en.wikipedia.org/wiki/Amazon_Alexa
[13] https://en.wikipedia.org/wiki/Cortana
[14] https://en.wikipedia.org/wiki/Google_Assistant

Tiếp cận dựa trên truy xuất thông tin (Retrieval-based models): mô hình sử dụng một kho lưu trữ các câu trả lời được xác định trước và một số thuật toán Heuristic để chọn một phản ứng phù hợp dựa trên đầu vào và ngữ cảnh. Các hệ thống này không tạo ra bất kỳ văn bản mới nào, chúng chỉ chọn một câu trả lời từ một tập hợp cố định sẵn có.

Tiếp cận dựa vào mô hình sinh sản (Generative models): mô hình không dựa vào câu trả lời được xác định trước, chúng tạo ra các phản hồi từ đầu. Các mô hình sinh văn bản thường dựa trên các kỹ thuật dịch máy nhưng thay vì dịch từ ngôn ngữ này sang ngôn ngữ khác, chúng dịch từ đầu vào thành đầu ra.

1.3.2 Phân theo độ dài đoạn hội thoại

Đoạn hội thoại càng dài thì càng khó để tự động hóa nó. Với các đoạn hội thoại ngắn (Short-Text Conversations) mục tiêu là tạo ra một phản hồi đơn cho một mẫu đơn đầu vào. Đối với những đoạn hội thoại dài (Long-Text Conversations) cần đi qua nhiều điểm ngắt câu và phải theo dõi những gì đã được nói trước đó. Các đoạn hội thoại hỗ trợ khách hàng là một ví dụ cho dạng này.

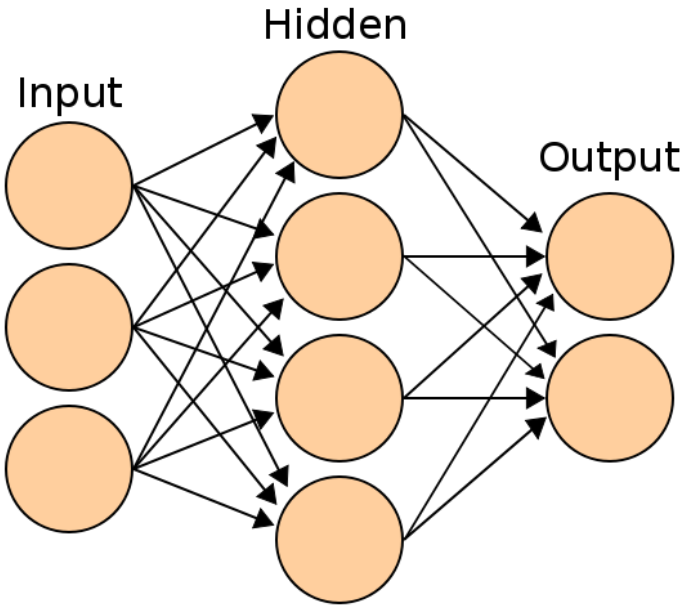
1.3.3 Phân theo miền

Miền mở (Open Domain): xây dựng chatbot trên miền mở cho phép người dùng có thể thực hiện cuộc trò chuyện ở bất kì đâu, không nhất thiết phải có mục tiêu hay ý định rõ ràng. Các cuộc hội thoại trên các mạng xã hội như Twitter và Reddit thường là miền mở, chúng có thể đi vào tất cả các đoạn thoại. Số lượng chủ đề là vô hạn do đó việc tạo ra các phản hồi hợp lý trở nên khó khăn hơn. Tuy nhiên, việc thu thập dữ liệu loại này khá đơn giản. Xây dựng hệ thống trả lời tự động dựa trên mô hình Sequence to Sequence với LSTM.

Miền đóng (Close Domain): ở miền này không gian của các đầu vào và đầu ra bị giới hạn ở một lĩnh vực cụ thể nào đó. Ví dụ: hỗ trợ khách hàng kĩ thuật, hỗ trợ mua sắm, hướng dẫn y tế, … Ở những hệ thống này chúng chỉ cố gắng hoàn thành tốt nhất nhiệm vụ của mình và mặc dù người dùng có thể hỏi bất kì điều gì nhưng hệ thống không yêu cầu xử lý các trường hợp ngoại lệ đó.

2. Mạng nơ-ron nhân tạo ANN

Artificial Neural Network (ANN[15]) gồm 3 thành phần chính: Input layer và output layer chỉ gồm 1 layer, hidden layer có thể có 1 hay nhiều layer tùy vào bài toán cụ thể. ANN hoạt động theo hướng mô tả lại cách hoạt động của hệ thần kinh với các neuron được kết nối với nhau. Trong ANN, trừ input layer thì tất cả các node thuộc các layer khác đều full-connected với các node thuộc layer trước nó. Mỗi node thuộc hidden layer nhận vào ma trận đầu vào từ layer trước và kết hợp với trọng số để ra được kết quả [25].

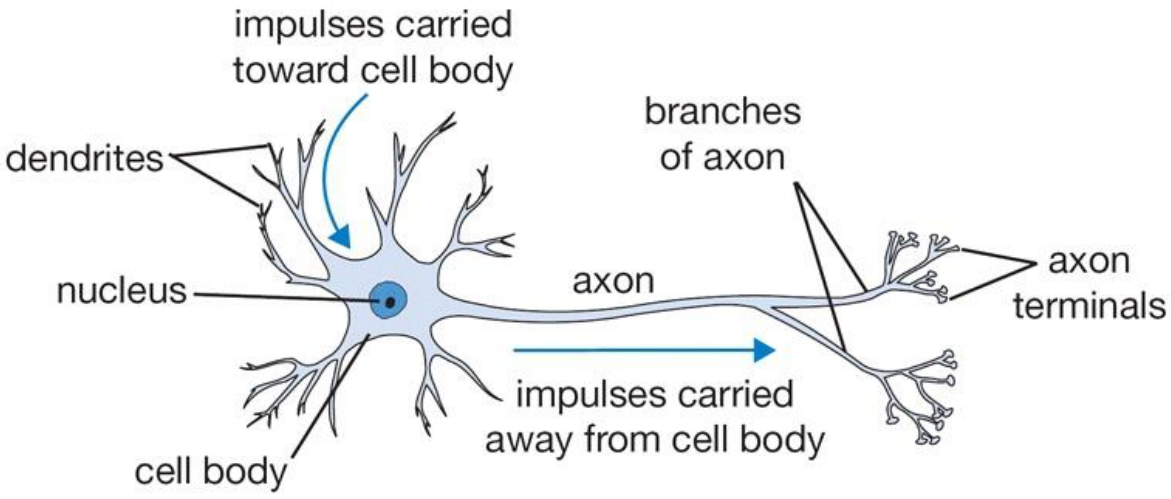


Hình 1 Mạng ANN cơ bản

2.1 Perceptron

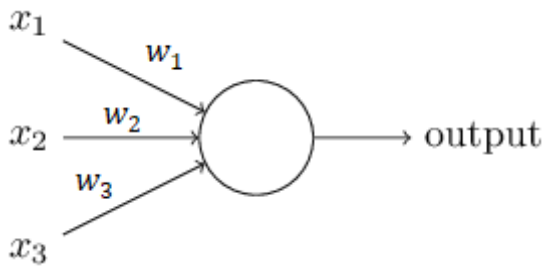
Một mạng nơ-ron được cấu thành bởi các nơ-ron đơn lẻ được gọi là các perceptron[16]. Nên trước tiên cần tìm hiểu xem perceptron là gì đã rồi tiến tới mô hình của mạng nơ-ron sau. Nơ-ron nhân tạo được lấy cảm hứng từ nơ-ron sinh học như mô tả bên dưới.

[15] https://en.wikipedia.org/wiki/Artificial_neural_network
[16] https://en.wikipedia.org/wiki/Perceptron



Hình 2 Nơ-ron thần kinh (Nguồn: https://cs231n.github.io/)

Như hình trên có thể thấy rằng một nơ-ron có thể nhận nhiều đầu vào và cho ra một kết quả duy nhất. Mô hình của perceptron cũng tương tự như vậy:



Hình 3 Mô hình perceptron

Một perceptron sẽ nhận một hoặc nhiều đầu **x** vào dạng nhị phân và cho ra một kết quả **o** dạng nhị phân duy nhất. Các đầu vào được điều phối tầm ảnh hưởng bởi các tham số trọng lượng tương ứng **w** của nó, [25] còn kết quả đầu ra được quyết định dựa vào một ngưỡng quyết định **b** nào đó:

$$o = \begin{cases} 0 \; if \; \sum w_i x_i \leq threshold \\ 1 \; if \; \sum w_i x_i > threshold \end{cases}$$

Đặt $b = -$threshold, ta có thể viết lại thành:

$$o = \begin{cases} 0 \; if \; \sum w_i x_i + b \leq 0 \\ 1 \; if \; \sum w_i x_i + b > 0 \end{cases}$$

Nếu gắn x_0=1 và w_0=b, ta còn có thể viết gọn lại thành:

$$o = \begin{cases} 0 \; if \; \sum \boldsymbol{w^T x} \leq 0 \\ 1 \; if \; \sum \boldsymbol{w^T x} > 0 \end{cases}$$

2.2 Sigmoid Function

Với đầu vào và đầu ra dạng nhị phân, chúng ta rất khó có thể điều chỉnh một lượng nhỏ đầu vào để đầu ra thay đổi chút ít, nên để linh động, chúng ta có thể mở rộng chúng ra cả khoảng [0,1]. Lúc này đầu ra được quyết định bởi một hàm $\sigma(\boldsymbol{w^T x})$. Hàm sigmoid[17] có công thức:

$$\sigma(z) = \frac{1}{1 + \; e^{-z}}$$

Một số hàm khác như **tanh, ReLU** để thay thế hàm sigmoid bởi dạng đồ thị của nó cũng tương tự như sigmoid. Một cách tổng quát, hàm perceptron được biểu diễn qua một hàm kích hoạt (activation function) **f(z)** như sau:

$$o = f(z) = f(\boldsymbol{w^T x})$$

2.3 Kiến trúc mạng NN

Mạng NN là sự kết hợp của của các tầng perceptron hay còn được gọi là perceptron đa tầng (multilayer perceptron) (hình 2).

Trong mạng NN, mỗi nút mạng là một sigmoid nơ-ron nhưng hàm kích hoạt của chúng có thể khác nhau. Tuy nhiên trong thực tế người ta thường để chúng cùng dạng với nhau để tính toán cho thuận lợi.

