ĐẠI HỌC QUỐC GIA HÀ NỘI

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ

NHỮ BẢO VŨ

XÂY DỰNG MÔ HÌNH ĐỐI THOẠI CHO TIẾNG VIỆT TRÊN MIỀN MỞ DỰA VÀO PHƯƠNG PHÁP HỌC CHUỖI LIÊN TIẾP

LUẬN VĂN THẠC SĨ CÔNG NGHỆ THÔNG TIN

HÀ NỘI - 2016

ĐẠI HỌC QUỐC GIA HÀ NỘI

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ

NHỮ BẢO VŨ

XÂY DỰNG MÔ HÌNH ĐỐI THOẠI CHO TIẾNG VIỆT TRÊN MIỀN MỞ DỰA VÀO PHƯƠNG PHÁP HỌC CHUỖI LIÊN TIẾP

Ngành: Công nghệ thông tin

Chuyên ngành: Hệ thống thông tin

Mã số: 60480104

LUẬN VĂN THẠC SĨ CÔNG NGHỆ THÔNG TIN

NGƯỜI HƯỚNG DẪN KHOA HỌC: TS. Nguyễn Văn Nam

HÀ NỘI – 2016

LỜI CAM ĐOAN

Tôi là Nhữ Bảo Vũ, học viên khóa K21, ngành Công nghệ thông tin, chuyên ngành Hệ Thống Thông Tin. Tôi xin cam đoan luận văn "Xây dựng mô hình đối thoại cho tiếng Việt trên miền mở dựa vào phương pháp học chuỗi liên tiếp" là do tôi nghiên cứu, tìm hiểu và phát triển dưới sự hướng dẫn của TS. Nguyễn Văn Nam. Luận văn không phải sự sao chép từ các tài liệu, công trình nghiên cứu của người khác mà không ghi rõ trong tài liệu tham khảo. Tôi xin chịu trách nhiệm về lời cam đoan này.

Hà Nội, ngày tháng năm 2016

LỜI CẢM ƠN

Đầu tiên tôi xin gửi lời cảm ơn tới các thầy cô Trường Đại học Công nghệ, Đại học Quốc Gia Hà Nội đã tận tình giảng dạy và truyền đạt kiến thức trong suốt khóa học cao học vừa qua. Tôi cũng xin được gửi lời cảm ơn đến các thầy cô trong Bộ môn Hệ thống thông tin cũng như Khoa công nghệ thông tin đã mang lại cho tôi những kiến thức vô cùng quý giá và bổ ích trong quá trình học tập tại trường.

Đặc biệt xin chân thành cảm ơn thầy giáo, TS. Nguyễn Văn Nam, người đã định hướng, giúp đỡ, trực tiếp hướng dẫn và tận tình chỉ bảo tôi trong suốt quá trình nghiên cứu, xây dựng và hoàn thiện luận văn này.

Tôi cũng xin được cảm ơn tới gia đình, những người thân, các đồng nghiệp và bạn bè đã thường xuyên quan tâm, động viên, chia sẻ 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.

Hà Nội, ngày tháng năm 2016

MỤC LỤC

DANH MỤC KÝ HIỆU VÀ CÁC CHỮ VIẾT TẮT

Từ viết tắt	Từ chuẩn	Diễn giải
AI	Artificial Intelligence	Trí tuệ nhân tạo
ML	Machine Learning	Máy học, máy móc có khả năng học tập
ANN	Artificial Nerual Network	Mạng nơ ron nhân tạo
RNN	Recurrent Neural Network	Mạng nơ ron tái phát
CNN	Convolutional Neural Networks	Mạng nơ ron tích chập
LSTM	Long short-term memory	Mạng cải tiến để giải quyết vấn đề phụ thuộc quá dài
NLP	Natural Languague Processing	Xử lý ngôn ngữ tự nhiên
VNTK	Vietnamese Languague Toolkit	Bộ công cụ xử lý ngôn ngữ tiếng Việt
NLTK	Natural Language Toolkit	Bộ công cụ xử lý ngôn ngữ tự nhiên bằng Python
Python	Python	Ngôn ngữ lập trình python, nền tảng lập trình phía máy chủ
SDK	Support Development Kit	Bộ công cụ hỗ trợ phát triển
CPU	Central Processing Unit	Bộ xử lý trung tâm
GPU	Graphics Processing Unit	Bộ vi xử lý chuyên dụng nhận nhiệm vụ tăng tốc, xử lý đồ họa cho bộ vi xử lý trung tâm CPU
API	Application Programming Interface	Giao diện lập trình ứng dụng
QA	Question Answering	Các cặp câu hỏi đáp
BLEU	Bilingual Evaluation Understudy	Thuật toán để đánh giá chất lượng của một văn bản được sinh ra từ một mô hình ngôn ngữ tự nhiên

DANH MỤC HÌNH VẼ VÀ ĐỒ THỊ

TÓM TẮT

Con người và máy móc luôn có các mối quan hệ chặt chẽ với nhau. Chúng ta đang tham gia vào một sự thay đổi văn hóa rất lớn trong vài năm qua, vì con người vốn là sinh vật chịu trách nhiệm về hành động, trong khi máy móc là thiết bị an toàn trong một số tình huống không mong muốn. Tuy nhiên, hiện nay các vai trò đã được đảo ngược, các máy móc thường phụ trách công việc trong khi con người chỉ đơn giản là giám sát, theo dõi.

Mô hình hóa đối thoại là một nhiệm vụ quan trọng trong bài toán hiểu ngôn ngữ tự nhiên, và máy học thông minh. Các phương pháp tiếp cận trước đây thường giới hạn trong một lĩnh vực cụ thể, ví dụ như đặt vé trực tuyến, tư vấn ghi danh trực tuyến, tìm kiếm thông tin y tế, … và yêu cầu phải thiết kế được các bộ luật học bằng tay, mất nhiều công sức mà hiệu quả đạt được không cao, khó mở rộng mô hình và các ứng dụng có liên quan.

Trong đề tài này, chúng tôi sẽ nghiên cứu, xây dựng một mô hình đối thoại cho tiếng Việt, dựa trên phương pháp học chuỗi liên tiếp, sequence-to-sequence, để sinh ra câu trả lời từ một chuỗi đầu vào tương ứng. Lợi thế của phương pháp này là mô hình có thể được huấn luyện end-to-end trên tập dữ liệu có sẵn, và yêu cầu ít hơn các luật bằng tay. Kết quả chính của chúng tôi đạt được một mô hình đối thoại sử dụng các mạng học sâu để sinh ra câu trả lời bằng tiếng Việt, tương ứng với một câu hỏi chuỗi đầu vào. Mô hình ban đầu đã cho kết quả rất tính cực, có thể giải quyết được những vấn đề cơ bản về ngữ nghĩa, ngữ cảnh và tính cách riêng trong hệ thống đối thoại.

GIỚI THIỆU CHUNG

Máy học (ML) và trí tuệ nhân tạo (AI) đang nhanh chóng được đưa vào ứng dụng trong các sản phẩm công nghiệp, thúc đẩy tính dân chủ về trí thông minh, mặc dù điều này chỉ đúng đối với tri thức bậc thấp. Bởi vì một mặt, một lượng lớn các dịch vụ, các công cụ sẵn sàng cho người dùng cuối, mặt khác, quyền lực thực sự đang tập trung vào tay của các ông lớn với các dữ liệu lớn sẵn có và tài nguyên tính toán thực sự để khai thác AI/ML đến các mức độ cao cấp hơn.

1. Động lực nghiên cứu và tính cấp thiết của bài toán thực tế

Trong bối cảnh mạng xã hội đã trở nên rất phổ biến như hiện nay, con người kết nối với con người thông qua mạng xã hội, bất cứ thời gian nào và ở bất cứ nơi đâu. Sẽ thật tốt hơn nếu có một hệ thống tự động thông minh hỗ trợ con người bằng cách trò chuyện, có khả năng nhắc nhở, làm trợ lý công việc và có thể theo dõi tình trạng sức khỏe cá nhân mọi lúc, mọi nơi.

Khái niệm Trợ lý ảo, Chatbot, hay Hệ thống trả lời tự động đang là chủ đề rất nóng từ đầu năm nay 2016, khi chính thức các công ty lớn như Microsoft (Cortana), Google (Google Assistant), Facebook (M), Apple (Siri), Samsung (Viv), WeChat, Slack đã giới thiệu các trợ lý ảo của mình, là các hệ thống trả lời tự động. Chính thức đặt cược lớn vào cuộc chơi chatbot, với mong muốn tạo ra một trợ lý ảo thực sự thông minh tồn tại trong hệ sinh thái các sản phẩm của mình.

Không chỉ các ông lớn! Một làn sóng khởi nghiệp mới đang cố gắng tạo ra các dịch vụ nhằm thay đổi cách khách hàng tương tác bằng các dịch vụ trợ lý ảo. Nhằm trợ giúp người dùng, khách hàng của mình có những trải nghiệm tốt nhất về sản phẩm và cách dịch vụ cung cấp. Nổi bật nhất trong đó phải kể đến các ứng dụng tích hợp trợ lý ảo như ***operator.com, x.ai, reply.ai***, các nền tảng dịch vụ như ***Chatfuel***. Gần đây nhất Microsoft đã tạo ra một framework cho phép các nhà phát triển tạo ra các chatbot trên nền tảng Web, hay Facebook cũng phát hành F8 SDK cho phép nhà phát triển tích hợp vào Messenger.

Ở trong nước, một số công ty như Quản lý Hồ sơ y tế điện tử ERM.,JSC và Vietcare đã phát triển tạo ra hệ thống trả lời tự động về kiến thức y khoa, hỏi đáp về sức khỏe thông tin y tế, hay RiveHub, Subiz, … cũng đang cố gắng tạo ra cho mình một hệ thống hỗ trợ, chăm sóc khách hàng và bán hàng tự động.

Rất nhiều công ty khác đang có hi vọng phát triển các trợ lý ảo có thể hiểu được ngôn ngữ tự nhiên của con người, có thể đối thoại và tương tác được với con người một cách tự nhiên. Nhiều người cho rằng việc sử dụng kỹ thuật xử lý ngôn ngữ tự nhiên NLP và các kỹ thuật học sâu Deep Learning để làm tăng được chất lượng và hiệu quả của hệ thống. Nhưng từ lý thuyết đến thực tế là cả một chặng đường dài, bằng cách nào đó, con người có thể tích hợp trí tuệ nhân tạo (AI) vào các sản phẩm công nghiệp của mình.

Như vậy, hệ thống trả lời tự động có những nhiệm vụ và vai trò quan trọng, có thể trợ giúp được con người rất nhiều trong rất nhiều lĩnh vực: y tế, giáo dục, thương mại điện tử, …, xứng đáng để nghiên cứu và đưa ra các sản phẩm phù hợp với thực tế. Với sự ra đời của framework sequence-to-sequence [7] gần đây, nhiều hệ thống huấn luyện đã sử dụng các mạng nơ-ron để sinh ra các câu trả lời mới khi đưa vào mạng một câu hỏi hoặc một thông điệp. Đây là một hướng tiếp cận mới có nhiều triển vọng trong việc xây dựng một hệ thống đối thoại. Qua đó, chúng tôi đã nghiên cứu dựa trên khung làm việc sequence-to-sequence, để xây dựng mô hình đối thoại cho tiếng Việt, từ có có thể áp dụng được vào các bài toán thực tế.

2. Mục tiêu của luận văn

Với cơ sở thực tiễn trên, luận văn này đặt ra mục tiêu nghiên cứu các mô hình có thể phát sinh văn bản sử dụng các mạng học sâu Deep Neural Networks, dựa trên khung làm việc sequence-to-sequence, 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 kho dữ liệu phụ đề nguồn mở OpenSubtitles 2016 [1].

3. Cấu trúc của luận văn

Để mô tả kết quả nghiên cứu, luận văn được chia thành 5 chương với các nội dụng như sau:

CHƯƠNG 1: Tổng quan về hệ thống trả lời tự động; Giới thiệu tổng quan về hệ thống đối thoại người máy, nghiên cứu tổng quan về tình hình nghiên cứu trong và ngoài nước, phân loại các mô hình trả lời tự động

CHƯƠNG 2: Cơ sở mạng nơ-ron nhân tạo; Nghiên cứu về cơ sở mạng nơ-ron nhân tạo, các mô hình mạng nơ-ron cải tiến là cơ sở của mạng học sâu.

CHƯƠNG 3: Mô hình đối thoại với mạng nơ-ron; Nghiên cứu các mô hình phát sinh văn bản trong hệ thống đối thoại, sử dụng mạng nơ-ron, giới thiệu về mô hình đối thoại seq2seq và các vấn đề chung gặp phải khi xây dựng mô hình đối thoại.

CHƯƠNG 4: Xây dựng mô hình đối thoại cho tiếng Việt; Áp dụng các kết quả nghiên cứu được, đề xuất xây dựng một mô hình đối thoại cho tiếng Việt. Liệt kê các vấn đề và các giải pháp khắc phục khi huấn luyện mô hình sử dụng mạng nơ-ron tái phát.

CHƯƠNG 5: Thực nghiệm và đánh giá mô hình; Thực nghiệm mô hình đã xây dựng được với bộ dữ liệu nguồn mở OpenSubtitles 2016. Trình bày các công cụ, thư viện mã nguồn mở hỗ trợ trong việc tiền xử lý dữ liệu, cũng như trong quá trình huấn luyện mô hình đối thoại tiếng Việt.

KẾT LUẬN: Phần này đưa ra các kết luận và đánh giá kết quả đạt được của luận văn

TÀI LIỆU THAM KHẢO: Đưa ra danh sách các bài báo được sử dụng làm tham khảo, tham chiếu cho luận văn.

CHƯƠNG 1: TỔNG QUAN VỀ HỆ THỐNG TRẢ LỜI TỰ ĐỘNG

Bài toán xây dựng hệ thống trả lời tự động là một bài toán khó thuộc lĩnh vực xử lý ngôn ngữ tự nhiên. Bởi vì tính nhập nhằng, đa nghĩa, đa ngữ cảnh của ngôn ngữ tự nhiên. Bài toán đặt ra nhiều thách thức để phát hiện ra được câu trả lời phù hợp nhất, thông tin hữu ích nhất. Chương này sẽ giới thiệu tổng quan về hệ thống đối thoại người máy, tìm hiểu các nghiên cứu ở trong và ngoài nước để thấy được tình hình nghiên cứu và các phương pháp tiếp cận của các nghiên cứu trước đây.

1.1 Hệ thống đối thoại người máy

Các hệ thống đối thoại người máy (Dialogue systems), còn được gọi là trợ lý tương tác hội thoại, trợ lý ảo và đôi khi được gọi với thuật ngữ là chatbot, được sử dụng rộng rãi trong các ứng dụng khác nhau, từ các dịch vụ kỹ thuật cho đến các công cụ có thể học ngôn ngữ và giải trí [17]. Các hệ thống đối thoại có thể được chia thành ***các hệ thống hướng mục tiêu*** trên một miền ứng dụng, ví dụ như các dịch vụ hỗ trợ kỹ thuật, và ***các hệ thống không có định hướng mục tiêu***, ví dụ như các công cụ học ngôn ngữ hoặc các nhân vật trò chơi máy tính [3]. Trong luận văn này, chúng tôi tập trung vào trường hợp thứ hai, là đi xây dựng một mô hình đối thoại cho tiếng Việt trên miền mở do có sẵn nguồn dữ liệu lớn từ các phụ đề Phim tiếng Việt được lấy trên OpenSubtitles năm 2016 [1].

Phương pháp hướng dữ liệu qui mô lớn, trong đó sử dụng dữ liệu đã được ghi tự động để truy vấn tri thức và phát sinh văn bản, đang trở nên ngày càng quan trọng trong lời nói và sự hiểu biết ngôn ngữ và phát sinh ngôn ngữ tự nhiên. Một loạt các phương pháp học máy hướng dữ liệu đã được chứng minh là có hiệu quả bằng việc xử lý ngôn ngữ tự nhiên, bao gồm các công việc liên quan đối thoại như đối thoại chính sách học tập [17], theo dõi trạng thái đối thoại [18,19] và phát sinh ngôn ngữ tự nhiên [20]…

Một trong những thách thức chính trong phát triển của hệ thống đối thoại người máy hướng nhiệm vụ, và trong việc mở rộng chúng trong nhiều miền ứng dụng, được nhắc đến trong [22], là sự sẵn có của dữ liệu trên một miền hội thoại cụ thể. Hệ thống đối thoại cần kết hợp và khai thác nhiều thành phần, ví dụ như

nhận dạng giọng nói, hiểu ngôn ngữ tự nhiên, giám sát hội thoại, phát sinh ngôn ngữ tự nhiên, và mỗi thành phần này yêu cầu sẵn có nguồn dữ liệu trên miền cụ thể, tài nguyên và các mô hình. Bao gồm các mô hình ngôn ngữ, mô hình ngữ âm, mô hình hiểu ngôn ngữ, các miền bản thể ontology, các kịch bản tương tác, các khuôn mẫu phát sinh ngôn ngữ, …

Mặc dù, nhiều vấn đề AI đã được hưởng lợi ích từ các nguồn dữ liệu ngày càng lớn, thu thập dữ liệu end-to-end cho các hệ thống đối thoại hướng nhiệm vụ vẫn còn là một vấn đề khó khăn. Phương pháp tiếp cận hiện tại để thu thập dữ liệu thoại dẫn đến chi phí phát triển cao và tiêu tốn thời gian cho các nhà phát triển hệ thống. Trừ khi các nguồn lực bên ngoài đã có sẵn (không phải trường hợp cho hầu hết các lĩnh vực), trong miền tập dữ liệu yêu cầu phải có một hệ thống triển khai có khả năng duy trì một cuộc đối thoại với người dùng. Điều này dẫn đến một vấn đề khởi động: do thiếu dữ liệu để huấn luyện hệ thống ban đầu, các nhà phát triển hệ thống mang gánh nặng về việc phát triển văn phạm và các mô hình ngôn ngữ, hoặc là thủ công hoặc với các nghiên cứu Wizard-of-Oz [22]. Thu thập dữ liệu hội thoại với phiên bản đầu tiên của một hệ thống được triển khai có thiếu sót: chất lượng dữ liệu thu thập có thể phải chịu những bất cập của hệ thống chính nó, và người dùng có thể chịu ảnh hưởng ngôn ngữ của chúng để điều chỉnh cho những khuyết điểm của hệ thống trong việc theo hết một cuộc đối thoại. Kết quả là, tốc độ của tập dữ liệu có thể chậm hơn so với mong muốn. Cuối cùng, quá trình phát triển tốn kém này phải được lặp đi lặp lại trên một lần nữa cho mỗi miền hoặc hệ thống mới, hoặc ngay cả khi chức năng mới được thêm vào.

