TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**ĐỒ ÁN CUỐI KÌ**

**XỬ LÍ NGÔN NGỮ TỰ NHIÊN**

*Người hướng dẫn*: **GV. LÊ ANH CƯỜNG**

*Người thực hiện*: **THÁI NGỌC PHÚ – 518H0245**

**NGUYỄN THỊ LAN ANH – 518H0470**

Lớp **: 18H50204**

Khoá  **: 22**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2021**

TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**ĐỒ ÁN CUỐI KÌ**

**XỬ LÍ NGÔN NGỮ TỰ NHIÊN**

*Người hướng dẫn*:  **GV. LÊ ANH CƯỜNG**

*Người thực hiện*: **THÁI NGỌC PHÚ – 518H0245**

**NGUYỄN THỊ LAN ANH – 518H0407**

Lớp **: 18H50204**

Khoá  **: 22**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2021**

LỜI CẢM ƠN

Chúng em xin cảm ơn thầy LÊ AH CƯỜNG đã giảng dạy nhiệt tình và cung cấp các tài liệu cần thiết để chúng em có thể hoàn thành đồ án này, chúng em xin cảm ơn Khoa CNTT đã tạo điều kiện mở lớp để chúng em có thể tham gia và hoàn thành lớp học này.

**ĐỒ ÁN ĐƯỢC HOÀN THÀNH**

**TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

Tôi xin cam đoan đây là sản phẩm đồ án của riêng tôi / chúng tôi và được sự hướng dẫn của TS BÙI THANH HÙNG;. Các nội dung nghiên cứu, kết quả trong đề tài này là trung thực và chưa công bố dưới bất kỳ hình thức nào trước đây. Những số liệu trong các bảng biểu phục vụ cho việc phân tích, nhận xét, đánh giá được chính tác giả thu thập từ các nguồn khác nhau có ghi rõ trong phần tài liệu tham khảo.

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

**Nếu phát hiện có bất kỳ sự gian lận nào tôi xin hoàn toàn chịu trách nhiệm về nội dung đồ án của mình.** Trường đại học Tôn Đức Thắng không liên quan đến những vi phạm tác quyền, bản quyền do tôi gây ra trong quá trình thực hiện (nếu có).

*TP. Hồ Chí Minh, ngày tháng năm*

*Tác giả*

*(ký tên và ghi rõ họ tên)*

*Phạm Hồng Toàn*

*Nguyễn Thị Lan Anh*

PHẦN ĐÁNH GIÁ CỦA GIẢNG VIÊN

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Tp. Hồ Chí Minh, ngày tháng năm

(kí và ghi họ tên)

TÓM TẮT

Tạo dữ liệu các giọng vùng miền là file wav ,sau đó chuyển đổi các file wav thành file ảnh và dùng mô hình CNN để huấn luyện nhận biết các giọng vùng miền

Sau khi train và test đã đạt được kết quả chính xác là trên 70% và dự đoán đúng 9/12 file

**Chatbot là gì?**

Khi truy cập vào một fanpage trên Facebook, bạn gửi tin nhắn tìm hiểu thông tin và nhận được lời hồi đáp ngay sau đó. Những tin nhắn như vậy có thể không đến từ người quản lý fanpage mà đến từ chatbot. Chatbot là một hình thức trò chuyện được các fanpage ưa chuộng sử dụng bởi tốc độ hồi đáp nhanh, với những thông tin chính xác được lập trình từ trước. Điều này mang đến trải nghiệm tuyệt vời hơn cho người truy cập fanpage khi không phải chờ đợi quá lâu để nhận được câu trả lời mong muốn.

Chatbot, hay còn được gọi là talkbot, chatterbot, bot, bot IM, hay thực thể trò chuyện tự động, là một chương trình máy tính thực hiện một cuộc trò chuyện thông qua việc tiếp nhận thông tin bằng văn bản và giọng nói. Chatbot được thiết kế để mô phỏng các cuộc hội thoại của con người, diễn giải, xử lý yêu cầu của người dùng và cuối cùng đưa ra câu trả lời một cách nhanh chóng. Chatbot được sử dụng trong các hệ thống hội thoại của các fanpage cho nhiều mục đích cụ thể, phổ biến nhất là dịch vụ khách hàng và thu thập thông tin.