Ở mỗi tầng, số lượng các nút mạng (nơ-ron) có thể khác nhau tuỳ thuộc vào bài toán và cách giải quyết. Nhưng thường khi làm việc người ta để các tầng ẩn có số lượng nơ-ron bằng nhau. Ngoài ra, các nơ-ron ở các tầng thường được liên kết đôi một với nhau tạo thành mạng kết nối đầy đủ (full-connected network). Khi đó ta có thể tính được kích cỡ của mạng dựa vào số tầng và số nơ-ron.

2.4 Lan truyền tiến

[17] https://en.wikipedia.org/wiki/Sigmoid_function

Như đã thấy thấy thì tất cả các nốt mạng (nơ-ron) được kết hợp đôi một với nhau theo một chiều duy nhất từ tầng vào tới tầng ra. Tức là mỗi nốt ở một tầng nào đó sẽ nhận đầu vào là tất cả các nốt ở tầng trước đó mà không suy luận ngược lại. Hay nói cách khác, việc suy luận trong mạng NN là suy luận tiến [26].

$$z_i^{l+1} = \sum_{j=1}^{n(l)} w_{ij}^{(l+1)} a_j^{(l)} + b_i^{(l+1)}$$

$$a_i^{(l+1)} = f(z_i^{(l+1)})$$

Trong đó, $n(l)$ là số lượng nút ở tầng l tương ứng và $a_j^{(l)}$ là nút mạng thứ j của tầng l. Còn $w_{ij}^{(l+1)}$ là tham số trọng lượng của đầu vào $a_j^{(l)}$ đối với nút mạng thứ I của tầng $l+1$ và $b_i^{(l+1)}$ là độ lệch (bias) của nút mạng thứ I của tầng $l+1$. Đầu ra của nút mạng này được biểu diễn bằng $a_i^{(l+1)}$ ứng với hàm kích hoạt$f(z_i)$.

Riêng với tầng vào, thông thường a^1 cũng chính là đầu vào x tương ứng của mạng. Để tiện tính toán ta coi $a_0^{(l)}$ là một đầu vào và $w_{i0}^{(l+1)} = b_i^{(l+1)}$ là tham số trọng lượng của đầu vào này. Lúc đó có thể viết lại công thức trên dưới dạng vec-tơ:

$$z_i^{(l+1)} = \boldsymbol{w}_i^{(l+1)} . \boldsymbol{a}^{(l)}$$

$$a_i^{(l+1)} = f(z_i^{(l+1)})$$

Nếu nhóm các tham số của mỗi tầng thành một ma trận có các cột tương ứng với tham số mỗi nút mạng thì ta có thể tính toán cho toàn bộ các nút trong một tầng bằng véc-tơ:

$$\boldsymbol{z}^{(l+1)} = \boldsymbol{W}^{(l+1)} . \boldsymbol{a}^{(l)}$$

$$\boldsymbol{a}^{(l+1)} = f(\boldsymbol{z}^{(l+1)})$$

2.5 Học với mạng NN

Cũng tương tự như các bài toán học máy khác thì quá trình học vẫn là tìm lấy một hàm lỗi để đánh giá và tìm cách tối ưu hàm lỗi đó để được kết quả hợp lý nhất có thể:

$$J(W) = -\frac{1}{m}\sum_{i=1}^{m}\sum_{k=1}^{K}\left(y_k^{(i)} \log\left(\sigma_k^{(i)}\right) + \left(1 - y_k^{(i)}\right)\log\left(1 - \sigma_k^{(i)}\right)\right)$$

Trong đó, m là số lượng dữ liệu huấn luyện huấn luyện, K là số nút ra, $y_k^{(i)}$ là đầu ra thực tế của nút thứ k dữ liệu thứ i. σ_k là đầu ra ước lượng được cho nút thứ k tương ứng.

2.6 Lan truyền ngược và đạo hàm:

Để tính đạo hàm của hàm lỗi $\nabla J(W)$ trong mạng NN, chúng ta sử dụng một giải thuật đặc biệt là giải thuật lan truyền ngược (backpropagation[18]) [26] [27]. Nhờ có giải thuật được sáng tạo vào năm 1986 này mà mạng NN thực thi hiệu quả được và ứng dụng ngày một nhiều cho tới tận ngày này.

- Bước 1 - Lan truyền tiến

  Lần lượt tính các $\boldsymbol{a}^{(l)}$ từ $l = 2 \rightarrow L$ theo công thức:

  $$\boldsymbol{z}^{(l)} = \boldsymbol{W}^{(l)} . \boldsymbol{a}^{(l-1)}$$
  $$\boldsymbol{a}^{(l)} = f(\boldsymbol{z}^{(l)})$$

  Trong đó, tầng vào $\boldsymbol{a}^{(l)}$ chính bằng giá trị vào của mạng **x**

- Bước 2 - Tính đạo hàm theo z ở tầng ra

  $$\frac{\partial J}{\partial \boldsymbol{z}^{(L)}} = \frac{\partial J}{\partial \boldsymbol{a}^{(L)}} \frac{\partial \boldsymbol{a}^{(L)}}{\partial \boldsymbol{z}^{(L)}}$$

  Với $\boldsymbol{a}^{(L)}$, $\boldsymbol{z}^{(L)}$ được tính ở bước 1

- Bước 3 - Lan truyền ngược

  Tính đạo hàm theo z ngược lại từ $l = (L-1) \rightarrow 2$ theo công thức:

  $$\frac{\partial J}{\partial \boldsymbol{z}^{(l)}} = \frac{\partial J}{\partial \boldsymbol{z}^{(l+1)}} \frac{\partial \boldsymbol{z}^{(l+1)}}{\partial \boldsymbol{a}^{(l)}} \frac{\partial \boldsymbol{a}^{(l)}}{\partial \boldsymbol{z}^{(l)}} = \left(\left(\boldsymbol{W}^{(l+1)}\right)^T \frac{\partial J}{\partial \boldsymbol{z}^{(l+1)}}\right) \frac{\partial \boldsymbol{a}^{(l)}}{\partial \boldsymbol{z}^{(l)}}$$

[18] https://en.wikipedia.org/wiki/Backpropagation

Với $\boldsymbol{z}^{(l)}$ được tính ở bước 1 và $\frac{\partial J}{\partial \boldsymbol{z}^{(l+1)}}$ được tính ở vòng lặp trước đó

- Bước 4 - Tính đạo hàm

  Tính đạo hàm theo tham số w bằng công thức:

$$\frac{\partial J}{\partial \boldsymbol{W}^{(l)}} = \frac{\partial J}{\partial \boldsymbol{z}^{(l)}} \frac{\partial \boldsymbol{z}^{(l)}}{\partial \boldsymbol{W}^{(l)}} = \frac{\partial J}{\partial \boldsymbol{z}^{(l)}} \left(\boldsymbol{a}^{(l-1)}\right)^T$$

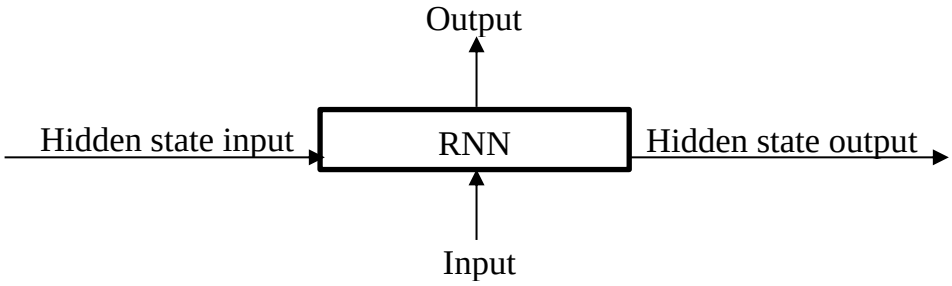
  Với $\boldsymbol{a}^{(l-1)}$ được tính ở bước 1 và $\frac{\partial J}{\partial \boldsymbol{z}^{(l)}}$ được tính ở bước 3

3. Mạng nơ-ron hồi quy RNN

Xử lí ngôn ngữ tự nhiên (Natural Language Processing – NLP) có những đặc thù riêng khiến nó khác trong khoa học máy tính hoặc các lĩnh vực khác. Một trong số đó là xử lý các đối tượng có độ dài thay đổi. Ở các cấp độ khác nhau, NLP đang xử lý các đối tượng có thể có độ dài khác nhau, ví dụ, một từ trong ngôn ngữ có thể chứa nhiều ký tự. Các câu được hình thành từ các chuỗi từ có độ dài thay đổi. Đoạn văn hoặc tài liệu bao gồm số lượng câu khác nhau. Sự biến đổi như vậy không chỉ là đặc thù của NLP mà còn có thể phát sinh trong các lĩnh vực khác nhau, như trong xử lý tín hiệu hoặc xử lý video. Ngay cả các vấn đề về thị giác máy tính tiêu chuẩn cũng có thể được coi là một chuỗi các đối tượng, như vấn đề chú thích hình ảnh khi Mạng nơ-ron (NN) có thể tập trung vào nhiều vùng khác nhau của cùng một hình ảnh để mô tả hình ảnh tốt hơn. Do đó RNN[19] xuất hiện. Ý tưởng về RNN là một mạng với đầu vào và đầu ra cố định, được áp dụng cho chuỗi các đối tượng và có thể truyền thông tin dọc theo chuỗi này. Thông tin này được gọi là trạng thái ẩn và thường chỉ là một vectơ số. Trên sơ đồ sau, chúng ta có RNN với một đầu vào là vectơ số, đầu ra là vectơ khác. Điều làm cho nó khác với một mạng chuyển tiếp hoặc tích chập tiêu chuẩn là hai cổng phụ: một đầu vào và một đầu ra. Đầu vào bổ sung cung cấp trạng

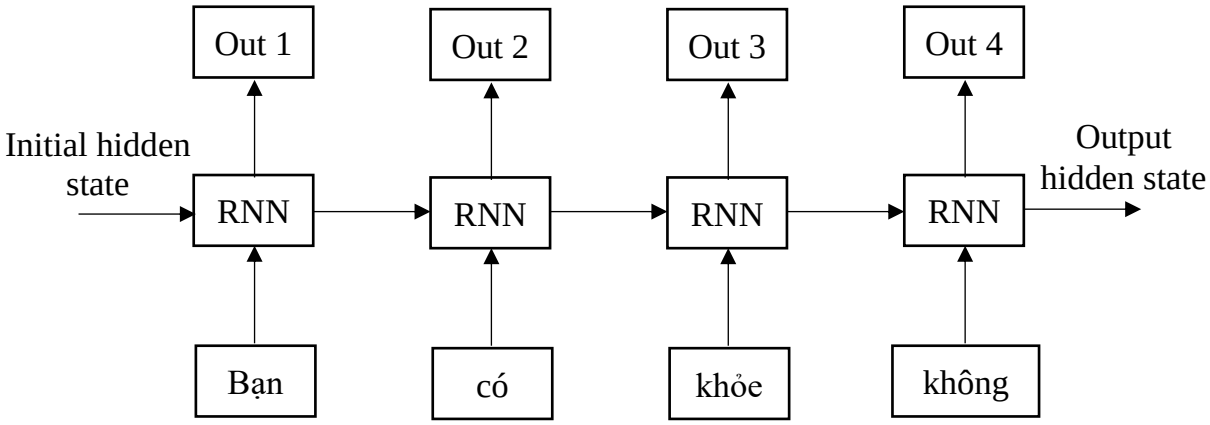
[19] https://en.wikipedia.org/wiki/Recurrent_neural_network

thái ẩn từ mục trước vào đơn vị RNN và đầu ra bổ sung cung cấp trạng thái ẩn được chuyển đổi sang chuỗi tiếp theo.