1.2 Tình hình nghiên cứu trong và ngoài nước

Hệ thống trả lời tự động đã được các nhà nghiên cứu quan tâm từ rất lâu rồi, bao gồm các trường đại học, các viện nghiên cứu và các doanh nghiệp. Việc nghiên cứu về hệ thống trả lời tự động có ý nghĩa trong khoa học và thực tế. Đã có rất nhiều các hội nghị thường niên về xử lý ngôn ngữ tự nhiên, khai phá dữ liệu, xử lý dữ liệu lớn, tương tác người máy, … như TREC, CLEF, tại Việt Nam có KSE, RIVF, ATC, …

Theo ý tưởng của Russel và cộng sự [21], thì một hệ thống AI phải được **kiểm tra** (hành động dưới sự ràng buộc hình thức và phù hợp với các điều kiện kỹ thuật); phải được **xác nhận** (không theo đuổi các hành vi không mong muốn

dưới sự ràng buộc trước); phải **an toàn** (ngăn chặn các thao tác có chủ ý của các bên thứ ba, hoặc bên ngoài hoặc bên trong); và phải **được kiểm soát** (con người cần phải có cách để thiết lập lại kiểm soát nếu cần thiết).

Việc thiết kế hệ thống đối thoại là một nhiệm vụ đầy thách thức và là một trong những mục tiêu ban đầu của trí tuệ nhân tạo (Turing, 1950) [24]. Trong nhiều thập kỷ, việc thiết kế tác nhân đối thoại đã giúp các hệ thống dựa trên cơ sở tri thức và cơ chế dựa trên luật Rule-based để hiểu các thông điệp đầu vào của con người và tạo ra các phản hồi đáp ứng hợp lý [25,26,27]. Phương pháp tiếp cận hướng dữ liệu nhấn mạnh vào việc học trực tiếp từ các tập ngữ liệu của các cuộc đối thoại tiếng nói hoặc văn bản chữ viết. Gần đây, phương pháp này đã đạt được đà vì lợi thế dữ liệu phong phú [3], tăng sức mạnh tính toán, và các thuật toán học tốt hơn mà tự động hóa quá trình tính năng kỹ thuật [28,29].

Ritter và cộng sự (2010) [30] đã đề xuất phương pháp tiếp cận hướng dữ liệu cho việc xây dựng hệ thống đối thoại, và họ đã trích xuất ra 1,3 triệu cuộc hội thoại từ Twitter với mục đích là phát hiện ra các hành động trong cuộc hội thoại. Bằng việc xây dựng dựa trên sự tương đồng về phân phối trong khuôn khổ mô hình không gian vector, Banchs và Li (2012) [31] đã xây dựng một công cụ tìm kiếm để lấy câu trả lời thích hợp cho bất kỳ một thông điệp đầu vào. Phương pháp tiếp cận khác tập trung vào nhiệm vụ trên một lĩnh vực cụ thể như các trò chơi [32], và các nhà hàng ăn uống (2016) [33,34].

Cá nhân hóa hệ thống đối thoại đòi hỏi phải đầy đủ thông tin, từ mỗi người dùng và số lượng mẫu đủ lớn để xác định được khoảng không gian. Các phong cách Viết định lượng bằng độ dài từ, độ mạnh của động từ, tính phân cực, và phân phối các hành vi đối thoại đã được sử dụng để mô hình hóa người dùng, bởi Walker 2012, [35]. Những nỗ lực khác tập trung vào việc xây dựng một hồ sơ người dùng dựa trên nhân khẩu học, chẳng hạn như: giới tính, thu nhập, tuổi tác và tình trạng hôn nhân, bởi Bonin 2014, [36].

Với sự ra đời của framework sequence-to-sequence [7], nhiều hệ thống huấn luyện gần đây đã sử dụng các mạng nơ-ron tái phát (RNN) để sinh ra các câu trả lời mới khi đưa vào mạng một câu hỏi hoặc một thông điệp. Ví dụ, Lê Viết Quốc và Vinyals [6] đã đề xuất sử dụng tập dữ liệu là lịch sử hỗ trợ kỹ thuật IT-help desk để huấn luyện mạng LSTM để sinh ra câu trả lời mới. Sordoni và cộng sự (2015) [9] đã xây dựng các cuộc đối thoại Twitter giới hạn bối cảnh lịch sử đến

một thông điệp. Với sự giúp đỡ của các mô hình ngôn ngữ được tiền huấn luyện, chúng mã hóa mỗi tin nhắn vào một vector đại diện. Để loại bỏ sự cần thiết cho một mô hình ngôn ngữ, Serban và cộng sự (2015) [3] đã thử huấn luyện end-to-end trên một mạng RNN. Họ cũng bắt đầu hệ thống của mình với các word embeddings đã được huấn luyện từ trước.

Trong khi các hệ thống này có thể sản xuất ra các câu trả lời mới lạ, rất khó để hiểu được bao nhiêu khả năng được sử dụng bởi các mô hình ngôn ngữ tự nhiên so với việc mô hình hóa hội thoại đối thoại liền nhau. Thông thường các phản ứng châu về các câu thường xuyên nhất được quan sát trong tập ngữ liệu được huấn luyện [37].

Sự phớt lờ, hay sự lúng túng BLEU và deltaBLEU được chuyển thể từ mô hình ngôn ngữ và dịch máy, đã được sử dụng để đánh giá phản ứng mới được tạo ra, bởi Yao 2015 [15], Sordoni 2015 [9], Galley 2015 [38]. Những số liệu chỉ đo mức độ lưu loát từ vựng của phản ứng mới và không bắt phạt đối với các ứng viên không mạch lạc liên quan đến ngôn từ đàm thoại. Trong khi việc tìm kiếm các chỉ số tốt hơn thì vẫn còn tiếp tục, tự động đánh giá của các đáp án được sinh ra vẫn là một vấn đề mở, theo Shang, 2015 [39].

Việc xây dựng các chương trình trả lời tự động chatbots và conversational agents đã được theo đuổi bởi nhiều nhà nghiên cứu trong nhiều thập kỷ qua. Tuy nhiên, hầu hết các hệ thống hội thoại này đòi hỏi một quy trình xử lý khá phức tạp qua nhiều giai đoạn [49, 50]. Hướng tiếp cận của chúng tôi khác với các hệ thống thông thường bằng cách áp dụng mô hình sequence-to-sequence [7] để xây dựng một mô hình end-to-end cho vấn đề thiếu kiến thức miền. Về nguyên tắc, nó có thể kết hợp với các hệ thống khác để ghi nhận một danh sách các đáp án ứng viên, nhưng mô hình của chúng tôi dựa trên việc sản sinh câu trả lời được đưa ra bởi một mô hình xác suất huấn luyện để cực đại hóa xác suất của câu trả lời trong một số ngữ cảnh. Đây là một hướng tiếp cận mới có nhiều triển vọng trong việc xây dựng một hệ thống đối thoại.

1.3 Phân loại các mô hình trả lời tự động

Mô hình trả lời tự động dựa vào một số kỹ thuật và các tiêu chí khác nhau, có thể được phân loại như: Phân loại theo miền ứng dụng; Phân loại theo khả năng

trả lời mẫu hỏi; Phân loại theo mức độ dài, ngắn của đoạn đối thoại; Phân loại theo hướng tiếp cận

1.3.1 Phân loại theo miền ứng dụng

Miền mở (Open Domain): Mô hình trả lời tự động trên miền mở cho phép người dùng có thể tham gia trò chuyện với một chủ đề bất kỳ, không nhất thiết phải có một mục tiêu rõ ràng hay một ý định cụ thể nào. Các cuộc trò chuyện trên mạng xã hội như Facebook, Twitter và Reddit thường là miền mở, chúng có thể đi vào tất cả các chủ đề. Số lượng các chủ đề thảo luận được đề cập đến là không giới hạn, do đó, tri thức yêu cầu được tạo ra để trả lời các câu đối thoại thuộc miền mở trở nên khó hơn. Tuy nhiên, việc thu thập trích rút dữ liệu từ miền này khá phong phú và đơn giản.

Miền đóng (Close Domain): Mô hình trả lời tự động thuộc miền đóng thường tập trung vào trả lời các câu hỏi đối thoại liên quan đến một miền cụ thể, ví dụ như: Y tế, Giáo dục, Du lịch, Mua sắm, ...

Trong một miền đóng cụ thể, không gian các mẫu hỏi input và output là có giới hạn, bởi vì các hệ thống này đang cố gắng để đạt được một mục tiêu rất cụ thể. Hệ thống hỗ trợ kỹ thuật (Technical Customer Support) hay Tư vấn và hỗ trợ mua hàng (Shopping Assistants) là các ứng dụng thuộc miền đóng. Các hệ thống này không thể đối thoại về "Chính trị" hay "Pháp luật", chúng chỉ cần thực hiện các nhiệm vụ cụ thể một cách hiệu quả nhất có thể. Chắc chắn, người dùng vẫn có thể hỏi đáp bất cứ gì, nhưng hệ thống không yêu cầu phải xử lý những trường hợp ngoại lệ này.

1.3.2 Phân loại theo khả năng trả lời mẫu hỏi

Các hệ thống có khả năng trả lời các mẫu hỏi liên quan đến sự vật, hiện tượng, không gian, thời gian, … Câu trả lời là các từ khóa, chuỗi ký tự trong một tài liệu văn bản hoặc cơ sở dữ liệu tri thức.

- **Mô hình có cơ chế lập luận đơn giản**: Trích xuất các câu trả lời có sẵn trong tập tài liệu sau đó sử dụng các suy luận để tìm mối liên kết giữa câu trả lời và câu hỏi. Hệ thống sử dụng các nguồn tri thức như ontology về từng miền cụ thể và cơ sở tri thức chung.

- **Mô hình có khả năng tổng hợp**: Câu trả lời được trích rút từ nhiều mẫu tài liệu. Mẫu hỏi thường là về danh sách, cách thức, nguyên nhân, kết quả, …
- **Mô hình có khả năng lập luận tương tự**: Mẫu hỏi có tính chất suy đoán, câu trả lời ẩn trong tập tài liệu. Mô hình trả lời cần trích xuất được các luận chứng và sử dụng lập luận tương tự để tìm ra câu trả lời.

1.3.3 Phân loại theo mức độ dài, ngắn của đoạn đối thoại

Trả lời một mẫu hỏi càng dài thì càng khó để tự động hóa nó. Các đoạn đối thoại văn bản ngắn (**Short-text Conversations**) thì dễ hơn, trong đó, mục tiêu là tạo ra một câu trả lời đơn cho một mẫu hỏi đơn đầu vào. Ví dụ, bạn có thể nhận một mẫu hỏi cụ thể từ một người dùng và có thể đáp lại một câu trả lời thích hợp.

Ví dụ:

Q: *Vậy anh hứng_thú với cái gì ?*

A: *Những di_vật của chiến_tranh.*

Trong khi đó, các đoạn đối thoại dài (**Long-text Conversations**) cần đi qua nhiều điểm ngắt và cần giữ được trạng thái những gì đã được nói ra. Các đoạn đối thoại hỗ trợ khách hàng thường là các luồng hội thoại dài với nhiều câu hỏi, nhiều ý hỏi được nhắc đến.

Ví dụ:

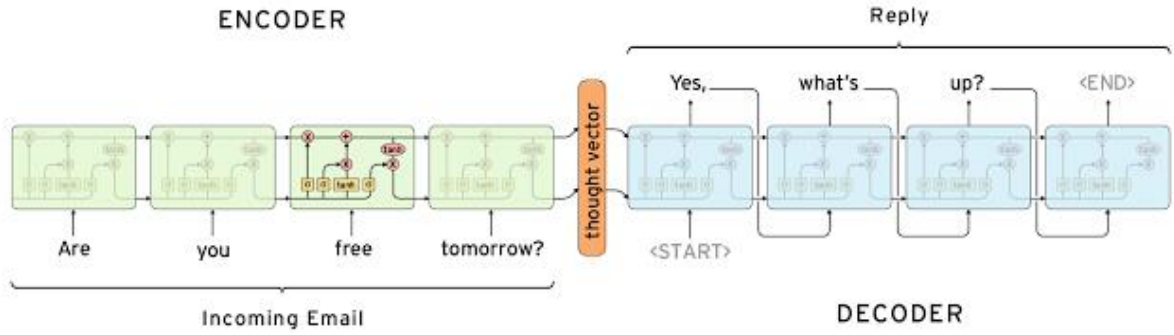
Q: *Thưa Thầy/ Cô ! Cho em hỏi : Người mất năng lực hành vi dân sự một phần mà gây thiệt hại về vật chất hoặc tinh thần cho người khác, thì có phải chịu trách nhiệm bồi thường thiệt hại do mình gây ra hay không ?*

A: *Chào anh! Trong BLDS chỉ có quy định về người bị hạn chế năng lực hành vi dân sự mà không có người mất năng lực hành vi dân sự một phần. Người bị hạn chế năng lực hành vi dân sự là người đã trưởng thành nên nếu gây thiệt hại thì bản thân họ phải có trách nhiệm bồi thường.*

1.3.4 Phân loại theo hướng tiếp cận

Tiếp cận dựa vào trích chọn thông tin (**Retrieval-based**): Các kỹ thuật thường sử dụng một kho đã định nghĩa trước các câu trả lời kết hợp với một vài phương pháp trích chọn Heuristic để nhặt ra một đáp án thích hợp nhất dự vào mẫu hỏi input và ngữ cảnh. Kỹ thuật heuristic sử dụng ở đây đơn giản có thể là sự so khớp các biểu thức dựa vào luật (rule-based), hoặc phức tạp như việc kết hợp học máy (Machine Learning) để phân lớp các câu hỏi và đáp án trả về. Những hệ thống kiểu này không sinh ra văn bản mới, chúng chỉ nhặt một đáp án từ một tập dữ liệu cố định sẵn có.

Tiếp cận dựa vào mô hình có thể sinh sản (**Generative-based**): Mô hình này không dựa trên tập trả lời định nghĩa trước. Chúng có khả năng tự sản sinh các đáp án từ đầu. Các mô hình có thể sinh sản thường dựa vào các kỹ thuật Máy Dịch (Machine Translation), nhưng thay vì dịch từ ngôn ngữ này sang ngôn ngữ khác, thì nó có thể "dịch" từ một **input** sang một **output**.



Hình 1.1: Mô hình chuỗi có thể sinh sản

Cả hai hướng tiếp cận này đều có những thuận lợi và khó khăn. Nhờ vào kho dữ liệu với các bộ luật được thiết kế bằng tay, mô hình dựa trên trích chọn thông tin (retrieval-based) không mắc phải các lỗi về ngữ pháp. Tuy nhiên, chúng không thể xử lý được các trường hợp các mẫu chưa được quan sát, không có trong bộ luật. Vì những lý do đó, các mô hình này không thể nhớ được các thông tin ngữ cảnh trước đó như "tên người" được đề cập trong đoạn hội thoại.

Mô hình có thể sinh sản thì "*thông minh hơn*". Chúng có thể nhớ lại được các thực thể được nhắc đến trong mẫu hỏi và tạo ra cảm giác bạn đang nói chuyện với con người. Tuy nhiên, những mô hình này thì rất khó để huấn luyện, rất có

thể bị mắc lỗi về ngữ pháp (đặc biệt trên các câu dài) và mô hình yêu cầu một lượng rất lớn dữ liệu để huấn luyện.

Các kỹ thuật học sâu Deep Learning có thể được sử dụng cho cả hai mô hình Retrieval-based hoặc Generative-based, nhưng các nhà nghiên cứu thường tập trung hướng vào mô hình Generative. Tuy nhiên, chúng ta vẫn đang ở giai đoạn đầu của việc tiếp cận với mô hình có thể sinh sản và có kết quả khả quan. Song thời điểm hiện tại, các hệ thống thương mại vẫn phù hợp với các mô hình Retrieval-based.

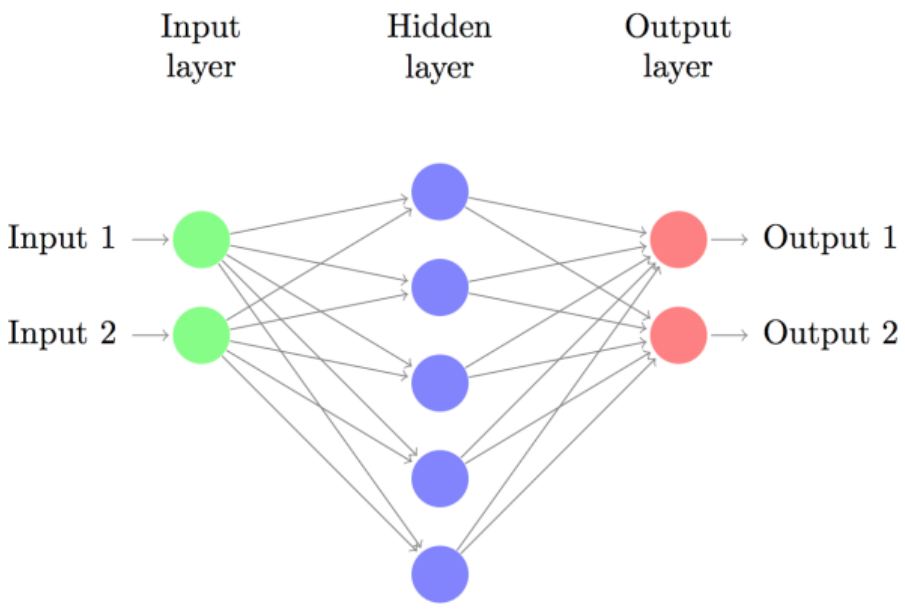
CHƯƠNG 2: CƠ SỞ MẠNG NƠ RON NHÂN TẠO

Chương này giới thiệu về cơ sở lý thuyết về mạng nơ rơn nhân tạo, cách thức hoạt động của mạng nơ-ron, các phiên bản mở rộng của mạng nơ-ron nhân tạo, như: Mạng nơ-ron tái phát (RNN), mạng cải tiến LSTM. Mạng nơ-ron tái phát **RNN** là một trong những mô hình Deep learning được đánh giá có nhiều ưu điểm trong các tác vụ xử lý ngôn ngữ tự nhiên. Đây cũng là cơ sở thực hiện để xây dựng mô hình đối thoại trong đề tài luận văn.