**Phân loại chatbot**

Chatbot là một hệ thống dựa vào các kỹ thuật về xử lý ngôn ngữ tự nhiên, vậy nên giống với hầu hết các bài toán về xử lý ngôn ngữ khác, các hệ thống Chatbot có thể được chia làm 2 loại

1. Hệ thống Chatbot dựa vào các quy luật, thói quen trong ngôn ngữ của người dùng **(Rule-based chatbots)**
2. Hệ thống Chatbot xây dựng trên một kho dữ liệu hội thoại cho trước **(Corpus-based chatbots)** - Kho văn bản này có thể được thu thập bằng lượng lớn dữ liệu từ các cuộc nói chuyện của người dùng, sử dụng các phương pháp trích xuất thông tin (Infomation Retrieval) hoặc các phương pháp máy học để tạo ra câu trả lời dựa vào ngữ cảnh trò chuyện với người dùng.

## Corpus-based chatbots

Thay vì việc sử dụng những luật xây dựng bằng tay như rule-based chatbots, hệ thống chatbot dựa trên kho dữ liệu hội thoại trong thực tế của người với để tìm ra được những phản hồi phù hợp cho hoàn cảnh của mình. Những dữ liệu hội thoại này có thể được thu thập trực tiếp trên một số platform trò chuyện hoặc lấy ra từ các lời thoại của nhân vật trong các bộ phim với nhau.

Corpus-based hiện nay thường sẽ có 2 kiểu hoạt động chính: Trích xuất thông tin cần thiết (Information Retrieval) và áp dụng các bài toán Deep learning theo dạng sequence to sequence (tương tự như Dịch tự động)

### Information Retrieval based chatbots (IR-based)

Người viết tập trung vào các chatbot được sử dụng trong miền ứng dụng đóng (closed domain) và trả lời theo mô hình truy xuất thông tin (retrieval-based model). Mô hình truy xuất thông tin là mô hình trong đó, chatbot đưa ra những phản hồi được chuẩn bị trước hoặc tuân theo những mô thức nhất định. Mô hình này khác với mô hình tự động sinh câu trả lời (generative), trong đó câu trả lời của chatbot được tự động sinh ra bằng việc học từ một tập dữ liệu các đoạn hội thoại (tham khảo tài liệu). Các hệ thống chatbot được triển khai trong thực tế phần lớn tuân theo mô hình truy xuất thông tin và được áp dụng trong những miền ứng dụng nhất định.

Cơ chế hoạt động của một hệ thống IR-based có bước đầu tiên là tìm trong cơ sở dữ liệu hội thoại, một câu nói có độ tương tự cao nhất với câu nói hiện tại của user. Từ đó có 2 cách để đưa ra câu trả lời:

1. Dùng chính câu nói đó

response=most\_similar(q)response = most\\_similar (q) *response*=*most*\_*similar*(*q*)

1. Dùng câu trả lời cho câu nói đó

response=process(most\_similar(q))response = process (most\\_similar (q)) *response*=*process*(*most*\_*similar*(*q*))

Trong thực tế, trong mỗi bài toán, chúng ta lại có cách tính độ tương tự câu nói khác nhau, nhưng cách thông thường nhất vẫn là sử dụng **thuật toán TF-IDF** để chuyển về dạng vector số thực và tính toán độ tương tự dựa vào **khoảng cách cosine**.

Mặc dù nhìn qua, cách đưa ra câu trả lời sau khi đã xử lý câu hội thoại tương tự được lấy ra trong cơ sở dữ liệu có vẻ là cách hợp lý hơn, tuy nhiên trong thực tế thì không phải vậy. Việc thêm một bước xử lý một cách gián tiếp (xử lý trên câu tương tự với câu nói của người dùng) sẽ khiến kết quả có khả năng bị nhiễu khá lớn.