Hình 4 RNN cơ bản

RNN đã giải quyết vấn đề độ dài thay đổi của đầu vào. Vì một RNN có hai đầu vào, nó có thể được áp dụng cho các chuỗi đầu vào có độ dài bất kỳ, chỉ bằng cách chuyển trạng thái ẩn được tạo bởi mục trước đó sang mục tiếp theo. Trong hình bên dưới, một RNN được áp dụng cho câu "this is a cat", tạo ra đầu ra cho mỗi từ trong chuỗi. Trong ứng dụng, chúng tôi có cùng một RNN được áp dụng cho mọi mục đầu vào, nhưng bằng cách có trạng thái ẩn, giờ đây nó có thể truyền thông tin dọc theo chuỗi. Điều này tương tự với các mạng nơ ron tích chập, khi chúng ta có cùng một bộ lọc được áp dụng cho các vị trí khác nhau của hình ảnh, nhưng điểm khác biệt là mạng chập không có thể truyền đi trạng thái ẩn.



Hình 5 Cách thông tin truyền qua RNN

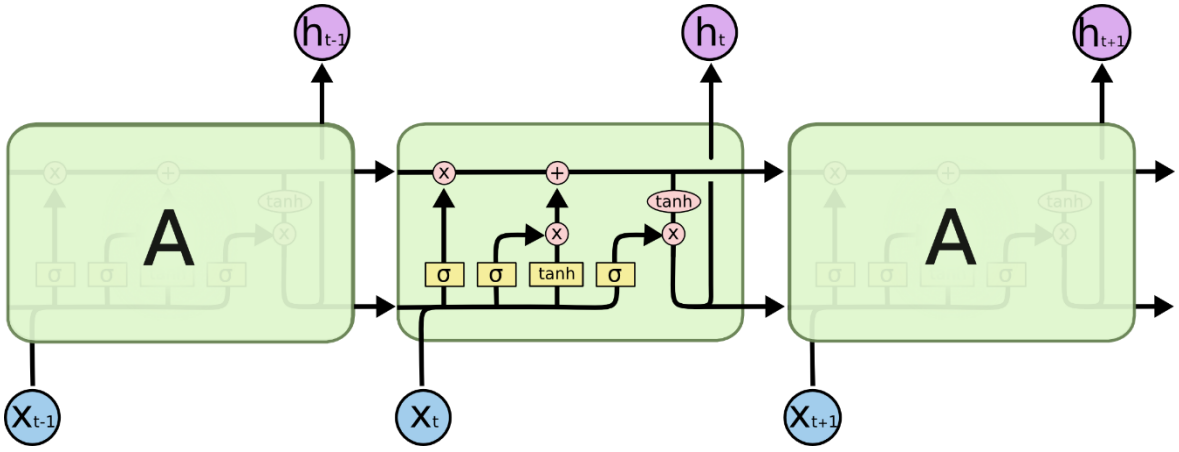
Đầu ra của RNN không chỉ phụ thuộc vào đầu vào mà còn ở trạng thái ẩn, có thể được thay đổi bởi chính nó. Vì vậy, mạng có thể chuyển một số thông tin từ đầu

chuỗi đến cuối và tạo ra đầu ra khác nhau cho cùng một đầu vào trong các bối cảnh khác nhau. Sự phụ thuộc ngữ cảnh này rất quan trọng trong NLP, vì trong ngôn ngữ tự nhiên, một từ duy nhất có thể có nghĩa hoàn toàn khác nhau trong các ngữ cảnh khác nhau và ý nghĩa của toàn bộ câu có thể được thay đổi bằng một từ duy nhất. Tất nhiên, tính linh hoạt như vậy đi kèm với chi phí riêng của nó. RNN thường đòi hỏi nhiều thời gian hơn để đào tạo và có thể tạo ra một số tạo ra các đầu ra không hợp lí bởi các nguyên nhân như biến mất gradient và phụ thuộc xa trong quá trình đào tạo. Vì vậy RNN có thể được coi là một khối xây dựng tiêu chuẩn của các hệ thống cần xử lý đầu vào có độ dài thay đổi. Có một vài phương pháp được đề xuất để giải quyết vấn đề biến mất gradient và phụ thuộc xa, tiêu biểu là LSTM[20].

4. Mạng bộ nhớ dài ngắn LSTM

Mạng LSTM [28] là một dạng đặc biệt của RNN, nó có khả năng học được các phụ thuộc xa. LSTM [29] được giới thiệu bởi Hochreiter & Schmidhuber (1997), và sau đó đã được cải tiến và phổ biến bởi rất nhiều người trong ngành. Chúng hoạt động cực kì hiệu quả trên nhiều bài toán khác nhau nên dần đã trở nên phổ biến như hiện nay. LSTM được thiết kế để tránh được vấn đề phụ thuộc xa (long-term dependency). Việc nhớ thông tin trong suốt thời gian dài là đặc tính mặc định của chúng, chúng ta không cần phải huấn luyện nó để có thể nhớ được. Tức là ngay nội tại của nó đã có thể ghi nhớ được mà không cần bất kì can thiệp nào.

[20] https://en.wikipedia.org/wiki/Long_short-term_memory



Hình 6 LSTM (Nguồn: dominhhai.github.io)

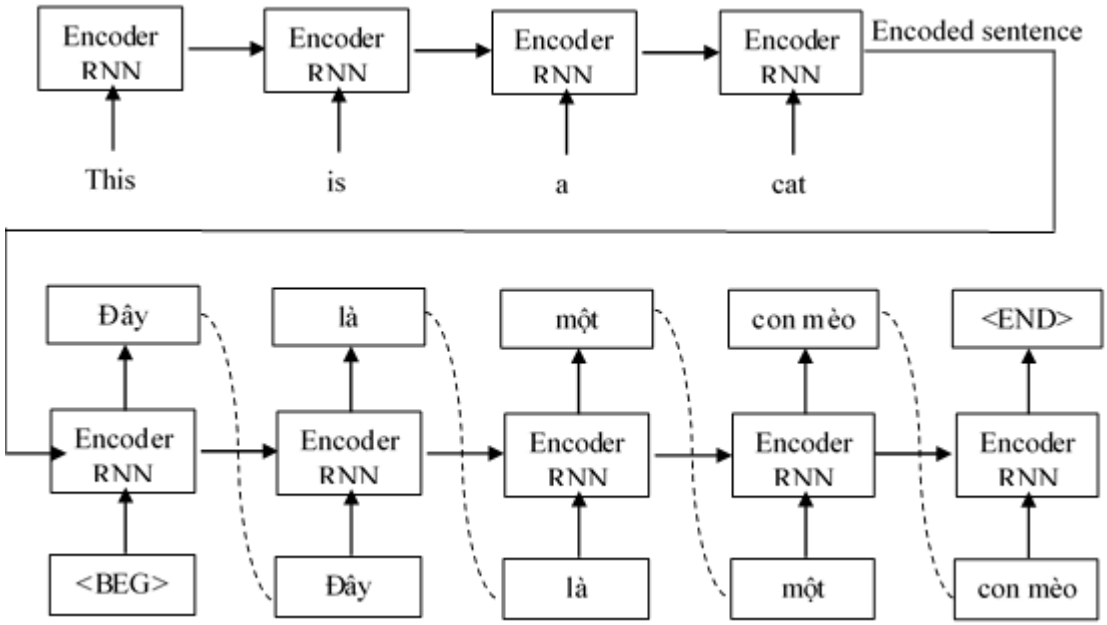
Bộ nhớ dài ngắn LSTM khắc phục tình trạng biến mất gradient (vanishing gradient[21]) tác động lên dữ liệu đầu vào bằng 3 tầng chính: tầng cổng quên, tầng đầu vào, tầng đầu ra. Sự tổ hợp của LSTM mang lại hiệu quả cao khi thực hiện quá trình lựa chọn những nội dung nổi bật đại diện cho dữ liệu cần xử lý, tác động lên chúng bằng những hàm kích hoạt và xuất ra giá trị ở mỗi đầu nơ-ron. Mạng nơ-ron RNN kết hợp LSTM đã và đang thu hút rất nhiều sự chú ý của những nhà nghiên cứu khi thực hiện xử lý những dữ liệu liên tục, đặc biệt trong các vấn đề về xử lý ngôn ngữ tự nhiên (NLP).