2.1 Kiến trúc mạng nơ ron nhân tạo

Mạng nơ ron nhân tạo (Artificial Neural Network – ANN) là một mô hình xử lý thông tin được mô phỏng dựa trên hoạt động của hệ thống thần kinh của sinh vật, bao gồm số lượng lớn các Nơ-ron được gắn kết để xử lý thông tin. ANN hoạt động giống như bộ não của con người, được học bởi kinh nghiệm (thông qua việc huấn luyện), có khả năng lưu giữ các tri thức và sử dụng các tri thức đó trong việc dự đoán các dữ liệu chưa biết (unseen data).

Một mạng nơ-ron là một nhóm các nút nối với nhau, mô phỏng mạng nơ-ron thần kinh của não người. Mạng nơ ron nhân tạo được thể hiện thông qua ba thành phần cơ bản: mô hình của nơ ron, cấu trúc và sự liên kết giữa các nơ ron. Trong nhiều trường hợp, mạng nơ ron nhân tạo là một hệ thống thích ứng, tự thay đổi cấu trúc của mình dựa trên các thông tin bên ngoài hay bên trong chạy qua mạng trong quá trình học.



Hình 2.1: Kiến trúc mạng nơ-ron nhân tạo

Kiến trúc chung của một ANN gồm 3 thành phần đó là **Input Layer**, **Hidden Layer** và **Output Layer** (Hình 2.1)

Trong đó, lớp ẩn (**Hidden Layer**) gồm các nơ-ron, nhận dữ liệu input từ các Nơ-ron ở lớp (Layer) trước đó và chuyển đổi các input này cho các lớp xử lý tiếp theo. Trong một mạng ANN có thể có nhiều Hidden Layer.

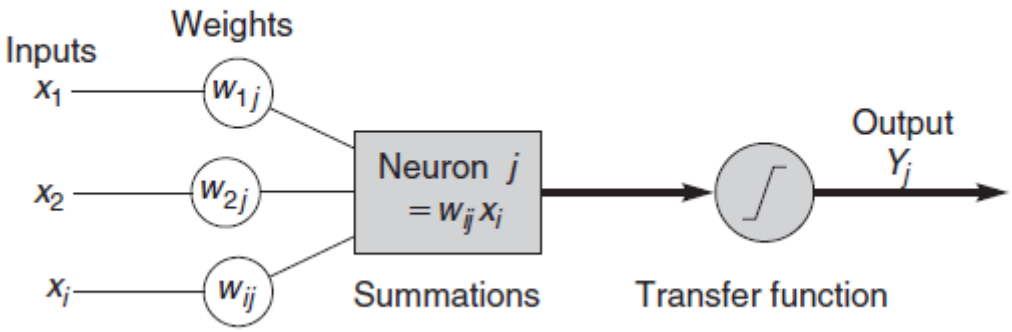
Lợi thế lớn nhất của các mạng ANN là khả năng được sử dụng như một cơ chế xấp xỉ hàm tùy ý mà "học" được từ các dữ liệu quan sát. Tuy nhiên, sử dụng chúng không đơn giản như vậy, một số các đặc tính và kinh nghiệm khi thiết kế một mạng nơ-ron ANN.

- **Chọn mô hình**: Điều này phụ thuộc vào cách trình bày dữ liệu và các ứng dụng. Mô hình quá phức tạp có xu hướng dẫn đền những thách thức trong quá trình học.
- **Cấu trúc và sự liên kết giữa các nơ-ron**
- **Thuật toán học**: Có hai vấn đề cần học đối với mỗi mạng ANN, đó là học tham số của mô hình (parameter learning) và học cấu trúc (structure learning). Học tham số là thay đổi trọng số của các liên kết giữa các nơ-ron trong một mạng, còn học cấu trúc là việc điều chỉnh cấu trúc mạng

bằng việc thay đổi số lớp ẩn, số nơ-ron mỗi lớp và cách liên kết giữa chúng. Hai vấn đề này có thể được thực hiện đồng thời hoặc tách biệt.

Nếu các mô hình, hàm chi phí và thuật toán học được lựa chọn một cách thích hợp, thì mạng ANN sẽ cho kết quả có thể vô cùng mạnh mẽ và hiệu quả.

2.2 Hoạt động của mạng nơ-ron nhân tạo



Hình 2.2: Quá trình xử lý thông tin của một mạng nơ-ron nhân tạo.

Inputs: Mỗi Input tương ứng với 1 đặc trưng của dữ liệu. Ví dụ như trong ứng dụng của ngân hàng xem xét có chấp nhận cho khách hàng vay tiền hay không thì mỗi input là một thuộc tính của khách hàng như thu nhập, nghề nghiệp, tuổi, số con,…

Output: Kết quả của một ANN là một giải pháp cho một vấn đề, ví dụ như với bài toán xem xét chấp nhận cho khách hàng vay tiền hay không thì output là yes hoặc no.

Connection Weights (Trọng số liên kết) : Đây là thành phần rất quan trọng của một ANN, nó thể hiện mức độ quan trọng, độ mạnh của dữ liệu đầu vào đối với quá trình xử lý thông tin chuyển đổi dữ liệu từ Layer này sang layer khác. Quá trình học của ANN thực ra là quá trình điều chỉnh các trọng số Weight của các dữ liệu đầu vào để có được kết quả mong muốn.

Summation Function (Hàm tổng): Tính tổng trọng số của tất cả các input được đưa vào mỗi Nơ-ron. Hàm tổng của một Nơ-ron đối với n input được tính theo công thức sau:

$$Y = \sum_{i=1}^{n} X_i W_i$$

Transfer Function (Hàm chuyển đổi): Hàm tổng của một nơ-ron cho biết khả năng kích hoạt của nơ-ron đó còn gọi là kích hoạt bên trong. Các nơ-ron này có thể sinh ra một output hoặc không trong mạng ANN, nói cách khác rằng có thể output của 1 Nơ-ron có thể được chuyển đến layer tiếp trong mạng Nơ-ron theo hoặc không. Mối quan hệ giữa hàm tổng và kết quả output được thể hiện bằng hàm chuyển đổi.

Việc lựa chọn **hàm chuyển đổi** có tác động lớn đến kết quả đầu ra của mạng ANN. Hàm chuyển đổi phi tuyến được sử dụng phổ biến trong mạng ANN là hoặc *sigmoid* hoặc *tanh*.

$$f(z) = \frac{1}{1 + \exp(-z)}.$$

$$f(z) = \tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}},$$

Trong đó, hàm *tanh* là phiên bản thay đổi tỉ lệ của *sigmoid* , tức là khoảng giá trị đầu ra của hàm chuyển đổi thuộc khoảng [-1, 1] thay vì [0,1] nên chúng còn gọi là hàm chuẩn hóa (Normalized Function).

Kết quả xử lý tại các nơ-ron (Output) đôi khi rất lớn, vì vậy hàm chuyển đổi được sử dụng để xử lý output này trước khi chuyển đến layer tiếp theo. Đôi khi thay vì sử dụng Transfer Function người ta sử dụng giá trị ngưỡng (Threshold value) để kiểm soát các output của các neuron tại một layer nào đó trước khi chuyển các output này đến các Layer tiếp theo. Nếu output của một neuron nào đó nhỏ hơn Threshold thì nó sẻ không được chuyển đến Layer tiếp theo.

Mạng nơ-ron của chúng ta dự đoán dựa trên lan truyền thẳng (forward propagation) là các phép nhân ma trận cùng với activation function để thu được

kết quả đầu ra. Nếu input x là vector 2 chiều thì ta có thể tính kết quả dự đoán $\hat{y}$ bằng công thức sau

$$z_1 = xW_1 + b_1$$
$$a_1 = \tanh(z_1)$$
$$z_2 = a_1W_2 + b_2$$
$$a_2 = \hat{y} = softmax(z_2)$$

Trong đó, z_i là input của layer thứ i, a_i là output của layer thứ i sau khi áp dụng activation function. W_1, b_1, W_2, b_2 là các thông số (parameters) cần tìm của mô hình mạng nơ-ron.

Huấn luyện để tìm các thông số cho mô hình tương đương với việc tìm các thông số W_1, b_1, W_2, b_2, sao cho độ lỗi của mô hình đạt được là thấp nhất. Ta gọi hàm độ lỗi của mô hình là *loss function*. Đối với *softmax function*, ta dùng **cross-entropy loss** (còn gọi là negative log likelihood).

Nếu ta có N dòng dữ liệu huấn luyện, và C nhóm phân lớp (trường hợp này là hai lớp nam, nữ), khi đó loss function giữa giá trị dự đoán $\hat{y}$ và y được tính như sau

$$L(y, \hat{y}) = -\frac{1}{N}\sum_{n\in N}\sum_{i\in C} y_{n,i} log\hat{y}_{n,i}$$

Ý nghĩa công thức trên nghĩa là: lấy tổng trên toàn bộ tập huấn luyện và cộng dồn vào hàm loss nếu kết quả phân lớp sai. Độ dị biệt giữa hai giá trị $\hat{y}$ và y càng lớn thì độ lỗi càng cao. Mục tiêu của chúng ta là tối thiểu hóa hàm lỗi này. Ta có thể sử dụng phương pháp gradient descent để tối tiểu hóa hàm lỗi. Có hai loại gradient descent, một loại với fixed learning rate được gọi là batch gradient descent, loại còn lại có learning rate thay đổi theo quá trình huấn luyện được gọi là SGD (stochastic gradient descent) hay minibatch gradient descent.

Gradient descent cần các gradient là các vector có được bằng cách lấy đạo hàm của loss function theo từng thông số $\frac{\partial L}{\partial W_1}$, $\frac{\partial L}{\partial b_1}$, $\frac{\partial L}{\partial W_2}$, $\frac{\partial L}{\partial b_2}$. Để tính các gradient này, ta sử dụng thuật toán *backpropagation (lan truyền ngược)*. Đây là cách hiệu quả để tính gradient khởi điểm từ output layer.

Áp dụng backpropagation ta có các đại lượng

$$\delta_3 = y - \hat{y}$$
$$\delta_2 = (1 - tanh^2 z_1) \circ \delta_3 W_2^T$$
$$\frac{\partial L}{\partial W_2} = a_1^T \delta_3$$
$$\frac{\partial L}{\partial b_2} = \delta_3$$
$$\frac{\partial L}{\partial W_1} = x^T \delta_2$$
$$\frac{\partial L}{\partial b_1} = \delta_2$$

2.3 Mạng nơ-ron tái phát và ứng dụng

Mạng nơ-ron tái phát **Recurrent Neural Network** (RNN) là một trong những mô hình Deep learning được đánh giá có nhiều ưu điểm trong các tác vụ xử lý ngôn ngữ tự nhiên (NLP). Trong phần này, tôi sẽ trình bày các khái niệm, các đặc điểm cũng như những ứng dụng của RNNs trong các bài toán thực tế.

2.3.1 Mạng nơ-ron tái phát

Ý tưởng của RNN đó là thiết kế một Neural Network sao cho có khả năng xử lý được thông tin dạng chuỗi *(sequential information)*, ví dụ một câu là một chuỗi gồm nhiều từ.

Recurrent có nghĩa là thực hiện lặp lại cùng một tác vụ cho mỗi thành phần trong chuỗi. Trong đó, kết quả đầu ra tại thời điểm hiện tại phụ thuộc vào kết quả tính toán của các thành phần ở những thời điểm trước đó.

Nói cách khác, RNN là một mô hình có trí nhớ (*memory*), có khả năng nhớ được thông tin đã tính toán trước đó. Không như các mô hình Neural Network truyền thống đó là thông tin đầu vào (input) hoàn toàn độc lập với thông tin đầu ra (output). Về lý thuyết, RNN có thể nhớ được thông tin của chuỗi có chiều dài bất kì, nhưng trong thực tế mô hình này chỉ nhớ được thông tin ở vài bước trước đó.

2.3.2 Các ứng dụng của RNN

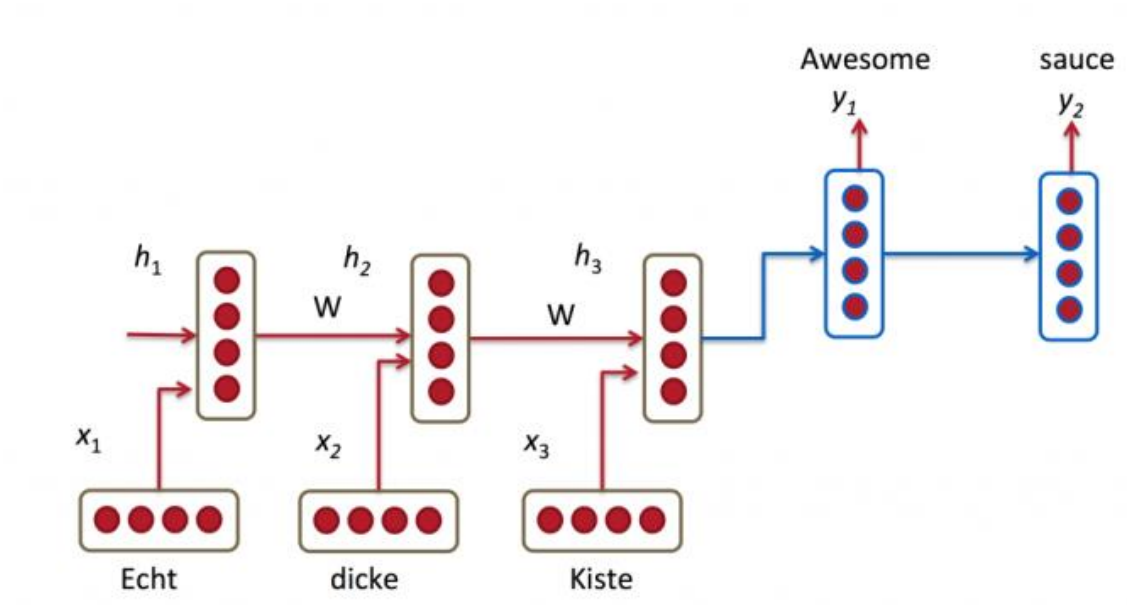
- **Mô hình ngôn ngữ và phát sinh văn bản**

Mô hình ngôn ngữ cho ta biết xác suất của một câu trong một ngôn ngữ là bao nhiêu (ví dụ xác suất ***p("hôm qua là thứ năm") = 0.001; p("năm thứ hôm là qua") = 0***). Đây cũng là bài toán dự đoán xác suất từ tiếp theo của một câu cho trước là bao nhiêu.

Từ bài toán này, chúng ta có thể mở rộng thành bài toán phát sinh văn bản (generating text/generative model). Mô hình này cho phép ta phát sinh ra văn bản mới dựa vào tập dữ liệu huấn luyện. Ví dụ, khi huấn luyện mô hình này bằng các văn bản truyện Kiều, ta có thể phát sinh được các đoạn văn tựa truyện Kiều. Tuỳ theo loại dữ liệu huấn luyện, ta sẽ có nhiều loại ứng dụng khác nhau.

- **Dịch máy**

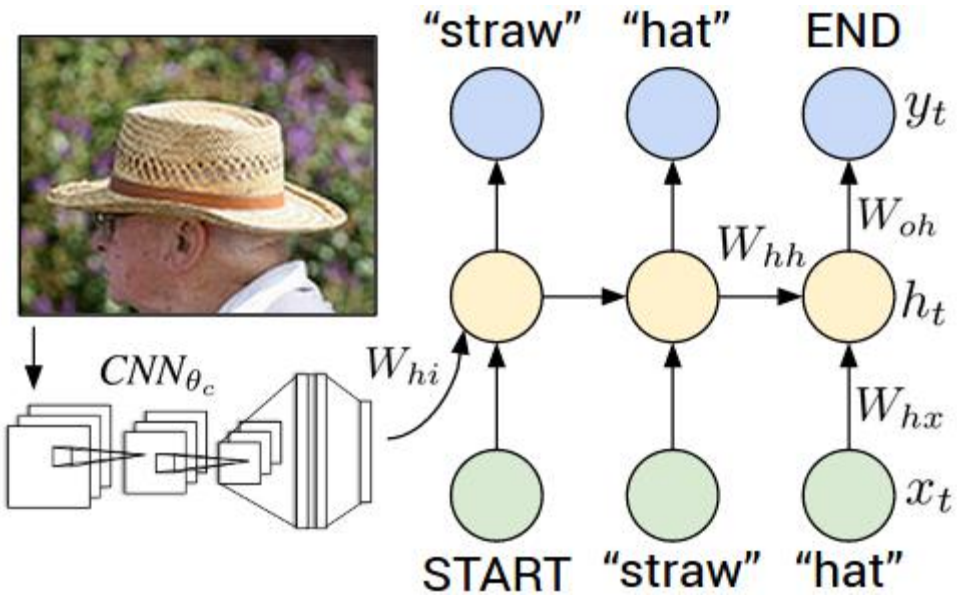
Bài toán dịch máy (Machine Translation) [5, 7] tương tự như mô hình ngôn ngữ. Trong đó, input là chuỗi các từ của ngôn ngữ nguồn (ví dụ tiếng Đức), output là chuỗi các từ của ngôn ngữ đích (ví dụ tiếng Anh). Điểm khác biệt ở đây đó là output chỉ có thể dự đoán được khi input đã hoàn toàn được phân tích. Điều này là do từ được dịch ra phải có đầy đủ thông tin của các từ trước đó.



Hình 2.3: Ứng dụng RNN trong máy dịch.

- **Phát sinh mô tả cho ảnh (Generating Image Descriptions)**

RNN kết hợp với Convolution Neural Networks [48] có thể phát sinh ra được các đoạn mô tả cho ảnh. Mô hình này hoạt động bằng cách tạo ra những câu mô tả từ các đặc trưng rút trích được trong bức ảnh.



Hình 2.4: Ứng dụng RNN phát sinh mô tả cho ảnh.

2.3.3 Huấn luyện mạng

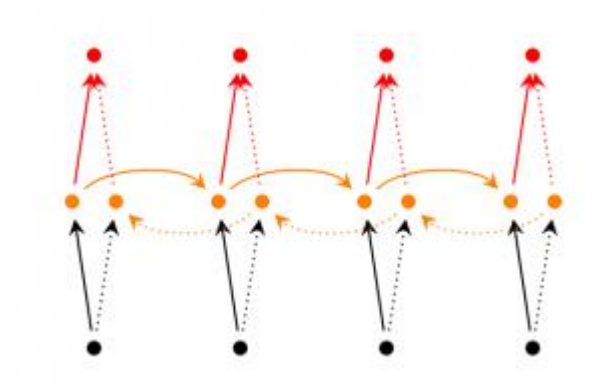
Huấn luyện RNN tương tự như huấn luyện Neural Network truyền thống. Chúng ta cũng sử dụng đến thuật toán backpropagation (lan truyền ngược) nhưng có một chút tinh chỉnh. Gradient tại mỗi output không chỉ phụ thuộc vào kết quả tính toán của bước hiện tại mà còn phụ thuộc vào kết quả tính toán của các bước trước đó.