**Các bước xây dựng**

**1. Xác định ý định người dùng**

Thông thường, người dùng thường truy cập hệ thống chatbot với mong muốn hệ thống sẽ đưa ra những hành động trợ giúp mình về một vấn đề nào đó. Ví dụ, người dùng của hệ thống chatbot hỗ trợ đặt vé máy bay có thể đưa ra yêu cầu đặt vé của mình khi bắt đầu cuộc hội thoại. Để đưa ra hỗ trợ được chính xác, chatbot cần xác định được ý định (intent) đó của người dùng. Việc xác định ý định của người dùng sẽ quyết định hội thoại tiếp theo giữa người và chatbot sẽ diễn ra như thế nào. Vì thế, nếu xác định sai ý định người dùng, chatbot sẽ đưa ra những phản hồi không đúng, không hợp ngữ cảnh. Khi đó, người dùng có thể thấy chán ghét và không quay lại sử dụng hệ thống. Bài toán xác định ý định người dùng vì thế đóng vai trò rất quan trọng trong hệ thống chatbot.

Đối với miền ứng dụng đóng, chúng ta có thể giới hạn rằng số lượng ý định của người dùng nằm trong một tập hữu hạn những intent đã được định nghĩa sẵn, có liên quan đến những nghiệp vụ doanh nghiệp mà chatbot có thể hỗ trợ. Với giới hạn này, bài toán xác định ý định người dùng có thể quy về bài toán phân lớp văn bản. Với đầu vào là một câu giao tiếp của người dùng, hệ thống phân lớp sẽ xác định intent tương ứng với câu đó trong tập các intent đã được định nghĩa.

Để xây dựng một mô hình phân lớp intent, chúng ta cần một tập dữ liệu huấn luyện bao gồm các cách diễn đạt khác nhau cho mỗi intent. Ví dụ, cùng một mục đích hỏi về thời tiết ở Hà Nội trong ngày hôm nay, người dùng có thể dùng những cách diễn đạt sau:

* *Thời tiết hôm nay ở Hà Nội thế nào ad?*
* *Hà Nội hôm nay có mưa không vậy?*
* *Hà Nội hôm nay bao nhiêu độ vậy?*
* *Cho mình hỏi, ra ngoài đường hôm nay có phải mang áo mưa không?*

Có thể nói, bước tạo dữ liệu huấn luyện cho bài toán phân lớp intent là một trong những công việc quan trọng nhất khi phát triển hệ thống chatbot và ảnh hưởng lớn tới chất lượng sản phẩm của hệ thống chatbot về sau. Công việc này cũng đòi hỏi thời gian, công sức khá lớn của nhà phát triển chatbot.

**Mô hình học máy cho bài toán phân lớp ý định người dùng**

Khi đã có dữ liệu huấn luyện cho bài toán phân lớp intent, chúng ta sẽ mô hình bài toán thành bài toán phân lớp văn bản. Bài toán phân lớp văn bản (text categorization) là một bài toán kinh điển trong ngành NLP và khai phá văn bản (Text Mining). Mô hình phân lớp văn bản cho bài toán phân lớp intent được phát biểu một cách hình thức như sau:

Chúng ta được cho trước một tập huấn luyện bao gồm các cặp (câu hội thoại, intent), D = {(x(1), y(1)),…, (x(n), y(n))}, trong đó x(i) là các câu hội thoại và y(i) là intent tương ứng cho x(i). Các intent y(i) nằm trong một tập hữu hạn *Κ* các intent được định nghĩa trước. Chúng ta cần học từ tập huấn luyện này một mô hình phân lớp Θ có chức năng phân lớp một câu hội thoại mới vào một trong các intent thuộc tập *K*. Kiến trúc của hệ thống phân lớp intent được minh hoạ trong hình 1.

Hình 1. Kiến trúc của hệ thống phân lớp intent

Hệ thống phân lớp intent có một số thành phần cơ bản:

* Tiền xử lý dữ liệu
* Trích xuất đặc trưng
* Huấn luyện mô hình
* Phân lớp

Trong bước tiền xử lý dữ liệu, chúng ta sẽ thực hiện các thao tác “làm sạch” dữ liệu như: loại bỏ các thông tin dư thừa, chuẩn hoá dữ liệu như chuyển các từ viết sai chính tả thành đúng chính tả, chuẩn hoá các từ viết tắt,… Việc tiền xử lý dữ liệu có vai trò quan trọng trong hệ thống chatbot do đặc thù của ngôn ngữ chat, nói: viết tắt, sai chính tả, hay dùng “teencode”.