5. Mô hình Sequence to Sequence

Một mô hình khác được sử dụng rộng rãi trong NLP được gọi là Encoder-Decoder hay seq2seq[22]. Mô hình ban đầu xuất phát từ dịch máy, khi hệ thống cần chấp nhận một chuỗi các từ trên ngôn ngữ nguồn và tạo ra một chuỗi khác trên ngôn ngữ đích. Ý tưởng đằng sau seq2seq là sử dụng RNN để xử lý chuỗi đầu vào và mã hóa chuỗi này thành một vec-tơ đại diện số có độ dài cố định. RNN này được gọi là bộ mã hóa. Sau đó, bạn đưa vectơ được mã hóa vào một RNN khác, được gọi là bộ giải mã, nó có trách nhiệm tạo ra chuỗi kết quả trong ngôn ngữ đích. Một ví dụ về ý

[21] https://en.wikipedia.org/wiki/Vanishing_gradient_problem
[22] https://nlp.stanford.edu/~johnhew/public/14-seq2seq.pdf

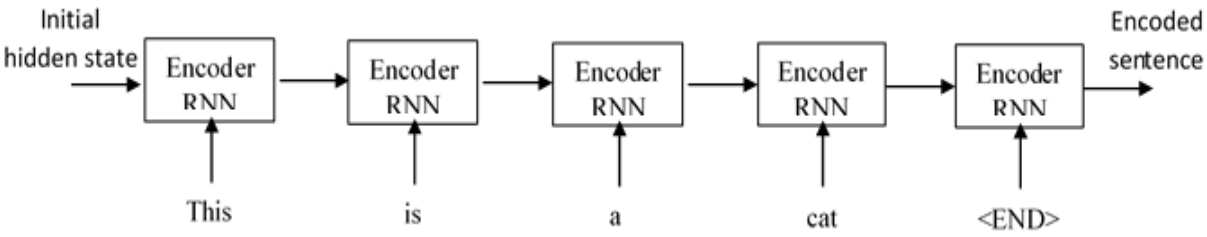
tưởng này được hiển thị như sau, trong đó chúng tôi đang dịch một câu tiếng Anh sang tiếng Việt:



Hình 7 Mô hình seq2seq dịch tiếng Anh sang tiếng Viêt

6. Phương pháp Cross Entropy

Để hiểu phương pháp này, chúng ta cần hiểu một ví dụ. Đó là chúng ta cần tạo một hệ thống dịch máy từ một ngôn ngữ tiếng Anh sang ngôn ngữ tiếng Việt bằng mô hình seq2seq. Giả sử rằng chúng ta có một bộ dữ liệu lớn, các bản dịch mẫu với các câu tiếng Anh mà chúng ta sẽ đào tạo mô hình của mình. Trong phần mã hóa chúng ta chỉ cần áp dụng bộ mã hóa RNN của mình cho câu đầu tiên trong cặp huấn luyện, tạo ra một đại diện được mã hóa của câu. Ứng cử viên rõ ràng cho đại diện này sẽ là trạng thái ẩn được trả về từ RNN cuối cùng. Ở giai đoạn mã hóa, chúng ta bỏ qua các đầu ra của RNN, chỉ tính đến trạng thái ẩn từ RNN cuối cùng. Chúng ta cũng mở rộng câu của mình bằng mã thông báo đặc biệt <END>, báo hiệu cho bộ mã hóa kết thúc câu. Quá trình này được thể hiện trong sơ đồ sau:



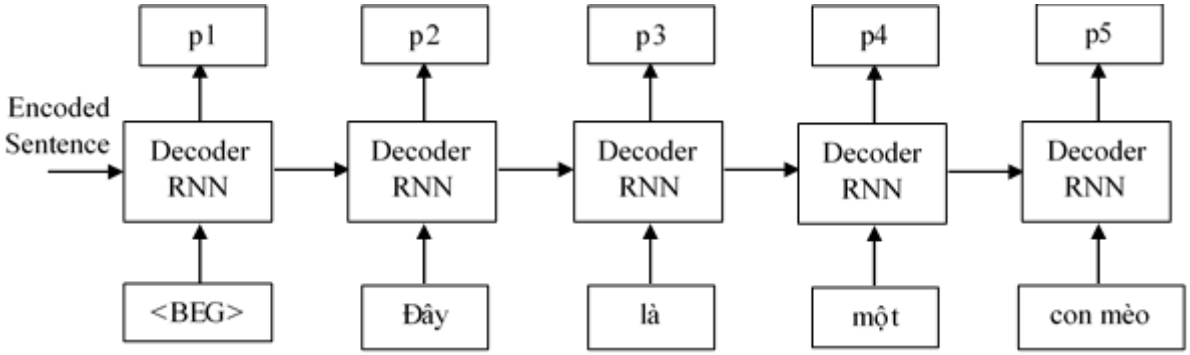
Hình 8 Quá trình mã hóa

Để bắt đầu giải mã, chúng ta chuyển đại diện được mã hóa sang trạng thái ẩn đầu vào của bộ giải mã và chuyển mã thông báo <BEG> làm tín hiệu để bắt đầu giải mã. Trong bước này, bộ giải mã RNN phải trả lại cho chúng ta mã thông báo đầu tiên của câu đã dịch. Tuy nhiên, khi bắt đầu huấn luyện, khi cả RNN của bộ mã hóa và bộ giải mã được khởi tạo với trọng số ngẫu nhiên, đầu ra của bộ giải mã sẽ là ngẫu nhiên và mục tiêu của chúng tôi sẽ là đẩy nó về phía bản dịch chính xác bằng cách sử dụng giảm radient ngẫu nhiên (Stochastic Gradient Descent - SGD).

Cách tiếp cận truyền thống coi vấn đề này là phân loại, khi bộ giải mã của chúng ta cần trả về phân phối xác suất trên các từ (token hay mã thông báo) ở vị trí hiện tại của câu được giải mã. Thông thường, điều này được thực hiện bằng cách chuyển đổi đầu ra của bộ giải mã bằng cách sử dụng mạng chuyển tiếp feed-forwad và tạo ra một vec-tơ, có chiều dài bằng kích thước của từ điển của chúng ta. Sau đó, chúng ta lấy phân phối xác suất này và tổn thất tiêu chuẩn dùng cho việc phân loại: cross-entropy[23] (còn được gọi là log-likelihood loss [30] [19]).

Trong chuỗi được giải mã, mã thông báo <BEG> được tạo ra trên đầu vào. Có hai lựa chọn cho phần còn lại của chuỗi ở đây. Lựa chọn đầu tiên là cung cấp mã thông báo từ câu tham chiếu (câu trả lời chính xác trong tập dữ liệu). Ví dụ: nếu chúng ta có cặp huấn luyện “This is a cat” -> “Đây là một con mèo” cho bộ giải mã và sau đó sử dụng cross-entropy giữa đầu ra của RNN và mã thông báo tiếp theo trong câu. Chế độ đào tạo này được gọi là “teacher forcing” [31] và ở mỗi bước chúng ta cung cấp mã thông báo từ bản dịch chính xác, yêu cầu RNN tạo mã thông báo tiếp theo chính xác. Quá trình này được thể hiện trong sơ đồ sau:

[23] https://en.wikipedia.org/wiki/Cross_entropy



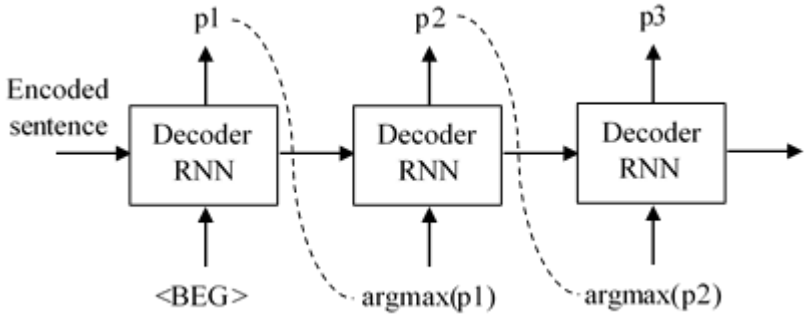
Hình 9 Quá trình giải mã

Công thức tính loss function như sau:
L = xentropy(p1,"Đây") + xentropy(p2,"là") + xentropy(p3,"một") + xentropy(p4,"con mèo") + xentropy(p5,"<END>")

Vì cả bộ giải mã và bộ mã hóa đều là các NN khác nhau, chúng ta chỉ có thể lan quyền ngược (backpropagate) tổn thất (loss) để đẩy cả hai hướng tới phân loại tốt hơn của ví dụ này trong tương lai.

Trong quá trình đào tạo, chúng ta biết cả chuỗi đầu vào và đầu ra mong muốn, vì vậy chúng ta có thể cung cấp chuỗi đầu ra hợp lệ cho bộ giải mã, bộ giải mã tạo mã thông báo tiếp theo của chuỗi.

Sau khi mô hình đã được đào tạo, chúng ta sẽ không có chuỗi đích (vì nó là do mô hình tạo ra). Vì vậy, cách để sử dụng mô hình sẽ là mã hóa chuỗi đầu vào bằng bộ mã hóa và sau đó yêu cầu mỗi decoder RNN tạo một mục của đầu ra tại mỗi thời điểm, đưa mã thông báo đó vào đầu vào của decoder RNN tiếp theo.



Hình 10 Cách giải mã trong chế độ curriculum learning

Việc chuyển kết quả trước đó vào đầu vào đã được thực hiện, nhưng có một hạn chế. Trong quá trình đào tạo, chúng ta đã không yêu cầu bộ giải mã RNN sử dụng

đầu ra của chính nó làm đầu vào, do đó, một lỗi trong quá trình tạo có thể gây nhầm lẫn cho bộ giải mã và dẫn đến đầu ra không tốt.

Để khắc phục điều này, một cách tiếp cận thứ hai để huấn luyện seq2seq xuất hiện, được gọi là học chương trình curriculum learning [16]. Quá trình này được minh họa trên hình 10. Nó không sử dụng toàn bộ đầu ra của decoder RNN mà chỉ sử dụng argmax của đầu ra của decoder RNN trước đó để làm đầu vào cho decoder RNN tiếp theo. Điều này bổ sung sự mạnh mẽ cho bộ giải mã, mang lại kết quả tốt hơn cho ứng dụng thực tế của mô hình. Nhưng có một nhược điểm, chế độ này có thể dẫn đến việc đào tạo rất dài, vì bộ giải mã của chúng ta học cách tạo ra mã thông báo đầu ra mong muốn. Để bù đắp cho điều này, trong thực tế, chúng ta thường đào tạo một mô hình sử dụng teacher forcing[24] và curriculum learning[25], chỉ cần chọn ngẫu nhiên giữa hai mô hình đó cho mỗi batch[26].

7. Điểm đánh giá song ngữ BLEU

BLEU[27] thường sử dụng để so sánh chất lượng đầu ra dịch máy thường được sử dụng trong các vấn đề NLP. BLEU là một trong những cách tiêu chuẩn để so sánh chuỗi đầu ra do máy tạo ra với một số bộ đầu ra tham chiếu. Nó cho phép nhiều đầu ra tham chiếu được sử dụng (một câu có thể được dịch theo nhiều cách khác nhau) và ở cốt lõi, nó tính toán tỷ lệ của unigram, bigram, v.v., được chia sẻ giữa đầu ra được của máy tạo ra và câu tham chiếu. "Bản dịch máy càng gần với bản dịch chuyên nghiệp của con người thì càng tốt" - đây là ý tưởng trung tâm của BLEU.