Ví dụ, để tính gradient tại thời điểm $t = 4$, ta cần backpropagation 3 bước trước đó và cộng dồn các gradient này lại với nhau. Kĩ thuật này gọi là Backpropagation Through Time (*BPPTT*) [42]. Điểm hạn chế ở đây đó là hidden layer không có trí nhớ dài hạn. Vấn đề này còn gọi là *vanishing/exploding gradient problem* và như vậy, LSTM được sinh ra để giải quyết vấn đề này.

2.3.4 Các phiên bản mở rộng của RNN

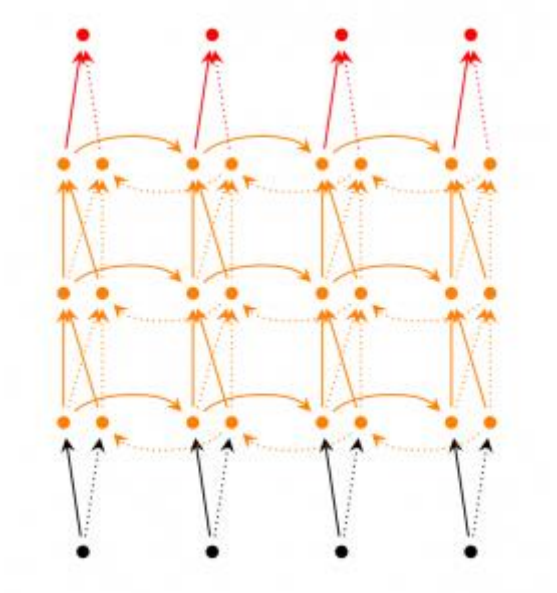
Trong vài năm qua, các nhà nghiên cứu đã phát triển nhiều loại mạng RNNs ngày càng tinh vi để giải quyết các mặt hạn chế của RNN. Dưới đây, là một số phiên bản mở rộng của RNN.

Bidirectional RNN (RNN hai chiều): dựa trên ý tưởng output tại thời điểm t không chỉ phụ thuộc vào các thành phần trước đó mà còn phụ thuộc vào các thành phần trong tương lai. Ví dụ, để dự đoán một từ bị thiếu (missing word) trong chuỗi, ta cần quan sát các từ bên trái và bên phải xung quanh từ đó. Mô hình này chỉ gồm hai RNNs nạp chồng lên nhau. Trong đó, các hidden state được tính toán dựa trên cả hai thành phần bên trái và bên phải của mạng.



Hình 2.5: Mạng RNN hai chiều.

Deep (Bidirectional) RNN: tương tự như Bidirectional RNN, điểm khác biệt đó là mô hình này gồm nhiều tầng Bidirectional RNN tại mỗi thời điểm. Mô hình này sẽ cho ta khả năng thực hiện các tính toán nâng cao nhưng đòi hỏi tập huấn luyện của chúng ta phải đủ lớn.



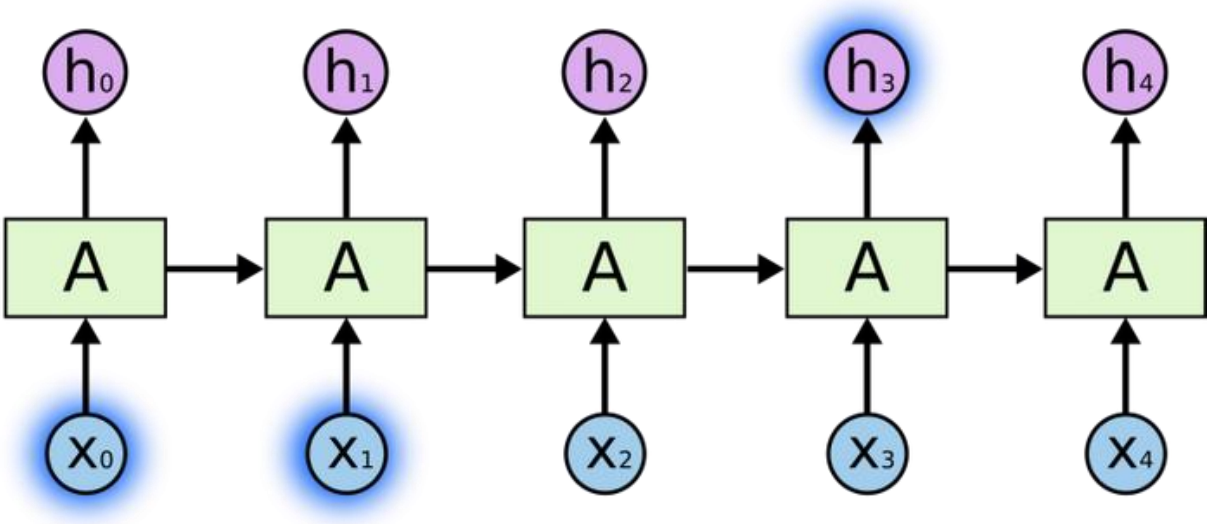
Hình 2.6: Mạng RNN nhiều tầng.

Long short-term memory networks (LSTM): mô hình này có cấu trúc tương tự như RNNs nhưng có cách tính toán khác đối với các trạng thái ẩn. Memory trong LSTMs được gọi là cells (hạt nhân). Ta có thể xem đây là một hộp đen nhận thông tin đầu vào gồm hidden state s_{t-1} và giá trị x_t. Bên trong các hạt nhân này, chúng sẽ quyết định thông tin nào cần lưu lại và thông tin nào cần xóa đi, nhờ vậy mà mô hình này có thể lưu trữ được thông tin dài hạn. Chi tiết mô hình mạng này được giới thiệu trong mục 2.4.

2.4 Mạng Long Short Term Memory

2.4.1 Vấn đề phụ thuộc quá dài

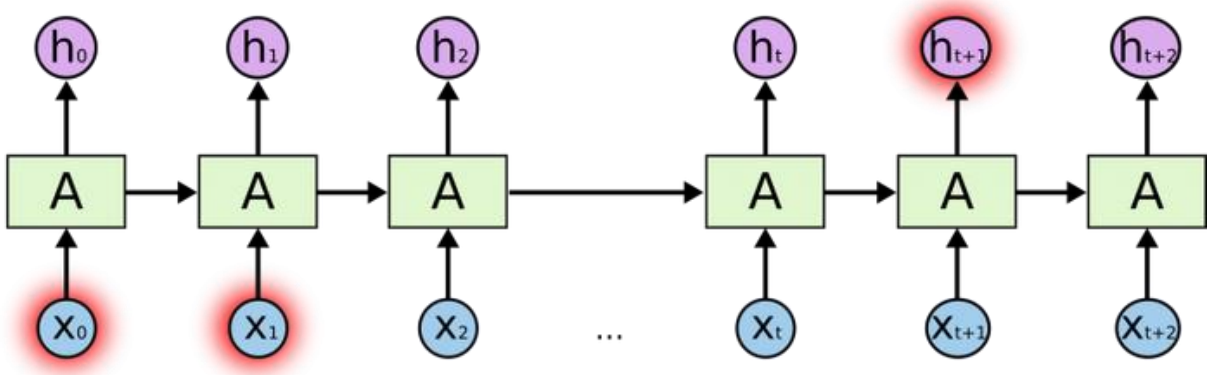
Ý tưởng ban đầu của RNN là kết nối những thông tin trước đó nhằm hỗ trợ cho các xử lý hiện tại. Nhưng đôi khi, chỉ cần dựa vào một số thông tin gần nhất để thực hiện tác vụ hiện tại. Ví dụ, trong mô hình hóa ngôn ngữ, chúng ta cố gắng dự đoán từ tiếp theo dựa vào các từ trước đó. Nếu chúng ta dự đoán 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, nạng RNN có thể học và sử dụng các thông tin quá khứ.



Hình 2.7: RNN phụ thuộc short-term.

Nhưng cũng có trường hợp chúng ta cần nhiều thông tin hơn, nghĩa là phụ thuộc vào ngữ cảnh. Ví dụ nhưng khi dự đoán từ cuối cùng trong đoạn văn bản *"Tôi sinh ra và lớn lên ở Việt_Nam ... Tôi có_thể nói thuần_thục Tiếng_Việt."* Từ thông tin gần nhất cho thấy rằng từ tiếp theo là tên một ngôn ngữ, nhưng khi chúng ta muốn biết cụ thể ngôn ngữ nào, thì cần quay về quá khứ xa hơn, để tìm được ngữ cảnh *Việt_Nam*. Và như vậy, RRN có thể phải tìm những thông tin có liên quan và số lượng các điểm đó trở nên rất lớn.

Không được như mong đợi, RNN không thể học để kết nối các thông tin lại với nhau.



Hình 2.8: RNN phụ thuộc long-term.

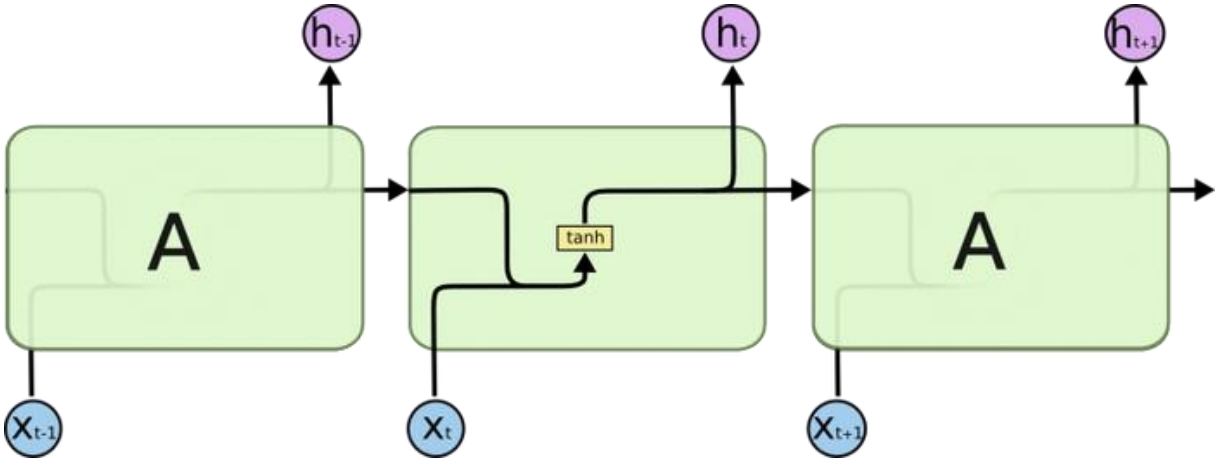
Về lý thuyết, RNN hoàn toàn có khả năng xử lý *"long-term dependencies"* [14], nghĩa là thông tin hiện tại có được là nhờ vào chuỗi thông tin trước đó. Đáng buồn là, trong thực tế, RNN dường như không có khả năng này. Vấn đề này đã được Hochreiter (1991) [German] and Bengio, và công sự đưa ra như một thách thức cho mô hình RNN.

Rất may là chúng ta đã có mạng LSTM giải quyết được vấn đề này!

2.4.2 Kiến trúc mạng LSTM

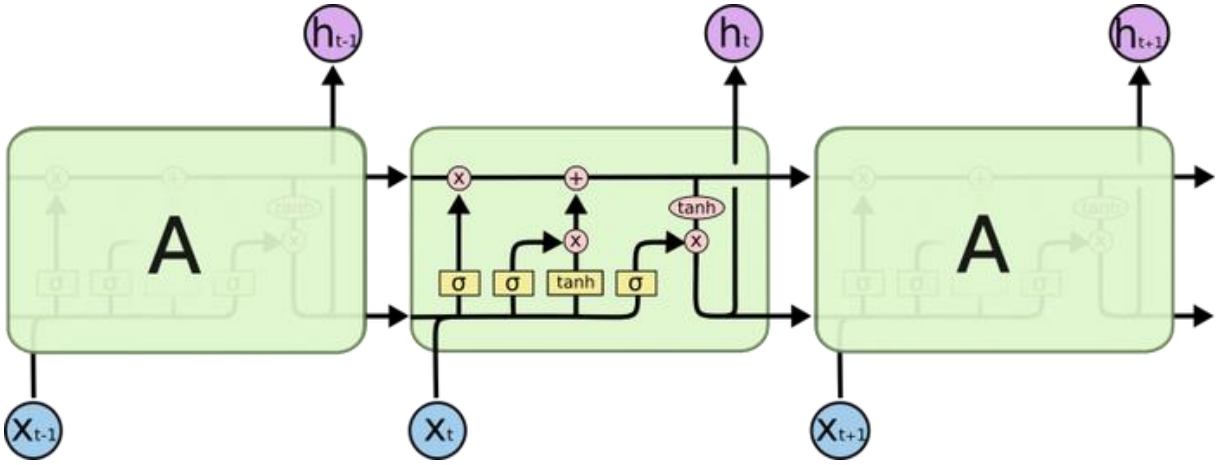
Long Short Term Memory network (LSTM) là trường hợp đặc biệt của RNN, có khả năng học long-term dependencies. Mô hình này được giới thiệu bởi Hochreiter & Schmidhuber (1997) [12], và được cải tiến lại. Sau đó, mô hình này dần trở nên phổ biến nhờ vào các công trình nghiên cứu gần đây. Mô hình này có 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 phụ thuộc quá dài [14]. Ta quan sát lại mô hình RNN bên dưới, các layer đều mắc nối với nhau thành các module neural network. Trong RNN chuẩn, module repeating này có cấu trúc rất đơn giản chỉ gồm một lớp đơn giản tanh layer.



Hình 2.9: Các mô-đun lặp của mạng RNN chứa một layer.

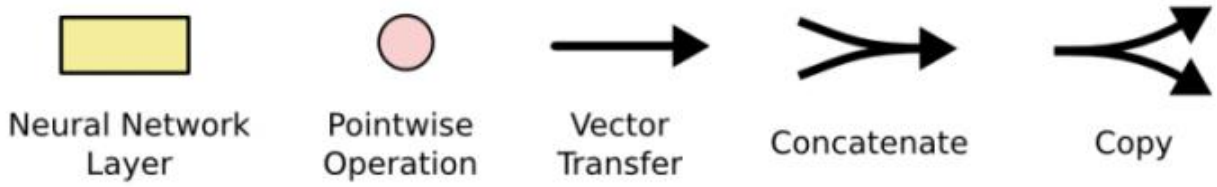
LSTM cũng có cấu trúc mắt xích tương tự, nhưng các module lặp có cấu trúc khác hẳn. Thay vì chỉ có một layer neural network, thì LSTM có tới bốn layer, tương tác với nhau theo một cấu trúc cụ thể.



Hình 2.10: Các mô-đun lặp của mạng LSTM chứa bốn layer.

Trong đó, các ký hiệu sử dụng trong mạng LSTM được giải nghĩa như hình 2.9 sau đây:

- Hình chữ nhật nền vàng là các lớp ẩn của mạng nơ-ron
- Hình tròn nền hồng biểu diễn toán tử Pointwise
- Đường kẻ gộp lại với nhau biểu thị phép nối các toán hạng
- Và đường rẽ nhánh biểu thị cho sự sao chép từ vị trí này sang vị trí khác



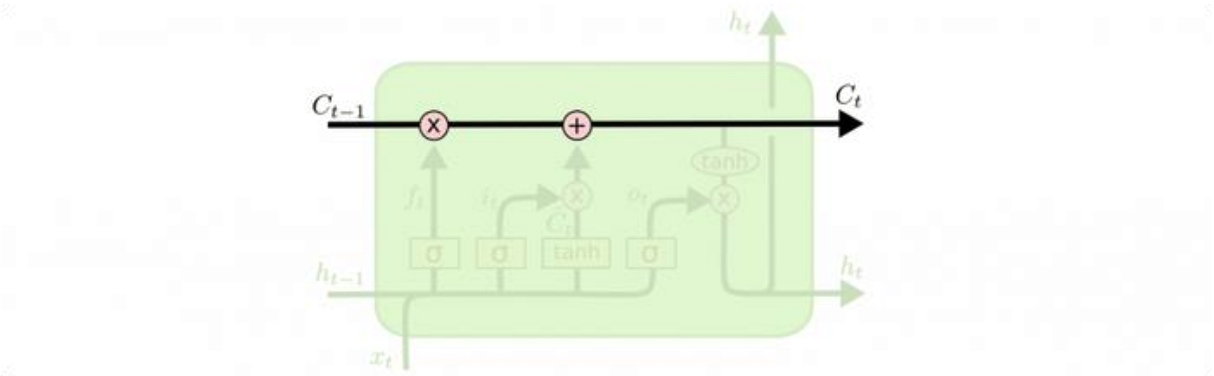
Hình 2.11: Các kí hiệu sử dụng trong mạng LSTM.

2.4.3 Phân tích mô hình LSTM

Có lẽ sau khi quan sát mô hình thiết kế của LSTM, chúng ta 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.

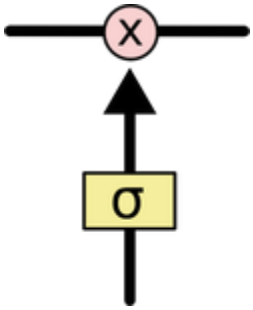
Mấu chốt của LSTM là cell state (tế bào trạng thái), đường kẻ ngang chạy dọc ở trên top diagram. Cell state giống như băng chuyền. Nó chạy xuyên thẳng toàn bộ mắc xích, chỉ một vài tương tác nhỏ tuyến tính (minor linear interaction)

được thực hiện. Điều này giúp cho thông tin ít bị thay đổi xuyên suốt quá trình lan truyền.



Hình 2.12: Tế bào trạng thái LSTM giống như một băng truyền.

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 (tuỳ chọn) để định nghĩa thông tin băng qua. Chúng được tạo bởi hàm sigmoid và một toán tử nhân pointwise.



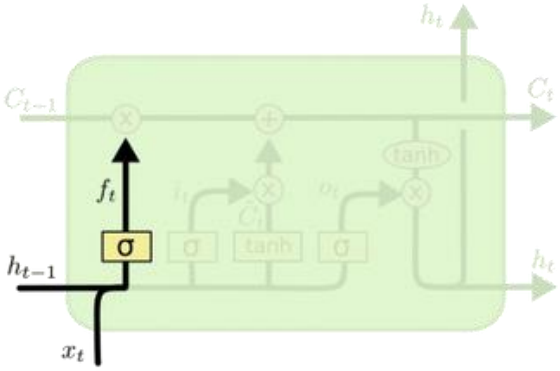
Hình 2.13: Cổng trạng thái LSTM.