Sau khi tiền xử lý dữ liệu và thu được dữ liệu đã được làm sạch, chúng ta sẽ trích xuất những đặc trưng từ dữ liệu này. Trong học máy, bước này được gọi là trích xuất đặc trưng (feature extraction hay feature engineering). Trong mô hình học máy truyền thống (trước khi mô hình học sâu được áp dụng rộng rãi), bước trích xuất đặc trưng ảnh hưởng lớn đến độ chính xác của mô hình phân lớp. Để trích xuất được những đặc trưng tốt, chúng ta cần phân tích dữ liệu khá tỉ mỉ và cần cả những tri thức chuyên gia trong từng miền ứng dụng cụ thể.

Bước huấn luyện mô hình nhận đầu vào là các đặc trưng đã được trích xuất và áp dụng các thuật toán học máy để học ra một mô hình phân lớp. Các mô hình phân lớp có thể là các luật phân lớp (nếu sử dụng decision tree) hoặc là các vector trọng số tương ứng với các đặc trưng được trích xuất (như trong các mô hình logistic regression, SVM, hay mạng Neural).

Sau khi có một mô hình phân lớp intent, chúng ta có thể sử dụng nó để phân lớp một câu hội thoại mới. Câu hội thoại này cũng đi qua các bước tiền xử lý và trích xuất đặc trưng, sau đó mô hình phân lớp sẽ xác định “điểm số” cho từng intent trong tập các intent và đưa ra intent có điểm cao nhất.

**Mô hình dựa trên đối sánh nội dung**

Mô hình phân lớp intent dựa trên học máy thống kê yêu cầu dữ liệu huấn luyện bao gồm các cách diễn đạt khác nhau cho mỗi intent. Dữ liệu huấn luyện này thông thường được chuẩn bị một cách thủ công. Bước chuẩn bị dữ liệu này tốn khá nhiều thời gian và công sức, đặc biệt là trong các ứng dụng mà số lượng intent tương đối lớn.

Một cách tiếp cận có thể giảm công sức chuẩn bị dữ liệu huấn luyện là sử dụng mô hình đối sánh thông tin (content matching). Trong cách tiếp cận này, chúng ta vẫn cần chuẩn bị dữ liệu, trong đó mỗi intent có ít nhất một câu hỏi tương ứng. Với một câu hội thoại cho trước, chúng ta sẽ áp dụng thuật toán đối sánh thông tin để đối sánh với từng câu hỏi trong tập dữ liệu. Câu trả lời cho câu hỏi có nội dung gần nhất với câu đầu vào sẽ được đưa ra. Trong ứng dụng thực tế, chúng ta có thể đưa ra danh sách (ví dụ top 3) các câu gần nhất để người dùng lựa chọn.

Phương pháp đối sánh thông tin khá thích hợp cho các hệ thống chatbot phục vụ cho công việc trả lời các câu hỏi thường gặp (FAQ).  Chúng ta có thể tận dụng dữ liệu FAQ sẵn có để tạo ngay một FAQ chatbot bằng phương pháp đối sánh nội dung mà không cần tạo dữ liệu huấn luyện như trong mô hình học máy thống kê.

Một trong những thách thức của mô hình đối sánh nội dung là việc xử lý những cách diễn đạt khác nhau cho cùng một câu hỏi yêu cầu những luật phải chuẩn bị bằng tay. Vì số lượng mẫu cho mỗi intent không nhiều, mô hình đối sánh sẽ phải sử dụng các luật hoặc tài nguyên ngữ nghĩa để xử lý những biến thể khác nhau khi diễn đạt một từ, một cụm từ, hay một ý trong câu. Các câu 1) và 2) trong ví dụ dưới đây sử dụng các cách diễn đạt khác nhau cho cùng một câu hỏi của khách hàng của một công ty viễn thông về tình trạng mạng chậm.