8. Học tăng cường Reinforcement Learning (RL)

RL và tạo văn bản (text generation) có thể trông rất khác nhau, nhưng có những kết nối có thể được sử dụng để cải thiện chất lượng của các mô hình seq2seq được đào tạo. Điều đầu tiên cần lưu ý là bộ giải mã của chúng ta đưa ra phân phối xác suất

[24] https://machinelearningmastery.com/teacher-forcing-for-recurrent-neural-networks/
[25] https://ronan.collobert.com/pub/matos/2009_curriculum_icml.pdf
[26] https://vi.wikipedia.org/wiki/Batch
[27] https://en.wikipedia.org/wiki/BLEU

ở mỗi bước, rất giống với các mô hình Policy Gradient[28] (PG) [16]. Từ quan điểm này, bộ giải mã của chúng ta có thể được coi là một tác nhân actor cố gắng quyết định mã thông báo (token) nào sẽ tạo ra ở mỗi bước. Có một số lợi thế của việc giải thích quá trình giải mã như vậy.

Trước hết, bằng cách coi quá trình giải mã của chúng ta là ngẫu nhiên, chúng ta có thể tự động tính đến nhiều chuỗi mục tiêu. Ví dụ, có rất nhiều câu trả lời có thể cho 'Xin chào!' 'Bạn khỏe không?'. Bằng cách tối ưu hóa mục tiêu log-likelihood, mô hình của chúng ta sẽ cố gắng học một số trung bình của tất cả các câu trả lời đó, nhưng trung bình của các cụm từ 'tôi ổn', 'cảm ơn!' và 'không tốt lắm' sẽ không nhất thiết là một cụm từ có ý nghĩa. Bằng cách trả về phân phối xác suất và lấy mẫu mã thông báo tiếp theo từ nó, tác nhân agent của chúng ta có thể tìm hiểu cách tạo ra tất cả các biến thể có thể, thay vì học một số câu trả lời trung bình. Sau đó sử dụng phương thức Policy Gradient để đẩy mạnh xác suất của các phần thành công và giảm cho các phần tệ hơn.

Bằng cách đưa tính ngẫu nhiên vào quá trình giải mã, chúng ta có thể lặp lại quá trình giải mã nhiều lần, thu thập các kịch bản giải mã khác nhau từ mẫu đào tạo duy nhất. Điều này có lợi khi tập dữ liệu đào tạo của chúng ta bị giới hạn.

Học tăng cường cho seq2seq có thể được viết theo giải thuật bên dưới:

1. Với mỗi mẫu trong dataset, được mã hóa thành đại diện E bằng bộ mã hóa encoder

2. Khởi tạo mã thông báo token hiện tại bằng T = '<BEG>'

3. Khởi tạo câu đầu ra bằng chuỗi rỗng: Out = []

4. Trong khi T != '<END>'

- Nhận phân phối xác suất của các mã thông báo và trạng thái ẩn mới, truyền đi mã thông báo hiện tại và trạng thái ẩn: p, H = Decoder (T, E)

- Lấy được mã thông báo đầu ra T_{out} từ phân phối xác xuất

- Ghi nhớ phân phối xác xuất p

- Thêm Tout vào câu đầu ra Out += T_{out}

[28] http://www.scholarpedia.org/article/Policy_gradient_methods

- Gán mã thông báo hiện tại $T = T_{out}$, E = H

5. Tính điểm BLEU giữa Out và những câu tham chiếu: Q = BLEU(Out, Outref)

6. Tính gradient

7. Cập nhật mô hình bằng phư ơng pháp giảm gradient ngẫu nhiên SGD[29]

8. Lặp lại cho đến khi hội tụ

9. Huấn luyện trình tự tự phê bình (Self-critical sequence training - SCST)

Cách tiếp cận được mô tả ở phần 8, mặc dù các mặt tích cực của nó, cũng có một số khó khăn. Trước hết, nó gần như vô dụng để đào tạo từ đầu. Ngay cả đối với các hộp thoại đơn giản, chuỗi đầu ra thường có ít nhất năm từ trở lên, mỗi từ được lấy từ từ điển vài nghìn từ. Ví dụ số lượng câu khác nhau có kích thước năm, với từ điển 1000 từ thì số câu khác nhau là 5^{1000}. Vì vậy, xác suất nhận được câu trả lời đúng khi bắt đầu đào tạo (khi trọng số của chúng tôi cho cả bộ mã hóa và giải mã là ngẫu nhiên) là nhỏ không đáng kể. Để khắc phục điều này, chúng ta có thể kết hợp cả hai cách tiếp cận loglikelihood[30] và RL[31] là trước tiên huấn luyện mô hình bằng log-likelihood (cross-entropy hay CE) và sau khi mô hình đạt đến một mức độ chất lượng nào đó, chuyển sang phương pháp REINFORCE để tinh chỉnh mô hình [18].

Một vấn đề khác với phương pháp REINFORCE là phương sai của gradient cao. Vấn đề này được giải quyết trong bài báo Self-critical sequence training for image captioning [24]. Các tác giả của bài báo đã sử dụng bộ giải mã ở chế độ argmax để tạo một chuỗi, nó sau đó được sử dụng để tính toán số liệu tương tự bằng BLEU. Chuyển sang chế độ argmax làm cho quá trình giải mã hoàn toàn xác định và cung cấp số liệu cho REINFORCE policy gradient. Phương pháp này còn được gọi là SCST.

[29] https://en.wikipedia.org/wiki/Stochastic_gradient_descent
[30] https://en.wikipedia.org/wiki/Likelihood_function
[31] https://en.wikipedia.org/wiki/Reinforcement_learning

SCST là một dạng của thuật toán REINFORCE, thay vì ước tính một đường cơ sở "baseline", để chuẩn hóa phần thưởng và giảm phương sai, nó sử dụng đầu ra của thuật toán suy luận ở thời gian thử nghiệm (test-time inference algorithm [33] [34] [35]) của chính nó để chuẩn hóa phần thưởng mà nó nhận được. Sử dụng phương pháp này, việc ước tính phần thưởng (phương pháp actor-critic phải làm) và ước tính chuẩn hóa (thuật toán REINFORCE phải làm làm) được tránh, nó đồng thời dung hòa mô hình theo quy trình suy luận ở thời gian thử nghiệm (test-time inference algorithm [33] [34] [35]) của nó. Quá trình cụ thể như sau:

Tạo chuỗi với RL, phần giải mã decoder LSTM có thể được xem như là 1 chính sách "policy" kí hiệu P_θ, trong đó θ là tập hợp các tham số mạng. Tại mỗi thời điểm "time-step" t, policy lựa chọn một hành động "action" bằng cách sinh ra 1 từ w_t và thu được một trạng thái mới "state". Khi mã thông báo "token" END được tạo, một phần thưởng "reward" r được đưa ra dựa trên điểm số (BLEU hoặc CIDEr, trong bài luận văn này sử dụng BLEU) của câu được dự đoán. Mục tiêu là tối đa hóa phần thưởng mong đợi:

$$L(\theta) = E_{w^s \sim p_\theta}[r(w^s)]$$

trong đó, $w^s = \{w_1^s,\ w_2^s, \dots, w_T^s\}$ là những từ được lấy mẫu ở mỗi bước. Giải thuật REINFORCE [19] cung cấp công thức cho gradient estimation của θ như sau:

$$\nabla L(\theta) \approx r(w^s)\nabla_\theta \log p_\theta (w^s)$$

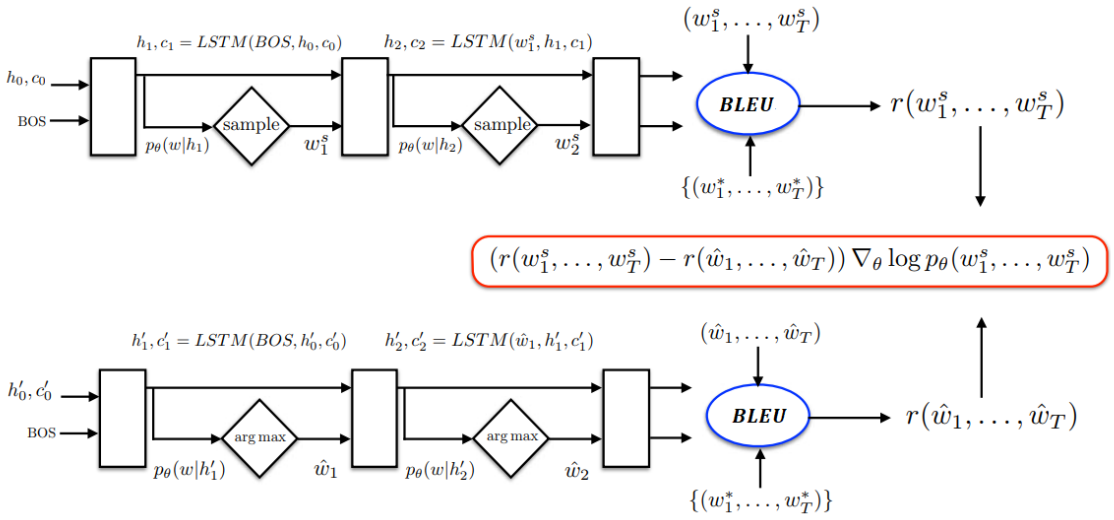
Giảm phương sai với Self-Critical: Chúng tôi giảm phương sai của gradient estimator bằng cách sử dụng phương pháp self-critical [32]:

$$\nabla L(\theta) \approx (r(w^s) - \mathrm{r}(\overline{w}))\nabla_\theta \log p_\theta (w^s)$$

trong đó, $\overline{w}$ là phần thưởng cơ bản được tính bởi mô hình hiện tại theo thuật toán suy luận được sử dụng tại thời điểm thử nghiệm (test time inference algorithm [33] [34] [35]) được xác định bằng công thức:

$$\overline{w} = arg_{w^t} \max(w_t|h_t)$$

Khi đó, những chuỗi có phần thường cao hơn $\overline{w}$ sẽ được tăng xác suất, trong khi những mẫu kết quả có phần thưởng thấp hơn sẽ bị giảm xuống.