Hàm kích hoạt Sigmoid có giá trị từ 0 – 1, mô tả độ lớn thông tin được phép truyền qua tại mỗi lớp mạng. 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.

Quá trình hoạt động của LSTM được thông qua các bước cơ bản sau:

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" – cổng chặn. Đầu vào là h_{t-1} và x_t, đầu ra là một

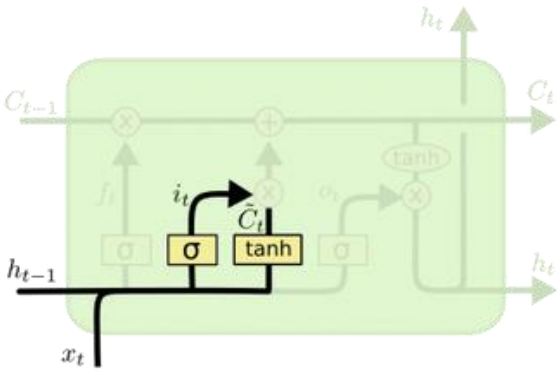
giá trị nằm trong khoảng [0, 1] cho cell state C_{t-1}. 1 tương đương với "giữ lại thông tin", 0 tương đương với "loại bỏ thông tin".



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

Hình 2.14: LSTM focus f.

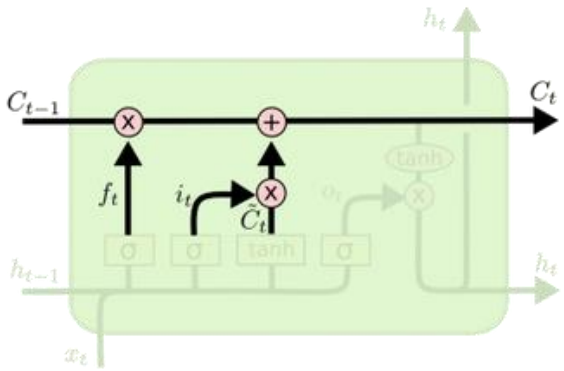
Bước tiếp theo, ta cần quyết định thông tin nào cần được lưu lại tại cell state. Ta có hai phần. Một, single sigmoid layer được gọi là "input gate layer" quyết định các giá trị chúng ta sẽ cập nhật. Tiếp theo, một $tanh$ layer tạo ra một vector ứng viên mới, $\tilde{C}_t$. được thêm vào trong ô trạng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



$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

Hình 2.15: LSTM focus i.

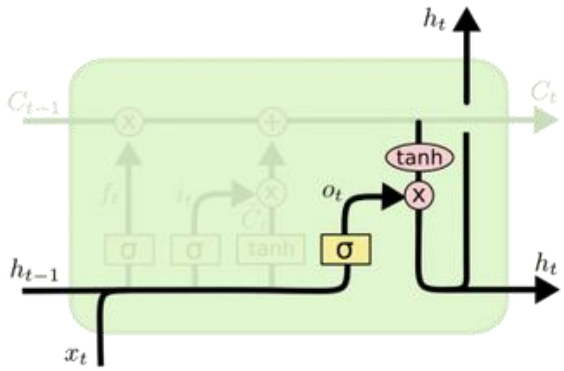
Ở bước tiếp theo, ta sẽ kết hợp hai thành phần này lại để cập nhật vào cell state. Lúc cập nhật vào cell state cũ, C_{t-1}, vào cell state mới C_t. Ta sẽ đưa state cũ hàm f_t, để quên đi những gì trước đó. Sau đó, ta sẽ thêm $i_t * C_t$. Đây là giá trị ứng viên mới, co giãn (scale) số lượng giá trị mà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



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

Hình 2.16: LSTM focus c.

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.



$$o_t = \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

Hình 2.17: LSTM focus o.

Mô hình LSTM là một bước đột phá mà chúng ta đạt được từ mô hình RNN.

CHƯƠNG 3: MÔ HÌNH ĐỐI THOẠI VỚI MẠNG NƠ-RON

Chương này sẽ giới thiệu về mô hình ngôn ngữ có thể sản sinh ra văn bản sau khi được huấn luyện bởi một mạng nơ-ron, đồng thời đề cập đến mô hình chuỗi tuần tự liên tiếp sequence to sequence. Và sẽ đi xem xét làm thế nào để xây dựng được một mô hình đối thoại sử dụng mạng nơ-ron.

3.1 Mô hình ngôn ngữ phát sinh văn bản

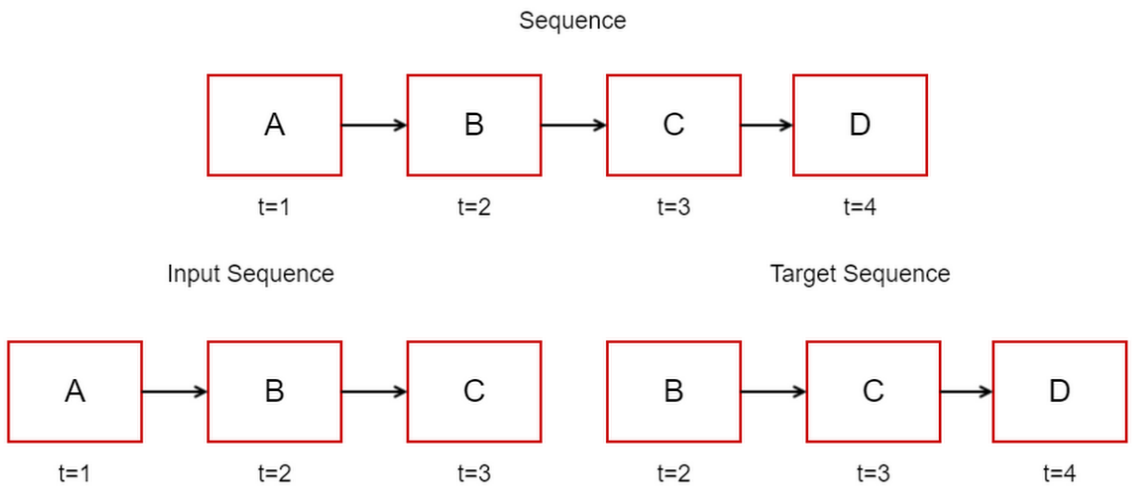
Nền tảng của việc xây dựng mô hình chuỗi tuần tự (ví dụ, mô hình dịch máy) là mô hình ngôn ngữ. Ở mức cao, một mô hình ngôn ngữ đón nhận chuỗi các phần tử đầu vào, nhìn vào từng phần tử của chuỗi và cố gắng để dự đoán các phần tử tiếp theo của chuỗi văn bản. Có thể mô tả quá trình này bằng phương trình hàm số sau đây:

$$Y_t = f(Y_{t-1})$$

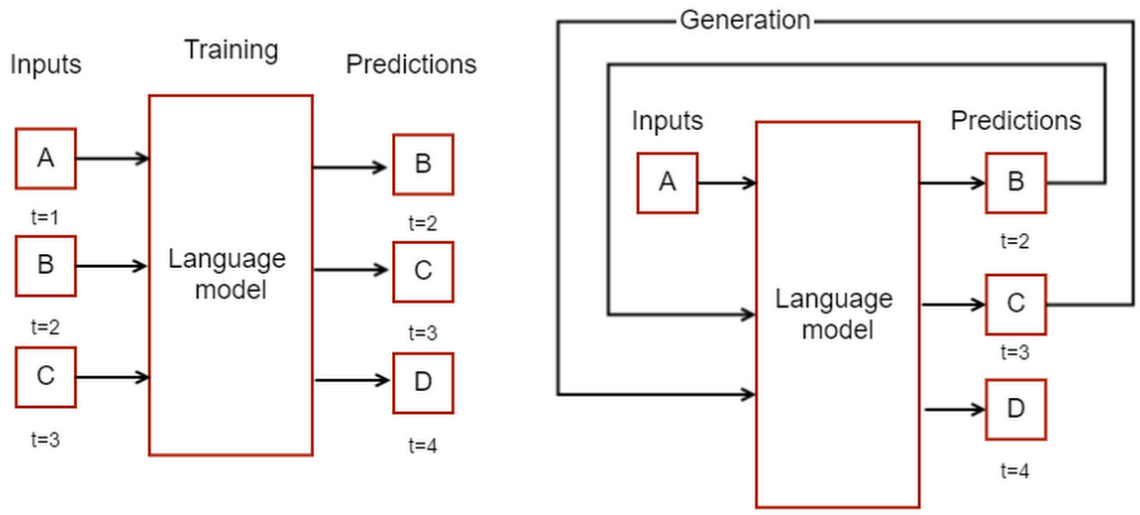
Trong đó, Y_t là phần tử chuỗi ở thời điểm t, Y_{t-1} là phần tử chuỗi ở thời điểm trước đó (t – 1), và f là hàm ánh xạ các phần tử trước đó của chuỗi sang phần tử tiếp theo của chuỗi. Bởi vì chúng ta đang đề cập đến mô hình chuỗi sử dụng mạng nơ-ron, f đại diện cho mạng nơ-ron mà có thể dự đoán được phần tử tiếp theo của một chuỗi, được cho trước bởi một phần tử hiện tại trong chuỗi đó.

Mô hình ngôn ngữ có thể sinh sản, khi được huấn luyện thì chúng có thể được sử dụng để sinh ra các chuỗi thông tin bằng cách cho kết quả đầu ra ở bước trước trở lại làm đầu vào của mô hình. Hình vẽ dưới đây là sơ đồ cho thấy việc huấn luyện và quá trình sinh sản của một mô hình ngôn ngữ.

Cho một chuỗi là ABCD. Một chuỗi đầu vào là một lát cắt của chuỗi cho đến phần tử cuối. Chuỗi đích target là một lát cắt của chuỗi từ phần tử thứ 2.



Hình 3.1: Mô hình phát sinh văn bản



Hình 3.2: Quá trình huấn luyện và phát sinh văn bản

Trong quá trình training, mô hình cố gắng dự đoạn phần tử tiếp theo của chuỗi target được cho bởi phần tử hiện tại của chuỗi target. Trong quá trình sinh, mô hình sinh sẽ lấy kết quả đã được sinh ra ở bước trước làm đầu vào cho lần dự báo tiếp theo.

Có nhiều cách khác nhau để xây dựng một mô hình ngôn ngữ, nhưng trong luận văn này chỉ đề cập đến việc huấn luyện mô hình ngôn ngữ dữ trên Mạng nơ-ron tái phát RNN. Như đã biết mạng RNN giống với mạng ANN truyền thống, nhưng chúng thao tác và xử lý các dữ liệu dạng chuỗi. Về cơ bản mạng RNN tiếp

nhận mỗi phần tử của chuỗi, nhân phân tử với một ma trận, sau đó cộng tổng kết quả với kết quả ở bước trước của mạng. Ta có phương trình biểu diễn như thế sau.

$$h_t = activation(X_t W_x + h_{t-1} W_h)$$

Để sử dụng mạng RNN cho mô hình ngôn ngữ, chúng ta sẽ đưa chuỗi đầu vào từ *t = 1* đến *t = seq_length – 1* và cố gắng dự đoán chuỗi tương tự từ t = 2 đến *t = seq_length – 1*. Khi đầu ra của RNN được dựa trên tất cả các đầu vào của sequence, thì output của nó được biểu diễn bởi hàm $Y_t = g(f(Y_{t-1}, Y_{t-2}, \dots, Y_{t1}))$. Hàm *f* tạo ra trạng thái tiếp theo của mạng RNN, trong khi hàm *g* ánh xạ trạng thái của RNN vào một giá trị trong tập các từ vựng target (vocabulary). Một cách đơn giản, *f* cho ra một trạng thái ẩn của mạng, trong khi hàm *g* cho đầu ra của mạng – giống với softmax layer của mạng nơ-ron đơn giản.

Không giống với các mô hình ngôn ngữ đơn giản là chỉ dự đoán xác suất cho từ tiếp theo khi được cho bởi từ hiện tại, mô hình RNN chụp lại toàn bộ bối cảnh của chuỗi đầu vào. Đo đó, RNN dự đoán xác suất tạo ra các từ tiếp theo dựa trên các từ hiện tại, cũng như tất cả các từ trước.

3.2 Mô hình chuỗi tuần tự liên tiếp seq2seq

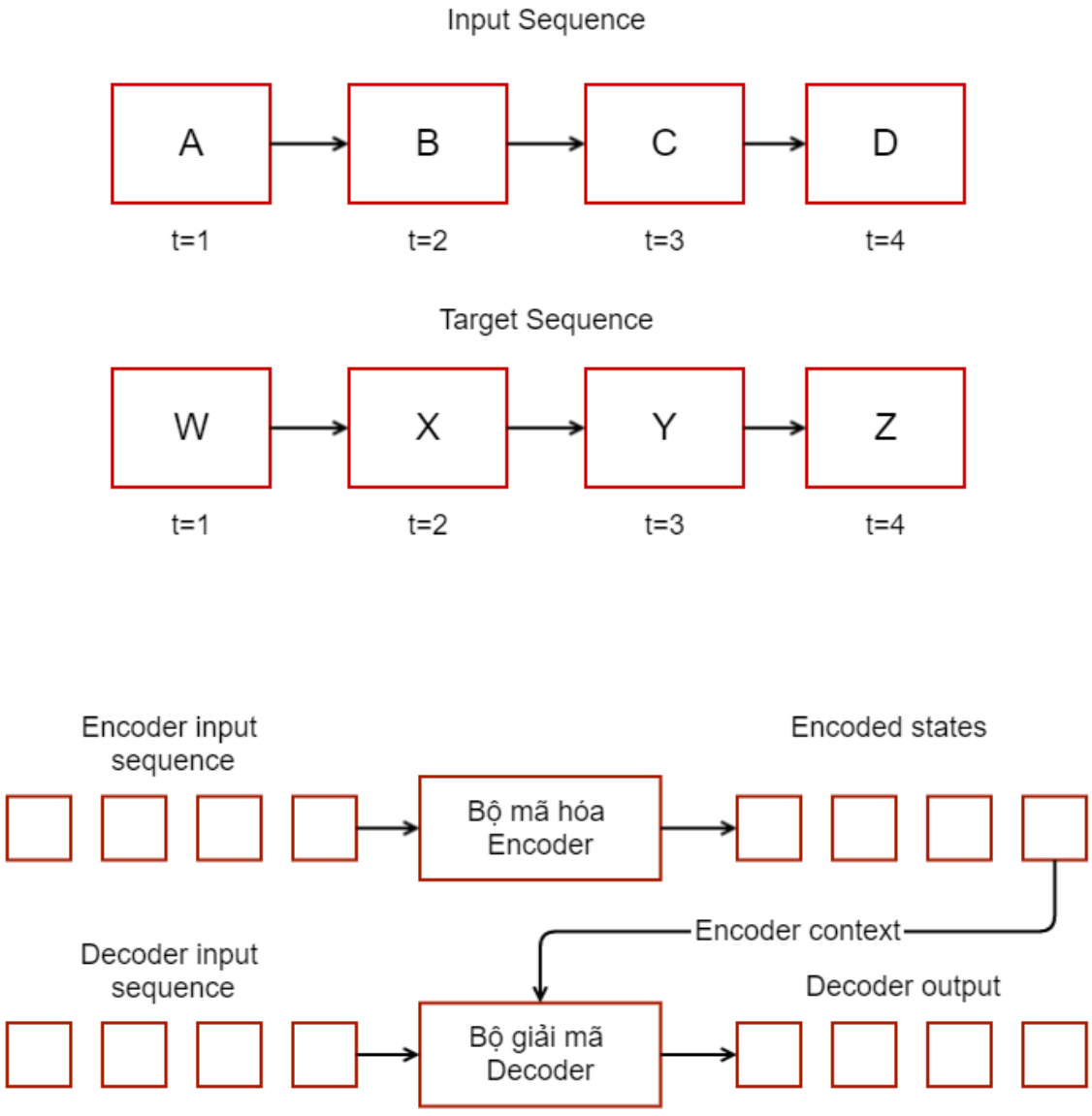
RNN có thể được sử dụng như là mô hình ngôn ngữ cho việc dự đoán các phần tử của một chuỗi khi cho bởi các phần tử trước đó của một chuỗi. Tuy nhiên, chúng ta vẫn còn thiếu các thành phần cần thiết cho việc xây dựng các mô hình đối thoại, hay các mô hình máy dịch, bởi vì chúng ta chỉ có thể thao tác trên một chuỗi đơn, trong khi việc dịch hoạt động trên cả hai chuỗi – chuỗi đầu vào và chuỗi được dịch sang.

Các mô hình chuỗi sang chuỗi được xây dựng bên trên mô hình ngôn ngữ bằng việc thêm vào một bộ mã hóa *Encoder* và một bộ giải mã *Decoder*. Trong bước mã hóa encode, một mô hình chuyển đổi một chuỗi đầu vào (ví dụ như một câu tiếng Anh) thành một đại diện cố định. Trong bước giải mã decode, một mô hình ngôn ngữ được huấn luyện trên cả hai chuỗi output được dịch ra và chuỗi đại điện cố định (từ bộ mã hóa encoder). Khi bộ mã hóa nhìn thấy cả hai thông tin chuỗi đầu và đã được mã hóa và chuỗi được dịch ra, nó có thể dự đoán thông minh hơn về các từ tương lai dựa và các từ hiện tại. Ví dụ, trong một mô hình ngôn ngữ chuẩn, chúng ta có thể thấy một từ "**đi**" trong tiếng Việt và không chắc từ tiếp

theo là về sự dịch chuyển bằng hai chi dưới (ví dụ: *tôi đi rất nhanh nhưng vẫn không đuổi kịp anh ấy*) hay chỉ một người nào đó đã *chết* (ví dụ: *Anh ấy ra đi mà không kịp nói lời trăng trối*). Tuy nhiên, nếu chúng ta đã đi qua một bối cảnh của bộ mã hóa, thì bộ giải mã nhận ra rằng các chuỗi đầu vào đang đề cập đến việc di chuyển của con người, chứ không phải sực việc một người đã chết. Với bối cảnh đó, bộ giải mã có thể lựa chọn từ kế tiếp thích hợp và cung cấp một bản dịch chính xác.

Mô hình chuỗi sang chuỗi Seq2seq, [5] được giới thiệu trong bài báo "Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation", kể từ đó đã trở thành mô hình cho các hệ thống đối thoại (Dialogue Systems) và Máy dịch (Machine Translation).

Như vậy, chúng ta đã có thể hiểu được cơ bản về mô hình chuỗi sang chuỗi, chúng ta có thể xem xét làm thế nào để xây dựng được một mô hình dịch sử dụng mạng nơ-ron: Với bộ mã hóa, sẽ sử dụng một mạng RNN. Mạng này sẽ xử lý chuỗi đầu vào, sau đó chuyển nó thành chuỗi đầu ra của nó vào bộ giải mã decoder như một biến ngữ cảnh. Với bộ giải mã, cũng sử dụng một mạng RNN. Nhiệm vụ của nó là xem kết quả được dịch, và sau đó cố gắng dự đoán từ tiếp theo trong chuỗi được giải mã khi đã biết được từ hiện tại trong chuỗi đã được dịch. Sau khi huấn luyện, mô hình có thể dịch bằng việc mã hóa câu mà chúng ta muốn dịch và sau đó chạy mạng ở chế độ sinh văn bản. Mô hình chuỗi sang chuỗi được mô phỏng như hình dưới đây:



Hình 3.3: Mô hình chuỗi liên tiếp (chuỗi sang chuỗi) seq2seq.

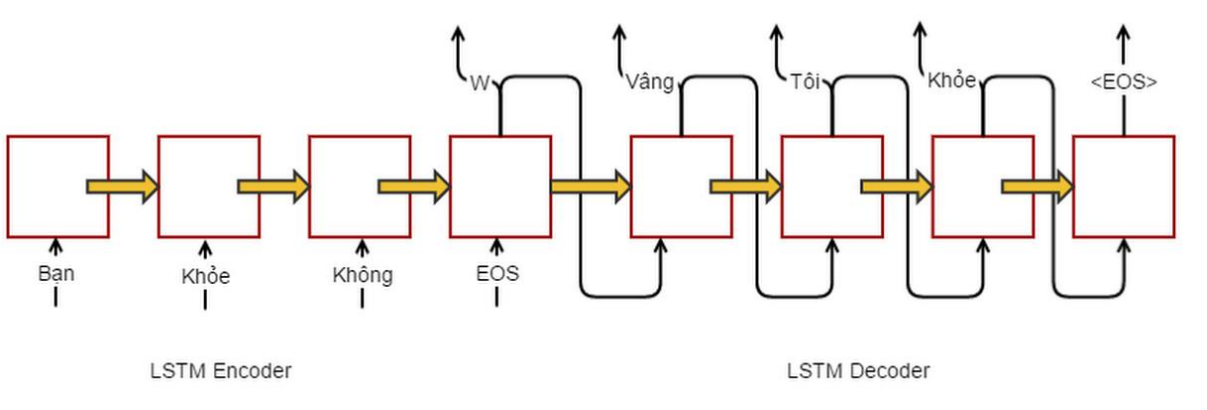
Mô hình chuỗi sang chuỗi, bộ mã hóa sinh ra một chuỗi các trạng thái. Bộ giải mã là một mô hình ngôn ngữ với một tham số bổ sung cho các trạng thái cuối cùng của bộ mã hóa.

Như vậy, chúng ta đã thấy được một mô hình ngôn ngữ đơn giản cho phép chúng ta mô hình hóa các chuỗi đơn giản bằng việc dự đoán tiếp theo trong một chuỗi khi cho một từ trước đó trong chuỗi. Thêm nữa là chúng ta đã thấy quá trình xây dựng một mô hình phức tạp có phân tách các bước như mã hóa một chuỗi đầu vào thành một bối cảnh, và sinh một chuỗi đầu ra bằng việc sử dụng một mạng

nơ-ron tách biệt. Trong phần sau sẽ trình bày cách thiết kế một mô hình đối thoại dựa vào mô hình chuỗi sang chuỗi seq2seq.

3.3 Mô hình đối thoại seq2seq

Bản thân mô hình seq2seq nó bao gồm hai mạng RNN: Một cho bộ mã hóa, và một cho bộ giải mã. Bộ mã hóa nhận một chuỗi (câu) đầu vào và xử lý một phần tử (từ trong câu) tại mỗi bước. Mục tiêu của nó là chuyển đổi một chuỗi các phần tử vào một vectơ đặc trưng có kích thước cố định mà nó chỉ mã hóa thông tin quan trọng trong chuỗi và bỏ qua các thông tin không cần thiết. Có thể hình dung luồng dữ liệu trong bộ mã hóa dọc theo trục thời gian, giống như dòng chảy thông tin cục bộ từ một phần tử kết thúc của chuỗi sang chuỗi khác.



Hình 3.4: Mô hình đối thoại seq2seq.

Mỗi trạng thái ẩn ảnh hưởng đến trạng thái ẩn tiếp theo và trạng thái ẩn cuối cùng được xem như tích lũy tóm tắt về chuỗi. Trạng thái này được gọi là bối cảnh hay vector suy diễn, vì nó đại diện cho ý định của chuỗi. Từ bối cảnh đó, các bộ giải mã tạo ra một chuỗi, một phần tử (word) tại một thời điểm. Ở đây, tại mỗi bước, các bộ giải mã bị ảnh hưởng bởi bối cảnh và các phần tử được sinh ra trước đó.

3.4 Những thách thức chung khi xây dựng mô hình đối thoại

Có một số thách thức thể hiện một cách rõ ràng hoặc không thể thấy rõ khi xây dựng một mô hình đối thoại nói chung đang là tâm điểm được chú ý bởi nhiều nhà nghiên cứu.

3.4.1 Phụ thuộc bối cảnh

Để sinh ra các câu trả lời hợp lý, các hệ thống đối thoại cần phải kết hợp với cả hai bối cảnh ngôn ngữ và bối cảnh vật lý. Trong các hội thoại dài, người nói cần theo dõi và nhớ được những gì đã được nói và nhưng thông tin gì đã được trao đổi. Đó là một ví dụ về bối cảnh ngôn ngữ. Phương pháp tiếp cận phổ biến nhất là nhúng cuộc hội thoại vào một Vector, nhưng việc làm này đối với đoạn hội thoại dài là một thách thức lớn. Các thử nghiệm trong các nghiên cứu [3], [15] đều đi theo hướng này. Hướng nghiên cứu này cần kết hợp các loại bối cảnh như: Ngày/giờ, địa điểm, hoặc thông tin về một người.

3.4.2 Kết hợp tính cách

Khi phát sinh các câu trả lời, các hệ thống trợ lý ảo lý tưởng là tạo ra câu trả lời phù hợp với ngữ nghĩa đầu vào cần nhất quán giống nhau. Ví dụ, chúng ta muốn nhận được câu trả lời với mẫu hỏi "*Bạn bao nhiêu tuổi*?" hay "*Tuổi của bạn là mấy*?". Điều này nghe có vẻ đơn giản, nhưng việc tổng hợp, tích hợp các kiến thức nhất quán hay "có tính cách" vào trong các mô hình đối thoại là một vấn đề rất khó để nghiên cứu.

message	Where do you live now?
response	I live in Los Angeles.
message	In which city do you live now?
response	I live in Madrid.
message	In which country do you live now?
response	England, you?

Hình 3.5: Vấn đề phụ thuộc bối cảnh và tính cách.

Rất nhiều các hệ thống được huấn luyện để trả lời câu hỏi thỏa đáng với ngôn ngữ, nhưng chúng không được huấn luyện để sinh ra các câu trả lời nhất quán về ngữ nghĩa. Mô hình như thế đang được nghiên cứu trong [10], tạo ra những bước đầu tiên tập trung vào hướng mô hình hóa tính cách.

CHƯƠNG 4: XÂY DỰNG MÔ HÌNH ĐỐI THOẠI CHO TIẾNG VIỆT

Chương này sẽ đi xây dựng một mô hình đối thoại sử dụng mạng nơ-ron tái phát, mà sẽ đọc chuỗi đầu vào tuần tự tại mỗi thời điểm, và dự đoán một chuỗi đầu ra, cũng một dấu hiệu tại một thời điểm. Đồng thời, cũng đề cập đến các vấn đề khi thiết kế và xây dựng mô hình sử dụng mạng nơ-ron tái phát.

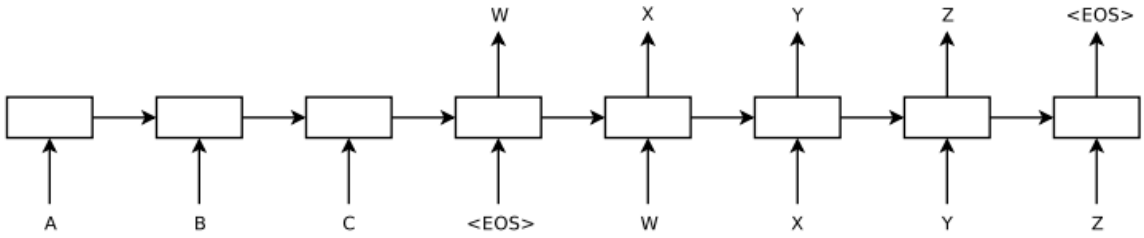
4.1 Kiến trúc ứng dụng

Mạng học sâu Deep Neural Networks (DNN) là mô hình học máy cực kỳ mạnh mẽ, đạt được những kết quả xuất sắc về các vấn đề khó như nhận dạng giọng nói [13, 7] và nhận dạng hình ảnh đối tượng [19, 6, 21, 20]. DNN rất mạnh bởi vì chúng có thể thực hiện tính toán song song tùy ý với một số lượng rất ít các bước. Hơn nữa, Mạng DNN lớn có thể được huấn luyện với lan truyền ngược giám sát 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, giám sát lan truyền ngược sẽ tìm thấy những thông số và giải quyết vấn đề.

Mặc dù có sự linh hoạt và sức mạnh của chúng, các mạng DNN chỉ có thể được áp dụng cho các vấn đề mà các đầu vào và các mục tiêu có thể được mã hóa đúng cách với vectơ 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 được biết trước. Ví dụ, nhận dạng giọng nói và dịch máy là vấn đề tuần tự. Tương tự như vậy, hỏi đáp cũng có thể được xem như là một ánh xạ của một chuỗi tuần tự các từ, đại diện cho một câu hỏi, sang một chuỗi từ đại diện cho câu trả lời.

Các chuỗi tuần tự đặt ra một thách thức đối với DNN, vì chúng yêu cầu các chiều của các yếu tố đầu vào và đầu ra được biểu diễn cố định. Trong luận văn này, chúng tôi cho thiết kế một ứng dụng đơn giản dựa vào kiến trúc Long Short-Term Memory (LSTM) [12] có thể giải quyết các vấn đề chuỗi tuần tự liên tiếp sequence-to-sequence. Ý tưởng là sử dụng một mạng LSTM để đọc chuỗi đầu vào, một bước thời gian tại một thời điểm, để có được biểu diễn vector kích thước cố định, và sau đó sử dụng một mạng LSTM để trích xuất các trình tự đầu ra từ vector đó (hình 4.1). Mạng LSTM thứ hai về cơ bản là một mạng nơ-ron tái phát dựa trên mô hình ngôn ngữ [40, 41], ngoại trừ việc nó được bổ sung thêm các điều kiện trên các chuỗi đầu vào. LSTM có khả năng học thành công trên dữ liệu phụ thuộc thời gian tầm xa, làm cho nó trở thành một sự lựa chọn tự nhiên cho ứng

dụng này do độ có trễ thời gian đáng kể giữa các đầu vào và đầu ra tương ứng của chúng (hình 4.1).



Hình 4.1: Kiến trúc mô hình đối thoại cho tiếng Việt.

Kiến trúc mô hình trên chúng tôi dựa vào kết quả nghiên cứu của Lê Viết Quốc cho bài toán hỏi đáp bằng tiếng Anh, trong [6], chúng tôi cũng sử dụng mô hình này sẽ đọc một câu đầu vào tiếng Việt, ví dụ: “A B C“ và sinh ra ra một câu tiếng Việt đầu ra “W X Y Z“. Mô hình sẽ dừng dự đoán sau khi sản xuất ra một mã hiệu kết thúc câu <EOS>. Lưu ý, mạng LSTM đọc câu đầu vào theo hướng ngược lại, bởi vì làm như vậy sẽ đưa ra nhiều các phụ thuộc ngắn hạn trong các dữ liệu mà làm cho các vấn đề được tối ưu hơn nhiều.

Tiếp cận của chúng tôi sử dụng một khung làm việc sequence-to-sequence (seq2seq) được mô tả trong [7]. Mô hình này dựa trên một mạng nơ-ron tái phát, mà sẽ đọc chuỗi đầu vào tuần tự, một dấu hiệu (token) tại mỗi thời điểm, và dự đoán chuỗi đầu ra, cũng một dấu hiệu tại một thời điểm. Trong suốt thời gian huấn luyện, chuỗi tuần tự đầu ra được đưa vào mô hình, và việc học có thể hoàn tất bởi quá trình lan truyền ngược. Mô hình này được huấn luyện để cực đại hóa cross-entropy theo đúng tuần tự cho bối cảnh của nó. Trong quá trình suy luận, mô hình cho chuỗi đầu ra đúng mà không quan sát được, bằng cách đơn giản chúng tôi nạp vào dấu hiệu token đã được dự đoán làm đầu vào để dự đoán dấu hiệu đầu ra tiếp theo. Đây là một phương pháp suy luận "tham lam". Một cách tiếp cận ít tham lam sẽ được sử dụng tìm kiếm Beam Search, đây là thuật toán tìm kiếm mà có thể phát hiện ra một đồ thị bằng việc mở rộng các nút tiềm năng trong một tập có giới hạn, bằng cách nạp một vài ứng cử viên ở các bước trước vào bước tiếp theo. Một chuỗi được dự đoán có thể được chọn dựa trên xác suất của chuỗi.

Cụ thể, giả sử rằng chúng ta quan sát một cuộc trò chuyện với hai lượt: người đầu tiên thốt ra "ABC", và người thứ hai trả lời "WXYZ". Chúng tôi có thể sử dụng một mạng nơ-ron tái phát, và huấn luyện để ánh xạ "ABC" sang "WXYZ" như trên hình 4.1 ở trên. Các trạng thái ẩn của mô hình khi đó nhận được ký tự

kết thúc chuỗi <EOS>, có thể được xem như là vector ngưỡng suy nghĩ vì nó lưu trữ các thông tin của câu, hoặc nghĩ, "ABC".

Thế mạnh của mô hình này nằm ở sự đơn giản và tính tổng quát của nó. Chúng ta có thể sử dụng mô hình này cho Máy dịch, Hỏi đáp, và các cuộc trò chuyện mà không cần thay đổi nhiều trong kiến trúc. Việc áp dụng kỹ thuật này để mô hình hóa cuộc đối thoại cũng rất đơn giản: các chuỗi đầu vào có thể được nối bối cảnh đã được trò chuyện với chuỗi đầu ra là câu trả lời.

Không giống như các nhiệm vụ đơn giản hơn như dịch thuật, tuy nhiên, một mô hình như sequence-to-sequence sẽ không thể "giải quyết" thành công vấn đề của việc mô hình hóa đối thoại do: các hàm mục tiêu được tối ưu hóa không nắm bắt được mục tiêu thực tế cần đạt được thông qua giao tiếp của con người, mà thường là thông tin dài hạn và dựa trên trao đổi thông tin chứ không phải là dự đoán bước tiếp theo. Việc thiếu một mô hình để đảm bảo tính thống nhất và kiến thức nói chung cũng là một hạn chế rõ ràng của một mô hình hoàn toàn không có giám sát.

4.2 Cài đặt mô hình

Mạng nơ-ron tái phát RNN [42, 43] là một mạng tổng quát của các mạng nơ-ron truyền thẳng cho các chuỗi tuần tự. Với mỗi chuỗi đầu vào $(x_1, \dots, x_T)$, là một mạng RNN chuẩn sẽ tính toán một chuỗi các kết quả đầu ra $(y_1, \dots, y_T)$, bằng cách duyệt phương trình sau:

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

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

Mạng RNN có thể dễ dàng ánh xạ tuần tự chuỗi bất cứ khi nào sự liên kết giữa đầu vào và đầu ra được biết đến trước khi hết hạn. Tuy nhiên, nó không là cách rõ ràng để áp dụng một mạng RNN cho các vấn đề mà đầu vào và đầu ra có độ dài khác nhau với các mối quan hệ phức tạp và không đơn điệu (thay đổi).

Cách làm đơn giản nhất cho việc học chuỗi nói chung là ánh xạ chuỗi đầu vào thành một vector có kích thước cố định sử dụng một mạng RNN, như đã đề cập đến trong mục 3.4, và sau đó, ánh xạ vector vào chuỗi đích sử dụng một mạng RNN khác (cách làm này được thực hiện bởi Cho và cộng sự [5]). Trong khi nó có thể hoạt động trên nguyên tắc kể từ khi RNN được cung cấp với tất cả các thông tin liên quan, nó sẽ gặp khó khăn trong việc huấn luyện do sự phụ thuộc

thời gian dài [12, 44]. Tuy nhiên, mạng LSTM [12] có thể học các vấn đề phụ thuộc thời gian dài, vì vậy, sử dụng mạng LSTM có thể thành công trong trường hợp này.

Mục tiêu của LSTM là để ước lượng xác suất có điều kiện $p(y_1, \dots, y_{T'}|x_1, \dots, x_T)$ trong đó $(x_1, \dots, x_T)$ là một chuỗi đầu vào và $(y_1, \dots, y_{T'})$ là chuỗi đầu ra tương ứng của nó có chiều dài T' có thể khác nhau từ T. Mạng LSTM tính xác suất có điều kiện này bằng cách có được thông tin đại diện mà số chiều cố định v của chuỗi đầu vào $(x_1, \dots, x_T)$ được tính bởi các trạng thái ẩn cuối cùng của mạng LSTM, và sau đó tính toán xác suất của $(y_1, \dots, y_{T'})$ với một công thức LSTM-LM tiêu chuẩn mà ban đầu trạng thái ẩn được thiết lập để đại diện v của $(x_1, \dots, x_T)$:

$$p(y_1, \dots, y_{T'}|x_1, \dots, x_T) = \prod_{t=1}^{T'} p(y_t|v, y_1, \dots, y_{t-1})$$

Trong phương trình này, mỗi phân phối xác suất $p(y_1, \dots, y_{T'}|x_1, \dots, x_T)$ được biểu diễn bởi một hàm softmax trên tất cả từ trong từ vựng. Chúng tôi sử dụng công thức LSTM của Graves, trong [45]. Chú ý là mỗi câu kết thúc với một ký hiệu đặc biệt end-of-sentence "<EOS>", cho phép mô hình để xác định một phân phối các chuỗi của tất cả các độ dài có thể. Xem lược đồ tổng quát trong hình 4.1, trong đó LSTM tính xác suất đại diện của “A”, “B”, “C”, “<EOS>” và sau đó sử dụng đại diện này để tính xác xuất của “W”, “X”, “Y”, “Z”, “<EOS>”.