Hình 11 Self-critical sequence training (SCST) tại decoder (Nguồn: [13])

10. Word embedding

Word embedding[32] là một nhóm các kỹ thuật đặc biệt trong xử lý ngôn ngữ tự nhiên, có nhiệm vụ ánh xạ một từ hoặc một cụm từ trong bộ từ vựng tới một vector số thực. Từ không gian một chiều cho mỗi từ tới không gian các vector liên tục. Các vector từ được biểu diễn theo phương pháp word embedding thể hiện được ngữ nghĩa của các từ, từ đó ta có thể nhận ra được mối quan hệ giữa các từ với nhau (tương đồng, trái nghịch, ...).

Các phương pháp thường được sử dụng trong word embedding bao gồm:

- Giảm kích thước của ma trận đồng xuất hiện.
- Neural network (Word2vec[33], GloVe[34],...)
- Sử dụng các mô hình xác xuất, …

[32] https://en.wikipedia.org/wiki/Word_embedding
[33] https://en.wikipedia.org/wiki/Word2vec
[34] https://en.wikipedia.org/wiki/GloVe_(machine_learning)

Trong bài luận văn này chúng tôi sử dụng word embedding được cung cấp sẵn của pytorch – Word Embedding with Pytorch[35].

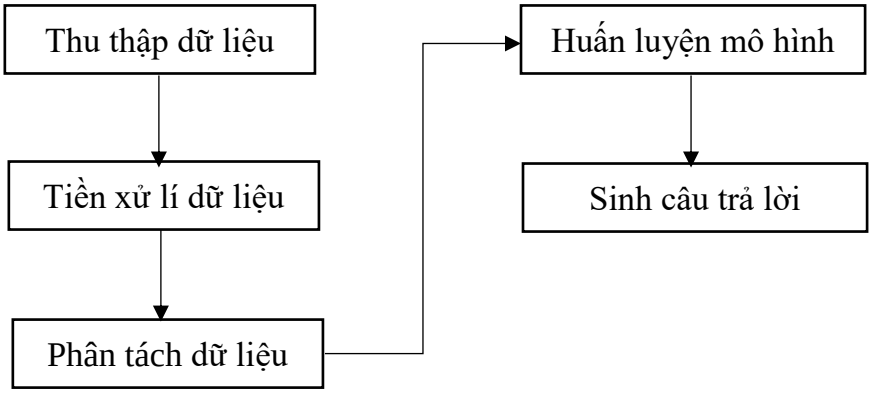
Theo đó, Pytorch sử dụng phân phối ngữ nghĩa (Distributional semantics[36]) (hay sự giống nhau về ngữ nghĩa - semantic similarity) để tạo word embedding. Pytorch dùng word embedding như một tham số của mô hình và sau đó được cập nhật trong quá trình huấn luyện.

[35] https://pytorch.org/tutorials/beginner/nlp/word_embeddings_tutorial.html
[36] https://en.wikipedia.org/wiki/Distributional_semantics

CHƯƠNG 3 – CÀI ĐẶT GIẢI PHÁP

Chương này trình bày các bước cài đặt giải pháp.

1. Các bước thực hiện



Hình 12 Các bước thực hiện

- Bước thu thập dữ liệu: dữ liệu phụ đề phim gồm 30.000 dòng dữ liệu được lấy từ các bộ phim trên Subscene[37].

- Bước tiền xử lý dữ liệu: bước này đọc dữ liệu từ file kết quả cảu bước thu thập, tiến hành xóa bỏ những cặp câu vô nghĩa, xóa bỏ những câu bị trùng, có khoảng trắng, … và ghi vào file kết quả.

- Bước phân tách dữ liệu: đọc dữ liệu từ bước tiền xử lý, tách từ theo nghĩa tiếng Việt, tạo tập từ điển và lưu kết quả cho quá trình huấn luyện.

- Bước huấn luyện dữ liệu: Huấn luyện với mô hình cross-entropy với số lượng epoch nhất định. Sau đó sử dụng mô hình với các tham số đã train của cross-entropy để tiếp tục huấn luyện bằng mô hình self-cristic.

- Bước sinh câu trả lời: nhận dữ liệu input là câu hỏi đầu vào, tiến hành phân tích câu hỏi dựa trên các ngữ cảnh học được. Sử dụng mô hình sinh ra câu trả lời và hiển thị kết quả.

2. Xây dựng mô hình

[37] https://subscene.com/

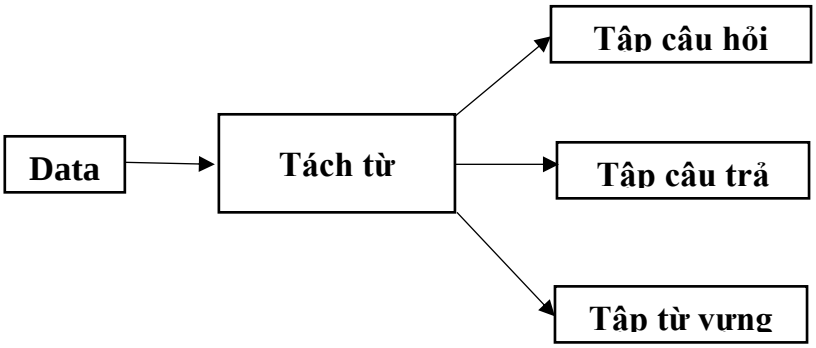
2.1 Thu thập dữ liệu

Sử dụng tập data dữ liệu phụ đề phim gồm 30.000 dòng dữ liệu.

2.2 Bước tiền xử lí dữ liệu

- Loại bỏ các kí tự đặc biệt như: @, #, $, %, ^, &, *, …
- Xóa các kí tự phân tách câu: ?, !, .,
- Xóa bỏ các thẻ HTML:
, <i>, …
- Xóa câu bị lỗi font: B#7897; #273; #7891
- Xóa các câu bị mất khoảng trắng: Cónghĩalà ôngbiết

2.3 Phân tách dữ liệu



Hình 13 Các bước phân tách dữ liệu

Dữ liệu sau bước tiền xử lí sẽ được chia thành tập câu hỏi và tập câu trả lời để thuận tiện cho việc học, sử dụng các câu trong tập đó để tách các từ riêng biệt bằng công cụ Underthesea[38]. Ví dụ, câu "Anh là sinh viên sao" sẽ được tách thành "Anh, là, sinh_viên, sao". Bước tiếp theo sẽ chọn ra những từ thông dụng nhất để xây dựng tập từ vựng, là tiền đề để xây dựng các tập đặc trưng đưa vào mô hình để huấn luyện.

2.4 Huấn luyện mô hình

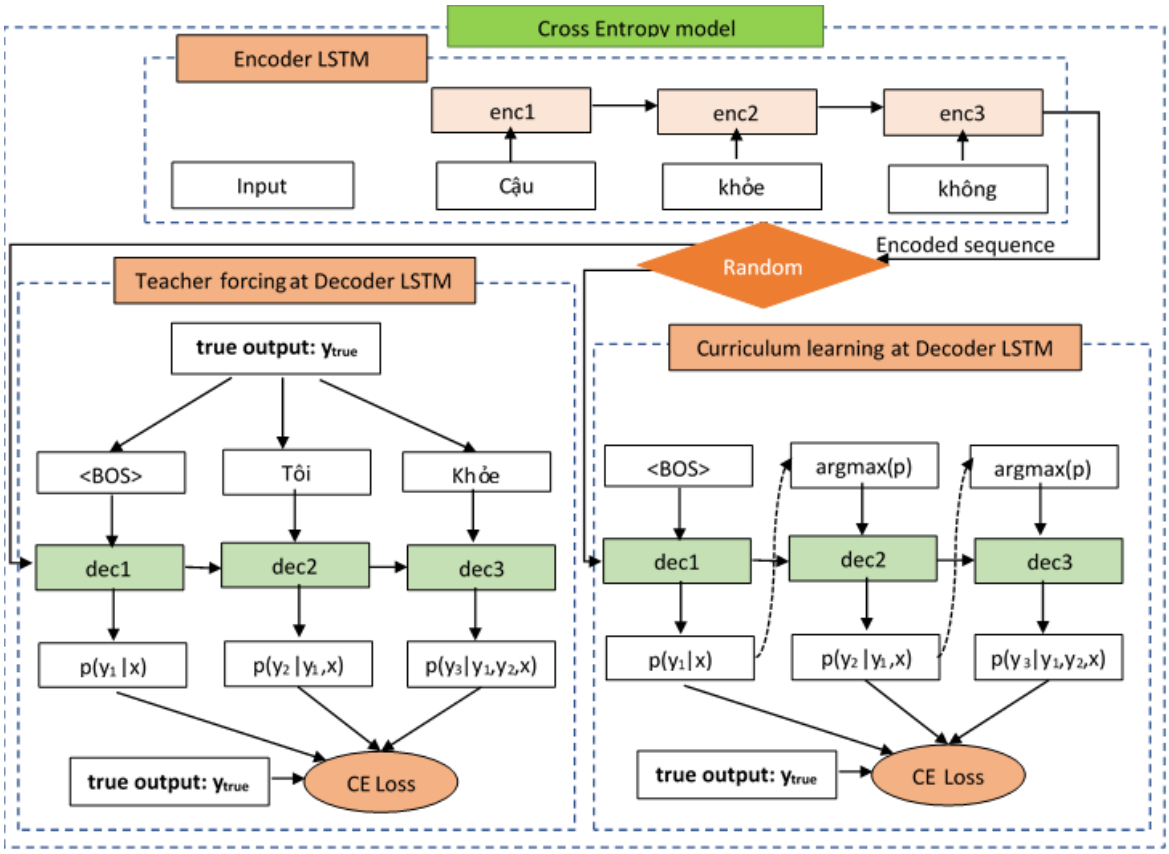
Bước này sẽ huấn luyện 2 mô hình chính. Đầu tiên là huấn luyện mô hình cross-entropy (CE) kết hợp với seq2seq và LSTM với số epoch[39] cụ thể. Sau đó sử dụng mô đã huấn luyện ở trên kết hợp RL và SCST để huấn luyện và tinh chỉnh mô hình. Với mỗi lần huấn luyện trên bao gồm các bước sau:

- Định nghĩa tham số: batch size, learning rate, num epoch, …

[38] https://underthesea.readthedocs.io/en/latest/readme.html
[39] https://stackoverflow.com/questions/31155388/meaning-of-an-epoch-in-neural-networks-training