4.3 Các vấn đề và giải pháp khắc phục

Có một vài thách thức trong việc sử dụng mô hình này. Một trong những vấn đề đáng ngại nhất là các mô hình không thể xử lý được các chuỗi dài. Bởi vì hầu như tất cả các ứng dụng chuỗi sang chuỗi, bao gồm cả độ dài các chuỗi. Vấn đề tiếp theo là kích thước từ vựng. Bộ giải mã phải chạy hàm softmax hơn trên một tập rất lớn các từ vựng (khoảng 20,000 từ) cho mỗi một từ xuất ra. Điều này sẽ làm chậm quá trình huấn luyện, cho dù phần cứng của bạn có thể đáp ứng được khả năng xử lý. Đại diện của một từ là rất quan trọng. Làm thế nào để có thể biểu diễn được các từ trong chuỗi ? Sử dụng one-hot vector (một cách đánh chỉ số sự xuất hiện của từ này trong dữ liệu từ điển – vocabulary) có nghĩa là chúng ta phải đối mặt với các vector thưa thớt lớn, do kích thước vốn từ vựng lớn mà không có

ý nghĩa về mặt ngữ nghĩa của từ được mã hóa bên trong các vector one-hot. Sau đây là một số vấn đề mà chúng ta sẽ gặp phải và cách khắc phục.

PADDING – Tạo độ dài cố định

Trước khi huấn luyện, chúng ta cần chuyển đổi độ dài của các phần tử trong chuỗi thành các chuỗi có độ dài cố định, bằng việc thêm vào các phần tử đệm PADDING. Các phần tử đệm đặc biệt mà chúng ta sẽ sử dụng:

1. **EOS**: Kết thúc câu (End of sentence)
2. **PAD**: Phần đệm bù (Filler)
3. **GO**: Bắt đầu giải mã (Start decoding)
4. **UNK**: Unknown; từ không biết, không có trong từ điển từ vựng

Xem xét một cặp ví dụ HỎI – ĐÁP sau đây:

***Q**: Bạn khỏe không ?*

***A**: Vâng tôi khỏe.*

Giả sử chúng ta muốn xử lý các đoạn hội thoại có độ dài 10, kết quả cặp Q/A trên sẽ được chuyển đổi thành như sau:

***Q** : [PAD, PAD, PAD, PAD, PAD, PAD, "?", "không", "khỏe", "Bạn"]*

***A** : [GO, "Vâng", "tôi", "khỏe", ".", EOS, PAD, PAD, PAD, PAD]*

BUCKETING – Tránh lu mờ thông tin

Bộ đệm đã giải quyết được vấn đề độ dài của các chuỗi, nhưng hãy xem xét một trường hợp các câu có độ dài lớn. Nếu câu dài nhất trong tập dữ liệu có độ dài là 100, chúng ta cần mã hóa tất cả các chuỗi còn lại bằng độ dài 100, để không mất thông tin của bất kỳ từ nào. Như vậy, chuyện gì xảy và với chuỗi từ *"Bạn khỏe không ?"*. Sẽ có 97 phần tử đệm PAD được sử dụng khi mã hóa một chuỗi câu. Điều này sẽ làm lu mờ thông tin thực tế trong câu.

Bucketing giải quyết vấn đề này bằng việc đặt các câu vào các xô buckets có kích thước khác nhau. Ví ta có một danh sách các xô buckets: [(5, 10), (10, 15), (20, 25), (40, 50)]. Nếu độ dài của mẫu hỏi là 4 như ví dụ trên sẽ được đặt

vào xô (5, 10). Mẫu hỏi sẽ được đệm với độ dài 5 và đáp án được đệm với độ dài 10. Trong lúc chạy mô hình (huấn luyện hoặc dự đoán), chúng ta sẽ sử dụng một mô hình khác cho mỗi bucket, tương ứng với các độ dài của mẫu hỏi và câu trả lời. Tất cả những mô hình này chia sẻ các tham số giống nhau và do đó hoạt động chính xác theo cùng một cách.

Nếu chúng ta sử dụng xô (5, 10), thì các câu sẽ được mã hóa thành:

***Q** : [PAD, "?", "không", "khỏe", "Bạn"]*

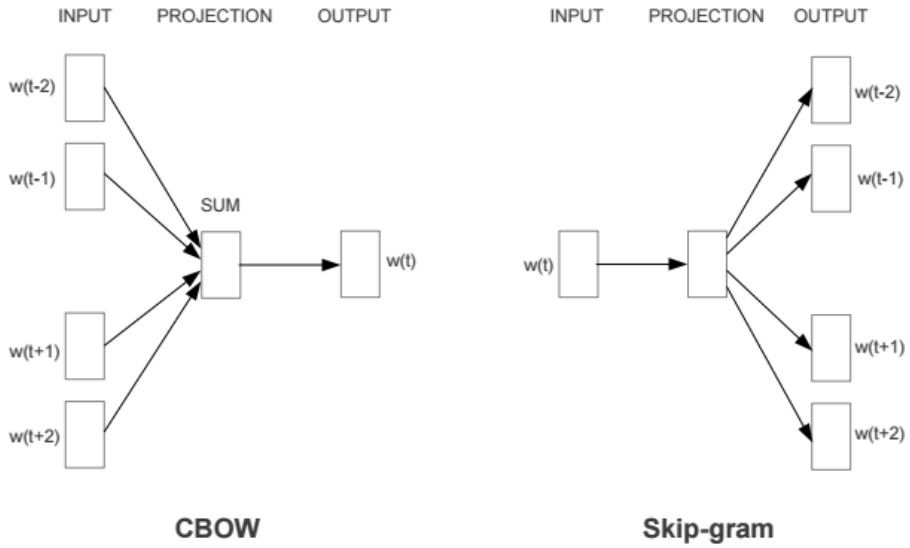
***A** : [GO, "Vâng", "tôi", "khỏe", ".", EOS, PAD, PAD, PAD, PAD]*

Word Embedding – Mật độ dày đặc

Word Embedding là một kỹ thuật cho việc học mật độ dày đặc thông tin đại diện của từ trong một không gian vector với số chiều nhỏ hơn. Mỗi một từ có thể được xem như là một điểm trong không gian này, được đại diện bởi một vector có độ dài cố định.

Word Embedding thường được thực hiện trong lớp đầu tiên của mạng: Trong đó lớp embedding sẽ ánh xạ một từ (chỉ số index của từ trong từ điển từ vựng) từ từ điển sang một vector dày đặc với kích thước đã cho. Trong mô hình seq2seq, trọng số của lớp embedding được huấn luyện giống nhau với các tham số khác của mô hình.

Trong nghiên cứu của Mikolov và cộng sự, 2013 [51]. Tác giả đề xuất hai kiến trúc để xây dựng word vector đó là Continuous Bag-of-Words model (**CBOW**), và Continuous Skip-gram model.



Trong đó, kiến trúc **CBOW**: Dự đoán từ hiện tại dựa trên ngữ cảnh của các từ trước đó. Skip-gram: Dự đoán các từ xung quanh khi cho bởi từ hiện tại.

CHƯƠNG 5: THỰC NGHIỆM VÀ ĐÁNH GIÁ MÔ HÌNH

Chương này tiến hành thực nghiệm mô hình đối thoại cho tiếng Viết bằng việc áp dụng mô hình đối thoại Seq2seq trên miền mở. Mô tả về dữ liệu thực nghiệm cũng như công cụ sử dụng trong hệ thống.

4.1 Dữ liệu và công cụ thực nghiệm

Chúng tôi thử nghiệm bộ dữ liệu trên miền mở sử dụng bộ dữ liệu phụ đề phim tiếng Việt được lấy từ nguồn mở OpenSubtitles 2016 [1].

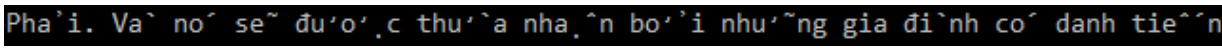
Đây là phiên bản sạch được công bố năm 2016, đã được cải thiện các hội thoại, dóng câu, kiểm tra song ngữ, và các siêu dữ liệu khác, gồm:

- 60 ngôn ngữ, 1,689 bitexts
- Tổng số file: 2,815,754
- Tổng số tokens: 17.18G
- Tổng số câu: 2.60G
- Trang chủ: http://www.opensubtitles.org/vi
- Download: http://opus.lingfil.uu.se/OpenSubtitles2016.php

Sau khi tiền xử lý dữ liệu, chúng tôi thu thập được bộ dữ liệu bao gồm 2,078,696 câu văn bản tiếng Việt. Các công đoạn làm sạch xử lý dữ liệu, chúng tôi đã thực hiện qua các bước sau:

- Loại bỏ các ký tự đặc biệt không phải chữ hoặc chữ số (bắt đầu, kết thúc và bên trong một câu tiếng Việt), **ex**: - *Xin chào, các bạn!, ...*
- Xóa bỏ các ký tự phân tách câu không phải dấu chấm, dấu hỏi hoặc dấu chấm than, **ex**: *@#$%^&*, ...*
- Xóa bỏ các bình luận, chú thích ý nghĩa các từ, thuật ngữ trong câu, **ex**: *Chatbot (chương trình tự động trả lời), ...*
- Xóa bỏ các ký tự lặp, ký tự phân tách không có ý nghĩa, **ex**: -,, ..., ...
- Xóa bỏ các thẻ html, **ex**: *Khi mặt trời ló dạng, ...*
- Biến đổi bảng mã html về dạng câu có ý nghĩa, **ex**: *Cho ch#250;ng t#244;i xem c#225;i c#242;n l#7841;i l#224; g#236; n#224;o.*

- Biến đổi bảng mã Unicode tổ hợp về dạng unicode dựng sẵn, **ex**:



- Loại bỏ các cặp câu không có ý nghĩa, ex: *Phụ_đề dịch bởi Unknow Subteam 2pi, ...*

Công cụ sử dụng:

Trong luận văn này để thử nghiệm được mô hình chúng tôi đã kết hợp sử dụng các thư viện mã nguồn mở và các công cụ tự xây dựng để xử lý dữ liệu, huấn luyện mô hình và dự báo.

- **NLTK**: Công cụ xử lý ngôn ngữ tự nhiên mã nguồn mở dành riêng cho NLP và được tích hợp vào Python. Nó đang ngày càng hoàn thiện và tích hợp các công cụ mới bởi hàng ngàn lập trình viên và cộng tác viên trên khắp thế giới. NLTK bao gồm những thư viện hàm, các công cụ phân tích, các corpus, wordnet, … giúp đơn giản hoá, tiết kiệm thời gian và công sức cho các lập trình viên.
- **VNTK**: Vietnamese languague toolkit, do chúng tôi xây dựng và phát triển để xử lý các vấn đề cơ bản của tiếng Việt, như: Tách câu, tách từ, làm sạch văn bản, ...
- **Subsent**: Công cụ hỗ trợ bóc tách dữ liệu từ các file phụ đề, do chúng tôi xây dựng và phát triển
- **Dongdu**: Thư viện hỗ trợ tách từ tiếng Việt [11], của tác giả Lưu Tuấn Anh
- **Tensorflow**: Một khung làm việc mã nguồn mở, do Google phát hành, được sử dụng để xây dựng các mô hình học máy, tạo môi trường nghiên cứu, thực hiện các thử nghiệm một cách nhanh chóng và dễ dàng, đặc biệt là có khả năng chuyển đổi các bản thiết kế prototype tới các ứng dụng trong sản xuất.
- **Python**: Ngôn ngữ lập trình để xây dựng mô hình đối thoại tiếng Việt.

4.2 Tách từ tập dữ liệu tiếng Việt

Tách từ là một quá trình xử lý nhằm mục đích xác định ranh giới của các từ trong câu văn, cũng có thể hiểu đơn giản rằng tách từ là quá trình xác định các từ đơn, từ ghép… có trong câu. Đối với xử lý ngôn ngữ, để có thể xác định cấu trúc ngữ pháp của câu, xác định từ loại của một từ trong câu, yêu cầu nhất thiết đặt ra là phải xác định được đâu là từ trong câu. Vấn đề này tưởng chừng đơn giản với con người nhưng đối với máy tính, đây là bài toán rất khó giải quyết.

Chính vì lý do đó tách từ được xem là bước xử lý quan trọng đối với các hệ thống Xử Lý Ngôn Ngữ Tự Nhiên, đặc biệt là đối với các ngôn ngữ thuộc vùng Đông Á theo loại hình ngôn ngữ đơn lập, ví dụ: tiếng Trung Quốc, tiếng Nhật, tiếng Thái, và tiếng Việt. Với các ngôn ngữ thuộc loại hình này, ranh giới từ không chỉ đơn giản là những khoảng trắng như trong các ngôn ngữ thuộc loại hình hòa kết như tiếng Anh…, mà có sự liên hệ chặt chẽ giữa các tiếng với nhau, một từ có thể cấu tạo bởi một hoặc nhiều tiếng. Vì vậy đối với các ngôn ngữ thuộc vùng Đông Á, vấn đề của bài toán tách từ là khử được sự nhập nhằng trong ranh giới từ.

Bởi vì các lý do trên, trước khi đưa vào mô hình huấn luyện và trả lời câu hỏi chúng tôi đã thực hiện tách từ tiếng Việt và sử dụng công cụ **DongDu** của tác giả Lưu Tuấn Anh trong [11] với độ chính xác tới 98% tính theo từ.

Kết quả sau khi chúng tôi làm sạch dữ liệu, tiền xử lý, tách từ, lựa chọn tập dữ liệu học chúng tôi thu được bộ dữ liệu, như sau:

- 120,885 words
- 362,655 tokens
- 1,824,063 QA

4.3 Khung làm việc Tensorflow

TensorFlow ™ là một thư viện phần mềm nguồn mở cho tính toán số sử dụng biểu đồ luồng dữ liệu. TensorFlow ban đầu được phát triển bởi các nhà nghiên cứu và kỹ sư làm việc trong nhóm Brain Google trong tổ chức nghiên cứu máy tình báo của Google nhằm mục đích tiến hành học máy và sâu nghiên cứu các mạng nơ-ron thần kinh, nhưng hệ thống là đủ nói chung có thể áp dụng trong một loạt các lĩnh vực khác như tốt.

TensorFlow ™ là một hệ thống học máy hoạt động ở quy mô lớn và trong môi trường phức tạp. TensorFlow [46, 47] sử dụng đồ thị luồng dữ liệu Dataflow để đại diện cho sự tính toán, chia sẻ trạng thái, và các hoạt động biến đổi trạng thái đó. Nó ánh xạ các nút của một đồ thị dataflow trên nhiều máy trong một cluster, và bên trong một máy trên nhiều thiết bị tính toán, bao gồm CPU, GPU đa lõi, các chíp ASIC tùy biến được gọi là tenxơ Processing Units (TPUs). Kiến trúc này rất linh hoạt cho phép cho các nhà phát triển ứng dụng: trong khi trước đây "tham số máy chủ" thiết kế quản lý chia sẻ trạng thái (shared state) được xây dựng sẵn trên hệ thống, TensorFlow cho phép các nhà phát triển để thử nghiệm các tối ưu hoá mới và các thuật toán huấn luyện.

TensorFlow hỗ trợ một loạt các ứng dụng, với sự hỗ trợ đặc biệt mạnh mẽ cho việc huấn luyện và suy luận trên các mạng học sâu Deep Learning [47]. Google đã phát hành TensorFlow như là một dự án mã nguồn mở, và nó đã trở thành sử dụng rộng rãi cho các nghiên cứu học máy.

Trong bài luận văn này, chúng tôi sử dụng TensorFlow để huấn luyện và tạo ra các mô hình đối thoại cho tiếng Việt, một kết quả rất khả quan khi sử dụng TensorFlow là chúng tôi đạt được những mô hình có chất lượng tốt.

4.4 Kết quả thực nghiệm

Do dữ liệu thu thập được khá lớn, chúng tôi đã chia làm 4 tập nhỏ mỗi tập 500,000 câu đối thoại phụ đề phim tiếng Việt. Và thử nghiệm trên 1 tập dữ liệu với các bộ tham số khác nhau, gồm:

- Profile 1:
    - 26,746 words
    - 74,870 QA
    - 5 Max length
    - Thời gian huấn luyện: 25 giờ
    - RAM: 4GB
    - CPU: 2 Core x4 @ 2.4GHz
- Profile 2:
    - 37,693 words
    - 180,700 QA
    - 7 Max length

    - Thời gian huấn luyện: 125 giờ
    - RAM: 4GB
    - CPU: 2 Core x4 @ 2.4GHz
- Profile 3:
    - 40,348 words
    - 320,419 QA
    - 10 Max length
    - Thời gian huấn luyện: 270 giờ
    - RAM: 4GB
    - CPU: 2 Core x4 @ 2.4GHz
- Profile 4:
    - 43,033 words
    - 381,474 QA
    - 12 Max length
    - Thời gian huấn luyện: 1,000 giờ
    - RAM: 4GB
    - CPU: 2 Core x4 @ 2.4GHz

Trong các lần thử nghiệm này chúng tôi đã huấn luyện mô hình mạng sử dụng hai lớp mạng LSTM (kích thước lớp ẩn 256) và kích thước embeding là 25.

Kết quả khi sử dụng huấn luyện với **profile 1**, tức là mẫu hỏi / câu trả lời có độ dài tối đa là 5. Và mạng này cho kết quả khá tốt !

Human: Xin chào **Machine: chào** **Human: Xin chào !** **Machine: xin chào !** **Human: Chào** **Machine: chào** **Human: Chào bạn** **Machine: này ward !** **Human: Anh đấy à ?** **Machine: ừ** **Human: Ồ vậy à** **Machine: được mà** **Human: Thật không ?** **Machine: thật chứ**	**Human: Tạm_biệt** **Machine: tạm_biệt** **Human: Tạm_biệt anh** **Machine: tạm_biệt** **Human: Tạm_biệt cô** **Machine: tạm_biệt các em** **Human: Tên tôi là Vũ** **Machine: chào mừng** **Human: Gặp anh sau nhé** **Machine: tạm_biệt** **Human: Gặp anh sau nhé !** **Machine: chúc may_mắn !** **Human: Cá có_thể bơi không ?** **Machine: không**

Human: Có tin được không ? **Machine: uh** **Human: Ai vậy ?** **Machine: anh st .** **Human: Anh đang ở đâu ?** **Machine: ở tư_gia**	**Human: Anh bao_nhiêu tuổi ?** **Machine: 20** **Human: Anh ấy bao_nhiêu tuổi ?** **Machine: 19** **Human: Cô ấy bao_nhiêu tuổi ?** **Machine: 19**

Một số trường hợp fail, kết quả chưa được tốt đối với profile 1.

Human: Hôm_nay là tháng mấy **Machine: okay** **Human: Anh là ai vậy** **Machine: em ổn rồi** **Human: Anh vẫn khỏe chứ ?** **Machine: chào anh gray** **Human: Con chó đâu ?** **Machine: thanh_tra montgomery** **Human: Tên anh là gì ?** **Machine: jonesy_._jonesy** **Human: Anh còn sống không ?** **Machine: kyle , tương_lai**	**Human: Ai là John ?** **Machine: tao làm_việc giải_thích** **Human: 2000** **Machine: miriam !** **Human: Tình_yêu là cái gì ?** **Machine: qua nhân_sự thùng** **Human: Cô ghét ai ?** **Machine: bài bông** **Human: Chim có_thể bay không ?** **Machine: quan_tâm đang buồn_cười** **Human: Ai có_thể hát được ?** **Machine: không có họ**

Các thử nghiệm khác với câu dài hơn so với input huấn luyện.

Kết quả khi sử dụng huấn luyện với **profile 2**, tức là mẫu hỏi / câu trả lời có độ dài tối đa là 7 với dữ liệu 37,693 words và 180,700 QA. Kết quả có một chút tiến bộ và khả quan hơn so với **profile 1**.

Human: Xin chào **Machine: chào** **Human: Xin chào !** **Machine: xin chào !** **Human: Chào** **Machine: chào** **Human: Chào bạn** **Machine: gracie đâu rồi ?** **Human: Anh đấy à ?** **Machine: đương_nhiên**	**Human: Tạm_biệt** **Machine: tạm_biệt** **Human: Tạm_biệt anh** **Machine: tạm_biệt** **Human: Tạm_biệt cô** **Machine: tạm_biệt** **Human: Tên tôi là Vũ** **Machine: um** **Human: Gặp anh sau nhé** **Machine: không có gì**

Human: Ồ vậy à **Machine: cám_ơn** **Human: Thật không ?** **Machine: thật** **Human: Có tin được không ?** **Machine: tốt thôi** **Human: Ai vậy ?** **Machine: là doug** **Human: Anh đang ở đâu ?** **Machine: nhà_ga okubo**	**Human: Gặp anh sau nhé !** **Machine: oh , tôi hy_vọng thế** **Human: Cá có_thể bơi không ?** **Machine: tôi không diễn_tả** **Human: Anh bao_nhiêu tuổi ?** **Machine: 20** **Human: Anh ấy bao_nhiêu tuổi ?** **Machine: 19** **Human: Cô ấy bao_nhiêu tuổi ?** **Machine: 19**

Thử nghiệm với các trường hợp fail trong **profile 1**. Chúng ta thấy ngay được kết quả đã có vẻ khả quan hơn nhiều và bám sát được ý nghĩa của câu hỏi hơn. Nhưng vẫn còn một số kết quả chưa được thỏa mãn.

Human: Hôm_nay là tháng mấy **Machine: bà không chảy_máu loạn tôi** **Human: Anh là ai vậy** **Machine: tôi biết** **Human: Anh vẫn khỏe chứ ?** **Machine: vẫn_khỏe** **Human: Con chó đâu ?** **Machine: con chó !** **Human: Tên anh là gì ?** **Machine: tên anh là gì ?** **Human: Anh còn sống không ?** **Machine: có**	**Human: Ai là John ?** **Machine: ai ?** **Human: 2000** **Machine: peter** **Human: Tình_yêu là cái gì ?** **Machine: năm .** **Human: Cô ghét ai ?** **Machine: không ?** **Human: Chim có_thể bay không ?** **Machine: thực_ra không phải là cu** **Human: Ai có_thể hát được ?** **Machine: tôi**

Kết quả khi sử dụng huấn luyện với **profile 3**, tức là mẫu hỏi / câu trả lời có độ dài tối đa là 10 với dữ liệu 40,348 words và 320,419 QA. Đúng như mong đợi, kết quả cho lần huấn luyện này đã đạt được rất tốt về mặt ngữ nghĩa. Các câu trả lời đã sát nghĩa hơn, bám sát với ý của câu hỏi, cùng với cặp hội thoại được đưa vào kiểm nghiệm với kết quả như sau:

Human: Xin chào **Machine: chào**	**Human: Tạm_biệt** **Machine: tạm_biệt**

```
Human: Xin chào !
Machine: xin chào !

Human: Chào
Machine: chào

Human: Chào bạn
Machine: chào

Human: Anh đấy à ?
Machine: yeah

Human: Ồ vậy à
Machine: phải

Human: Thật không ?
Machine: thật

Human: Có tin được không ?
Machine: không

Human: Ai vậy ?
Machine: tôi không biết

Human: Anh đang ở đâu ?
Machine: tôi không biết
```

```
Human: Tạm_biệt anh
Machine: tạm_biệt

Human: Tạm_biệt cô
Machine: tạm_biệt

Human: Tên tôi là Vũ
Machine: tôi là gwen

Human: Gặp anh sau nhé
Machine: tạm_biệt

Human: Gặp anh sau nhé !
Machine: chúc may_mắn !

Human: Cá có_thể bơi không ?
Machine: tôi không biết

Human: Anh bao_nhiêu tuổi ?
Machine: tôi không biết

Human: Anh ấy bao_nhiêu tuổi ?
Machine: 19

Human: Cô ấy bao_nhiêu tuổi ?
Machine: 19
```

Một số kết quả fail trong **profile 1**, đã có chất lượng được cải thiện đáng kể như bảng dưới đây, thay vì là phớt lờ thông tin thì nó đã trả lời một cách thẳng thắn hơn bằng câu „Tôi không biết“.

```
Human: Hôm_nay là tháng mấy
Machine: tôi không biết

Human: Anh là ai vậy
Machine: anh không biết

Human: Anh vẫn khỏe chứ ?
Machine: ừ

Human: Con chó đâu ?
Machine: mẹ cháu đâu ?

Human: Tên anh là gì ?
Machine: tôi không biết

Human: Anh còn sống không ?
Machine: không
```

```
Human: Ai là John ?
Machine: tôi không biết

Human: 2000
Machine: khoản của tôi

Human: Tình_yêu là cái gì ?
Machine: cháu không biết

Human: Cô ghét ai ?
Machine: tôi không biết

Human: Chim có_thể bay không ?
Machine: tôi không biết

Human: Ai có_thể hát được ?
Machine: tôi không biết
```

Một số kết quả khác, tức là câu trả lời có vẻ chưa làm thỏa mãn với câu hỏi của người hỏi, khi test với mạng này:

Human: Khi nào thì anh qua nhà tôi ? **Machine: không** **Human: Vậy_thì để mai nhé , có được được không ?** **Machine: không** **Human: Khi nào trời sáng , hãy gọi tôi nhé** **Machine: cảm_ơn** **Human: Về đến nhà hãy thông_báo cho mọi người biết nhé** **Machine: tôi không biết** **Human: Anh có ăn được món bánh mỳ không ?** **Machine: ừ**	**Human: Năm nào ?** **Machine: tôi không biết** **Human: Anh bao_nhiêu tuổi ?** **Machine: tôi không biết** **Human: Tên anh là gì ?** **Machine: tôi không biết** **Human: Khi nào thì anh đi vậy** **Machine: anh không biết**

Như vậy, chúng ta đã nhận thấy rằng mạng đã có thể trả lời những gì đã được học với chất lượng rất khả quan, phụ thuộc vào chất lượng của bộ dữ liệu hỏi đáp. Với nhưng câu hỏi khó, chưa được học mạng đã có nhưng câu trả lời từ chối „**Tôi không biết**“.

Với mô hình thử nghiệm Profile 4, thời gian huấn luyện khá lâu với bộ dữ liệu lớn. Nên chúng tôi chưa có kết quả cập nhật trong luận văn này. Nhưng rõ ràng, là chúng ta đã thấy được tín hiệu cải thiện chất lượng đáng kể. Mô hình ngày càng trở nên trung thực hơn, bám sát ngữ nghĩa hơn!

KẾT LUẬN

Luận văn này đã đưa ra 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 kho phụ đề mã mở OpenSubtitles2016 [1]. 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.

Từ kết quả thực nghiệm của luận văn này, chúng tôi có một số nhận xét: Với các chuỗi câu dài thì mạng huấn luyện mất nhiều thời gian hơn. Sau khoảng 300,000 lần lặp với độ dài 10 từ thì mạng vẫn cung cấp những câu trả lời lảng tránh, phớt lờ câu hỏi (bằng việc trả lời bằng các câu **"Tôi không biết"**, nhưng nó đã hiểu và cần tích hợp một số ngữ nghĩa cơ bản. Bằng việc thay đổi mô hình bằng cách điều chỉnh độ dài của mạng hoặc tối ưu cục bộ các cặp câu hỏi-đáp thì cho kết quả với chất lượng tốt hơn rất nhiều, bám sát ngữ nghĩa hơn.

Qua những kết quả đạt được ban đầu, chúng nhận thấy còn rất nhiều việc phải làm, cần phải tối ưu. Nhưng cách tiếp cận này ban đầu đã cho những kết quả rất tích cực và đúng đắn, có thể giải quyết được những vấn đề ngữ nghĩa, ngữ cảnh và tính cách trong hệ thống đối thoại.

Định hướng nghiên cứu tiếp theo, chúng tôi tiếp tục làm mượt dữ liệu, để tạo ra các mô hình mới có khả năng trả lời sát với ngữ cảnh, đạt chất lượng cao hơn, giảm khả năng lảng tránh và đưa tính cá nhân vào trong đoạn hội thoại.

TÀI LIỆU THAM KHẢO

1. Pierre Lison and Jörg Tiedemann, 2016, "OpenSubtitles2016: Extracting Large Parallel Corpora from Movie and TV Subtitles". In Proceedings of the 10th International Conference on Language Resources and Evaluation (LREC 2016)
2. Ryan Lowe, Nissan Pow, Iulian Serban, Joelle Pineau, 4 Feb 2016. "The Ubuntu Dialogue Corpus: A Large Dataset for Research in Unstructured Multi-Turn Dialogue Systems".
3. Iulian V. Serban, Alessandro Sordoni, Yoshua Bengio, Aaron Courville, Joelle Pineau, 6 Apr 2016. "Building End-To-End Dialogue Systems Using Generative Hierarchical Neural Network Models".
4. Wojciech Zaremba, Ilya Sutskever, Oriol Vinyals, 19 Feb 2015. "Recurrent Neural Network Regularization".
5. Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, Yoshua Bengio, Sep 2014. "Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation".
6. Oriol Vinyals, Quoc Le, 22 Jul 2015. "A Neural Conversational Model".
7. Ilya Sutskever, Oriol Vinyals, Quoc V. Le, 14 Dec 2014. "Sequence to Sequence Learning with Neural Networks" pp. 1–9.
8. Lifeng Shang, Zhengdong Lu, Hang Li, 27 Apr 2015. "Neural Responding Machine for Short-Text Conversation".
9. Alessandro Sordoni, Michel Galley, Michael Auli, Chris Brockett, Yangfeng Ji, Margaret Mitchell, Jian-Yun Nie, Jianfeng Gao, Bill Dolan, 22 Jun 2015. "A Neural Network Approach to Context-Sensitive Generation of Conversational Responses".
10. Jiwei Li, Michel Galley, Chris Brockett, Georgios P. Spithourakis, Jianfeng Gao, Bill Dolan, 8 Jun 2016. "A Persona-Based Neural Conversation Model".
11. Lưu Tuấn Anh, Yamamoto Kazuhide, 16 Feb 2013. "Pointwise for Vietnamese Word Segmentation".
12. S. Hochreiter and J. Schmidhuber, 1997. "Long Short-Term Memory" Neural Computation, vol. 9, pp. 1735–1780.

13. S. Sukhbaatar, A. Szlam, J. Weston, and R. Fergus, 31 Mar 2015. “End-To-End Memory Networks” pp. 1–11.
14. Christopher Olah, 27 Aug 2015. “Understanding LSTM Networks”.
15. Kaisheng Yao, Geoffrey Zweig, Baolin Peng, 29 Oct 2015. “Attention with Intention for a Neural Network Conversation Model”.
16. Jacob Andreas, Marcus Rohrbach, Trevor Darrell, Dan Klein, 7 Jan 2016. “Learning to Compose Neural Networks for Question Answering”.
17. Young, M. Gasic, B. Thomson, and J. D. Williams, 2013. “POMDP-based statistical spoken dialog systems: A review. Proceedings of the IEEE”, 101(5):1160–1179.
18. Williams, A. Raux, D. Ramachandran, and A. Black. The dialog state tracking challenge. In Special Interest Group on Discourse and Dialogue (SIGDIAL), 2013.
19. S. Kim, L. F. DHaro, R. E. Banchs, J. Williams, and M. Henderson. Dialog state tracking challenge 4. 2015.
20. Wen, M. Gasic, D. Kim, N. Mrksic, P. Su, D. Vandyke, and S. Young. Stochastic language generation in dialogue using recurrent neural networks with convolutional sentence reranking. Special Interest Group on Discourse and Dialogue (SIGDIAL), 2015.
21. Russell, S., Dewey, D., Tegmark, M. (2015). “Research Priorities for Robust and Beneficial Artificial Intelligence”. AI Magazine, 36 (4):105–114.
22. Walter S. Lasecki, Ece Kamar, Dan Bohus, January 2013. “Conversations in the Crowd: Collecting Data for Task-Oriented Dialog Learning”, pp1-10.
23. Rami Al-Rfou, Marc Pickett, Javier Snaider, Yun-hsuan Sung, Brian Strope, Ray Kurzweil, 1 Jun 2016. “Conversational Contextual Cues: The Case of Personalization and History for Response Ranking”, p1-10.
24. Alan M Turing. 1950. “Computing machinery and intelligence”. *Mind*, 59(236):433–460.
25. Joseph Weizenbaum. 1966. “Elizaa computer program for the study of natural language communication between man and machine”. Communications of the ACM, 9(1):36–45.
26. Roger C Parkinson, Kenneth Mark Colby, and William S Faught. 1977. “Conversational language comprehension using integrated pattern-matching and parsing”. Artificial Intelligence, 9(2):111–134.

27. Richard S Wallace. 2009. “The anatomy of ALICE”. Springer.
28. Jurgen Schmidhuber. 2015. “Deep learning in neural networks: An overview. Neural Networks”, 61:85–117.
29. Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. Nature, 521(7553):436–444.
30. Alan Ritter, Colin Cherry, and Bill Dolan. 2010. “Unsupervised modeling of twitter conversations”. In Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics, HLT ’10, pages 172–180, Stroudsburg, PA, USA. Association for Computational Linguistics.
31. Rafael E. Banchs and Haizhou Li. 2012. “Iris: a chat-oriented dialogue system based on the vector space model”. In Proceedings of the ACL 2012 System Demonstrations, pages 37–42, Jeju Island, Korea, July. Association for Computational Linguistics.
32. Karthik Narasimhan, Tejas Kulkarni, and Regina Barzilay. 2015. “Language understanding for text-based games using deep reinforcement learning”. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pages 1–11, Lisbon, Portugal, September. Association for Computational Linguistics.
33. T.-H. Wen, D. Vandyke, N. Mrksic, M. Gasic, L. M. Rojas-Barahona, P.-H. Su, S. Ultes, and S. Young. 2016. A Network-based End-to-End Trainable Task-oriented Dialogue System. ArXiv eprints, April.
34. Heriberto Cuayahuitl. 2016. Simpleds: “A simple deep reinforcement learning dialogue system”. CoRR, abs/1601.04574.
35. Marilyn Walker, Grace Lin, and Jennifer Sawyer. 2012. “An annotated corpus of film dialogue for learning and characterizing character style”. In Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Mehmet Ugur Do ˘ gan, Bente Maegaard, Joseph ˘ Mariani, Jan Odijk, and Stelios Piperidis, editors, Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC-2012), pages 1373–1378, Istanbul, Turkey, May. European Language Resources Association (ELRA). ACL Anthology Identifier: L12-1657.
36. Francesca Bonin, Jose San Pedro, and Nuria Oliver. 2014. “A context-aware nlp approach for noteworthiness detection in cellphone conversations”. In COLING, pages 25–36.

37. Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2015. A diversitypromoting objective function for neural conversation models. arXiv preprint arXiv:1510.03055.
38. Michel Galley, Chris Brockett, Alessandro Sordoni, Yangfeng Ji, Michael Auli, Chris Quirk, Margaret Mitchell, Jianfeng Gao, and Bill Dolan. 2015. “deltableu: A discriminative metric for generation tasks with intrinsically diverse targets”. In Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers), pages 445–450, Beijing, China, July. Association for Computational Linguistics.
39. Lifeng Shang, Zhengdong Lu, and Hang Li. 2015. “Neural responding machine for shorttext conversation”. arXiv preprint arXiv:1503.02364.
40. T. Mikolov, M. Karafiát, L. Burget, J. Cernockỳ, and S. Khudanpur, 2010. “Recurrent neural network based language model”. In INTERSPEECH, pages 1045–1048.
41. M. Sundermeyer, R. Schluter, and H. Ney, 2010. “LSTM neural networks for language modeling”. In INTERSPEECH.
42. P. Werbos, 1990. “Backpropagation through time: what it does and how to do it”. Proceedings of IEEE.
43. D. Rumelhart, G. E. Hinton, and R. J. Williams, 1986. “Learning representations by back-propagating errors”. Nature, 323(6088):533–536.
44. Y. Bengio, P. Simard, and P. Frasconi, 1994. “Learning long-term dependencies with gradient descent is difficult”. IEEE Transactions on Neural Networks, 5(2):157–166.
45. A. Graves, 5 Jun 2014. “Generating sequences with recurrent neural networks”. In Arxiv preprint arXiv:1308.0850.
46. Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, …, 31 May 2016. “TensorFlow: A system for large-scale machine learning”. In Arxiv preprint arXiv:1605.08695.
47. Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, …, 16 Mar 2016. “TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems”. In Arxiv preprint arXiv:1603.04467
48. Andrej Karpathy, Li Fei-Fei, 2015. “Deep Visual-Semantic Alignments for Generating Image Descriptions”.

49. Lester, J., Branting, K., and Mott, B, 2004. “Conversational agents. In Handbook of Internet Computing. Chapman & Hall”.
50. Will, T, 2007. “Creating a Dynamic Speech Dialogue”. VDM Verlag Dr.
51. Tomas Mikolov, Kai Chen, Greg Corrado, Jeffrey Dean, 7 Sep 2013. “Efficient Estimation of Word Representations in Vector Space”. In Arxiv preprint arXiv:1301.378.