- Load data: Đọc dữ liệu từ dataset.
- Split data: tập dữ liệu sẽ được chia thành tập train và tập test với tỉ lệ tùy chỉnh.
- Train: Huấn luyện mô hình

2.4.1 Huấn luyện mô hình cross entropy

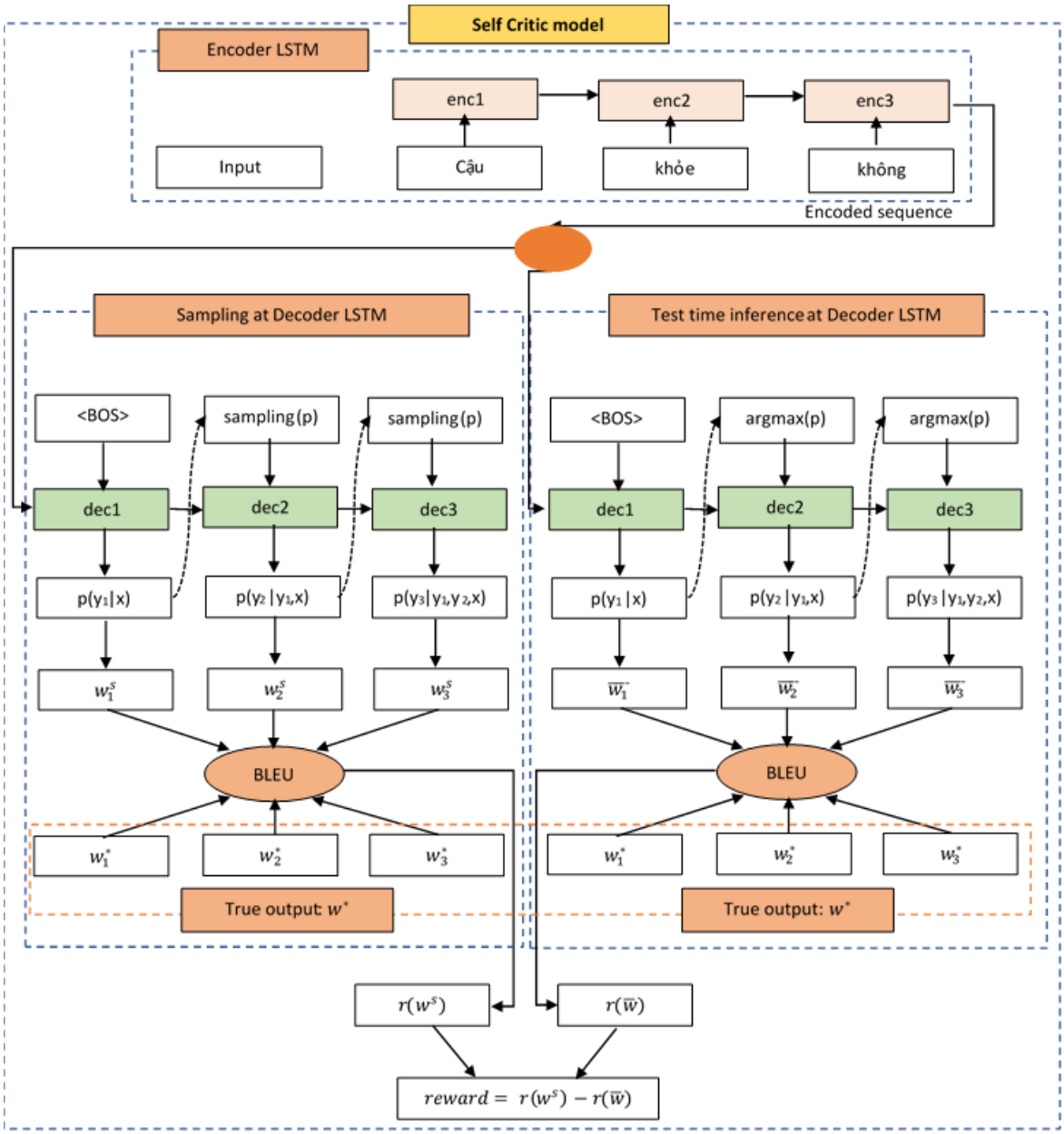


Hình 14 Mô hình cross entropy

Trong mô hình cross entropy, các vector từ được chuyển cho bộ mã hóa để mã hóa thành một đại diện gọi là encoded sequence. Sau đó, hệ thống sẽ chọn ngẫu nhiên giữa bộ giải mã dung phương pháp "teacher forcing[40]" hoặc "curriculum learning[41]". Sau khi 1 trong 2 phương pháp được chọn thì chúng ta có được CE Loss (cross entropy loss). Nó được tính dựa trên đầu ra của bộ mã hóa và đầu ra chính xác. Việc này được lặp lại đến một số epoch nhất định thì chuyển sang huấn luyện mô hình self-critic. Các công thức tính được mô tả ở phần 6 (Chương 2 cơ sở lí thuyết).

[40] https://cedar.buffalo.edu/~srihari/CSE676/10.2.1%20TeacherForcing.pdf
[41] http://ronan.collobert.com/pub/matos/2009_curriculum_icml.pdf

2.4.2 Huấn luyện mô hình self critic

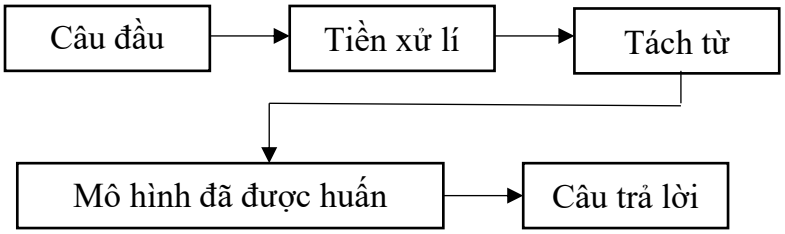


Hình 15 Mô hình Self-critic

Mô hình self-critic nhận mô hình đã được huấn luyện từ mô hình cross entropy. Sau khi có được encoded sequence (câu đã được mã hóa), câu này được giải mã với 2 cách khác nhau Sampling (giải mã ngẫu nhiên) và Test time inference (thuật toán suy luận thời gian thử nghiệm). Các phân phối xác xuất đầu ra của bộ giải mã (p_1,p_2, …) được dùng để lấy ra các từ tương ứng trong từ điển và các từ này được so sánh với các từ chính xác (đầu ra chính xác của đầu vào) bằng điểm số BLEU và nó cũng

được xem là reward. Với 2 cách giải mã trên, chúng tôi có được 2 reward. Và hiệu của 2 reward này được dùng để tính gradient. Các công thức được mô tả cụ thể trong phần 9 (Chương 2 cơ sở lí thuyết).

2.5 Sinh câu trả lời



Hình 16 Các bước sinh câu trả lời

Khi người dùng nhập câu truy vấn, chúng cũng sẽ trải qua bước tách từ. Dựa vào tập từ điển chúng sẽ được mã hóa thành một chuỗi các con số có độ dài cố định (word embedding). Bộ giải mã sẽ sinh ra câu trả lời dựa vào chuỗi đó cho tới khi gặp dấu hiệu kết thúc câu.

CHƯƠNG 4 – THỰC NGHIỆM VÀ ĐÁNH GIÁ KẾT QUẢ

1. Kết quả thực nghiêm

Profile:

- Intel(R) Xeon(R) CPU @ 2.30GHz 2 core
- RAM 8G
- Pytorch with GPU
- 30.000 dòng dữ liệu
- Epoch: 100 cross-entropy, 1000 seft-critic
- BLEU: 0.07
- 5 phút / 1 epoch

Sau hơn 3 ngày (48 giờ), chúng tôi đã huấn luyện mô hình với cấu hình trên và được 1000 epoch. Kết quả thực nghiệm trong bảng bên dưới.

input: xin chào output: chào cưng	input: chúc mừng sinh nhật output: tạm_biệt đã rất con
input: chào cậu nhé output: chào anh	input: cậu biết mấy giờ rồi không output: không tớ muốn lên
input: chào cậu nhé output: chào anh	input: tôi đang rất buồn output: không cô không ở đây
input: cậu tên gì thế output: không	input: chúc mừng năm mới output: dì mọi chuyện đó thì sao
input: cậu tên gì thế output: không	input: cậu bao nhiêu tuổi output: có_lẽ có_vẻ
input: cậu ổn chứ output: ừ tôi ổn	input: chúng ta đi ăn nhé output: sao cơ không
input: cậu ăn sáng chưa output: chưa	input: tôi đang buồn output: không có gì
input: cậu có gì cho tôi ăn nào output: tớ nói đồ ăn	input: tôi yêu cậu output: ông nói gì
input: cậu sống ở đâu output: ở bến tàu	

input: cậu có tiền không output: có input: cậu rảnh không output: không input: cậu có biết mặt trời không output: không chẳng hiểu	

2. Đánh giá kết quả

Tập dữ liệu của chúng tôi được lấy từ phụ để của các bộ phim trên Subscene[42]. Sau khi xử lí, chúng tôi có được tập dữ liệu gồm 30.000 dòng bao gồm 15.000 cặp câu. Mỗi cặp câu gồm một câu hỏi, một câu trả lời.

Chúng tôi đánh giá mô hình bằng phương pháp BLEU và thu được điểm BLEU trung bình là 0,07 cho những câu trả lời dựa trên tập test. Sau khi chạy thực nghiệm mô hình và đặt câu hỏi cho Bot, kết quả thu được khi đặt câu hỏi cho Bot là tỷ lệ trả lời chính xác còn thấp. Tuy nhiên, có thể nhận thấy rằng Bot đã có thể trả lời được những gì đã được học với chất lượng khả quan, phụ thuộc nhiều vào chất lượng của tập dữ liệu đầu vào.

Luận văn áp dụng phương pháp self-critic [24], theo bài báo này, họ áp dụng self-critic vào việc mô tả hình ảnh (image caption) với đánh BLEU là 31,9. Kết quả của luận văn thì BLEU là 0,07 cao hơn 0,01 so với luận văn của Đào Văn Chiến [35] 0,06. Nguyên nhân do đầu vào chưa chưa đủ. Và có sự khác biệt là đầu vào của bài báo trên là ảnh, và đầu vào của luận văn là text. Đôi khi chatbot sẽ đưa ra một số câu trả lời không đúng ngữ pháp và cú pháp, nguyên nhân là do chatbot chưa được học những câu này, hoặc những câu tương tự những câu này.

[42] https://subscene.com/

CHƯƠNG 5 – KẾT LUẬN

1. Kết quả đạt được và hạn chế

Vốn dĩ ảnh là tập hợp các điểm ảnh độc lập nên sự ra đời của kỹ thuật SCST làm cho các điểm ảnh đó có mối liên hệ với nhau (sử dụng trong việc mô tả hình ảnh – image captioning). Trong khi đó, NLP chú trọng các mối liên hệ giữa các từ trong câu. Chúng tôi thấy được sự tương tự trong hai trường hợp trên. Do đó, chúng tôi đã đề xuất thành công áp dụng kỹ thật SCST – (kỹ thuật này chưa được bài báo nào áp dụng cho NLP) cho hệ thống trả lời tự động của luận văn này.

Luận văn này đã nghiên cứu các lý thuyết và vấn đề trong quá trình thiết lập, huấn luyện và xây dựng một hệ thống đối thoại cho Tiếng Việt trên miền mở. Từ đó, đã xây dựng được mô hình đối thoại tự động cho tiếng Việt trên miền dữ liệu mở được lấy từ dữ liệu phụ đề phim gồm 30,000 dòng dữ liệu. Kết quả ban đầu đạt được là tiền đề để tạo ra các trợ lý ảo, xây dựng các ứng dụng thông minh có thể hiểu được ngôn ngữ tiếng Việt. Có khả năng áp dụng vào các bài toán thực tế, ví dụ như các hệ thống hỗ trợ hỏi đáp về y khoa, tư vấn mua hàng, hỗ trợ giải đáp kỹ thuật cho khách hàng, các dịch vụ khác, … Đặc biệt, có thể tạo ra một trợ lý ảo mà có thể theo dõi sức khỏe và tương tác với cá nhân mà chúng tôi đang hướng tới. Rõ ràng với kết quả hiện tại, Bot này chưa thể trở thành ứng dụng trên thị trường. Luận văn đã thành công trong việc xây dựng bằng một phương pháp mới Self-critical for sequence training SCST. Qua những kết quả ban đầu đạt được, chúng tôi nhận thấy còn nhiều việc cần phải làm để cho ra sản phẩm tốt hơn. Nhưng cách tiếp cận này ban đầu đã cho ra những kết quả khá tích cực. Định hướng tiếp theo, chúng tôi sẽ xử lí thêm phần dữ liệu để tạo ra mô hình có thể trả lời sát với ngữ cảnh và đạt chất lượng tốt hơn, đồng thời ứng dụng các công nghệ mới vào để cho sản phẩm càng hoàn thiện hơn nữa.

2. Hướng phát triển

Chatbot hiện vẫn còn những câu trả lời chưa hợp lí. Nguyên nhân do tập dữ liệu nhỏ. Hướng phát triển của luận văn là bổ sung tập dữ liệu lớn để cho dữ liệu phong

phú hơn. Ngoài ra còn có thể áp dụng beam search[43] [26] kết hợp với attention [26] để cải thiện kết quả.

[43] https://en.wikipedia.org/wiki/Beam_search

TÀI LIỆU THAM KHẢO

[1] Sutskever, I., Vinyals, O., & Le, Q. V., "Sequence to sequence learning with neural networks," *Advances in neural information processing systems,* pp. 3104-3112, 2014.

[2] Sordoni, A., Galley, M., Auli, M., Brockett, C., Ji, Y., Mitchell, M., ... & Dolan, B., "A neural network approach to context-sensitive generation of conversational responses," *arXiv preprint arXiv:1506.06714.,* 2015.

[3] Ritter, A., Cherry, C., & Dolan, W. B., "Data-driven response generation in social media," *Proceedings of the conference on empirical methods in natural language processing,* pp. 583-593, 2011.

[4] Vinyals, O., & Le, Q., "A neural conversational model," *arXiv preprint arXiv:1506.05869.,* 2015.

[5] Serban, I. V., Sordoni, A., Bengio, Y., Courville, A., & Pineau, J. , "Building end-to-end dialogue systems using generative hierarchical neural network models," *Thirtieth AAAI Conference on Artificial Intelligence,* 2016.

[6] Li, J., Monroe, W., Ritter, A., Galley, M., Gao, J., & Jurafsky, D., "Deep reinforcement learning for dialogue generation," *arXiv preprint arXiv,* 2016.

[7] Ranzato, M. A., Chopra, S., Auli, M., & Zaremba, W., "Sequence level training with recurrent neural networks," *arXiv preprint arXiv:1511.06732.,* 2015.

[8] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Machine learning,* pp. 229-256, 1992.

[9] Papineni, K., Roukos, S., Ward, T., & Zhu, W. J., "BLEU: a method for automatic evaluation of machine translation," *Proceedings of the 40th annual meeting on association for computational linguistics,* pp. 311-318, 2002.

[10] Vedantam, R., Lawrence Zitnick, C., & Parikh, D., "Cider: Consensus-based image description evaluation," *Proceedings of the IEEE conference on computer vision and pattern recognition,* pp. 4566-4575, 2015.

[11] Banerjee, S., & Lavie, A, "METEOR: An automatic metric for MT evaluation with improved correlation with human judgments," *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization,* pp. 65-72, 2005.

[12] Sutton, R. S., & Barto, A. G., "Introduction to reinforcement learning," *Cambridge: MIT press,* p. Vol. 135, 1998.

[13] Rennie, S. J., Marcheret, E., Mroueh, Y., Ross, J., & Goel, V, "Self-critical sequence training for image captioning," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition,* pp. 7008-7024, 2017.

[14] Schulman, J., Moritz, P., Levine, S., Jordan, M., & Abbeel, P., "High-dimensional continuous control using generalized advantage estimation," *arXiv preprint arXiv:1506.02438.,* 2015.

[15] Zaremba, W., & Sutskever, I., "Reinforcement learning neural turing machines-revised," *arXiv preprint arXiv:1505.00521.*, 2015.

[16] Mnih, A., & Gregor, K., " Neural variational inference and learning in belief networks," *arXiv preprint arXiv:1402.0030.*, 2014.

[17] D. Duong, "https://viblo.asia/p/tong-quan-ve-artificial-neural-network-1VgZvwYrlAw," 6 7 2018. [Online].

[18] D. M. Hai, "Mạng nơ-ron nhân tạo - Neural Networks," 04 23 2018. [Online]. Available: https://dominhhai.github.io/vi/2018/04/nn-intro/.

[19] H. D. Minh, "[NN] Về lan truyền ngược - Backpropagation," 27 4 2018. [Online]. Available: https://dominhhai.github.io/vi/2018/04/nn-bp/.

[20] C. Olah, "Understanding LSTM Networks," 8 2015. [Online]. Available: https://colah.github.io/posts/2015-08-Understanding-LSTMs/.

[21] Hochreiter, S., & Schmidhuber, J., "LSTM can solve hard long time lag problems," *Advances in neural information processing systems*, pp. 473-479, 1997.

[22] P. W. Glynn, " Likelihood ratio gradient estimation for stochastic systems," *Communications of the ACM*, pp. 75-84, 1990.

[23] Paulus, R., Xiong, C., & Socher, R. , "A deep reinforced model for abstractive summarization," *arXiv preprint arXiv*, 2017.

[24] Guo, T., Chang, S., Yu, M., & Bai, K., "Improving Reinforcement Learning Based Image Captioning with Natural Language Prior," *arXiv preprint arXiv:1809.06227.*, 2018.

[25] Đ. V. Chiến, "xây dựng hệ thống trả lời tự động dựa trên mô hình seqence to seqence với lstm," 2018.

[26] Wiseman, S., & Rush, A. M. , "equence-to-sequence learning as beam-search optimization," *Sequence-to-sequence learning as beam-search optimization*, arXiv preprint arXiv:1606.02960..

[27] Chorowski, J. K., Bahdanau, D., Serdyuk, D., Cho, K., & Bengio, Y, "Attention-based models for speech recognition," *Advances in neural information processing systems*, pp. 577-585, 2015.

[28] Q. Phạm, "Xây dựng mô hình không gian vector cho Tiếng Việt," 09 09 2018. [Online]. Available: https://viblo.asia/p/xay-dung-mo-hinh-khong-gian-vector-cho-tieng-viet-GrLZDXr2Zk0.

[29] Sordoni, A., Galley, M., Auli, M., Brockett, C., Ji, Y., Mitchell, M., ... & Dolan, B., "A neural network approach to context-sensitive generation of conversational responses," *arXiv preprint arXiv:1506.06714.*, 2015.

[30] Shang, L., Lu, Z., & Li, H., "Neural responding machine for short-text conversation," *arXiv preprint arXiv:1503.02364.*.

[31] Li, J., Galley, M., Brockett, C., Gao, J., & Dolan, B., "A diversity-promoting objective function for neural conversation models," *arXiv preprint arXiv:1510.03055.*, 2015.

[32] Wen, T. H., Gasic, M., Mrksic, N., Su, P. H., Vandyke, D., & Young, S. , "Semantically conditioned lstm-based natural language generation for spoken dialogue systems," *arXiv preprint arXiv:1508.01745.*, 2015.

[33] Yao, K., Zweig, G., & Peng, B., "Attention with intention for a neural network conversation model," *arXiv preprint arXiv:1510.08565.*, 2015.

[34] Luan, Y., Ji, Y., & Ostendorf, M., "LSTM based conversation models.," *arXiv preprint arXiv:1603.09457.*, 2016.

[35] Xu, Z., Liu, B., Wang, B., Sun, C., & Wang, X., "Incorporating loose-structured knowledge into lstm with recall gate for conversation modeling," *arXiv preprint arXiv:1605.05110, 3.*, 2016.

[36] Wen, T. H., Vandyke, D., Mrksic, N., Gasic, M., Rojas-Barahona, L. M., Su, P. H., ... & Young, S., "A network-based end-to-end trainable task-oriented dialogue system," *arXiv preprint arXiv:1604.04562.*, 2016.

[37] Li, J., Galley, M., Brockett, C., Spithourakis, G. P., Gao, J., & Dolan, B., "A persona-based neural conversation model," *arXiv preprint arXiv:1603.06155.*, 2016.

[38] Su, P. H., Gasic, M., Mrksic, N., Rojas-Barahona, L., Ultes, S., Vandyke, D., ... & Young, S, "Continuously learning neural dialogue management," *arXiv preprint arXiv:1606.02689.*, 2016.

[39] R. Salakhutdinov, "Learning deep generative models," *Annual Review of Statistics and Its Application*, pp. 361-385, 2015.