1. Ad ơi, sao mạng nhà em dạo này chậm thế?
2. Mạng lag lắm, dùng rất ức chế.

Trong ví dụ ở trên, nếu dùng mô hình đối sánh thông tin, hệ thống cần nhận ra từ “chậm” và “lag” (ngôn ngữ được dùng trên mạng) mang cùng ý nghĩa.

Hiện nay các tài nguyên ngữ nghĩa cho xử lý ngôn ngữ tiếng Việt còn chưa đầy đủ, nên cách tiếp cận dựa trên mô hình học máy thống kê hoặc mô hình lai (hybrid model) – kết hợp cả học máy thống kê và đối sánh thông tin có thể thích hợp hơn cho các chatbot tiếng Việt.

**2. Trích xuất thông tin**

Bên cạnh việc xác định intent trong câu hội thoại của người dùng, chúng ta cần trích xuất các thông tin cần thiết trong đó. Các thông tin cần trích xuất trong một câu hội thoại thường là các thực thể thuộc về một loại nào đó. Ví dụ, khi một khách hàng muốn đặt vé máy bay, hệ thống cần biết địa điểm xuất phát và địa điểm khách muốn đến, ngày giờ khách hàng muốn bay,…Thành phần NLU của các hệ thống chatbot thường hỗ trợ các loại thực thể sau (tham khảo tài liệu):

* Vị trí (Location)
* Thời gian (Datetime)
* Số (Number)
* Địa chỉ liên lạc (Contact)
* Khoảng cách (Distance)
* Khoảng thời gian (Duration)

Hình 2: Gán nhãn từ theo mô hình B-I-O trong trích xuất thông tin

Đầu vào của một module trích xuất thông tin là một câu hội thoại. Module trích xuất thông tin cần xác định vị trí của các thực thể trong câu (vị trí bắt đầu vàvị trí kết thúc của thực thể). Ví dụ sau minh hoạ một câu hội thoại và các thực thể được trích xuất từ đó.

* **Câu hội thoại**: Tôi muốn đặt vé máy bay đi Phú Quốc từ sân bay Nội Bài lúc 8 giờ tối ngày mai.
* **Câu có các thực thể được xác định**: Tôi muốn đặt vé máy bay đi [Phú Quốc]LOCATION từ sân bay [Nội Bài]LOCATION lúc [8 giờ tối ngày mai]TIME.

Trong câu trên có 3 thực thể (nằm trong các dấu [ ]) với các loại thực thể tương ứng (được viết với font chữ nhỏ hơn ở dưới).

Cách tiếp cận phổ biến cho bài toán trích xuất thông tin là mô hình hoá bài toán thành bài toán gán nhãn chuỗi (sequence labeling). Đầu vào của bài toán gán nhãn chuỗi là một dãy các từ, và đầu ra là một dãy các nhãn tương ứng các các từ trong đầu vào. Chúng ta sẽ sử dụng các mô hình học máy để học một mô hình gán nhãn từ một tập dữ liệu đầu vào bao gồm các cặp (x1…xn, y1…yn), trong đó x1…xn là dãy các từ, y1…ynlà dãy các nhãn. Độ dài của các dãy từ trong tập dữ liệu có thể khác nhau.

Trong bài toán trích xuất thông tin, tập nhãn cho các từ trong câu đầu vào thường được tạo ra theo  mô hình BIO, với B là viết tắt của “Beginning”, I là viết tắt của “Inside”, và O là viết tắt của “Outside”. Khi biết vị trí từ bắt đầu của một thực thể và các từ nằm trong thực thể đó, chúng ta có thể xác định vị trí của thực thể trong câu. Trong ví dụ ở trên, dãy các nhãn tương ứng với dãy của các từ trong câu hội thoại đầu vào được minh hoạ ở hình 2.

Thuật toán huấn luyện mô hình gán nhãn chuỗi phổ biến là mô hình Markov ẩn (HMM – Hidden Markov Models), mô hình CRF (Conditional Random Fields) . Với dữ liệu văn bản, mô hình CRF thường cho kết quả tốt hơn mô hình HMM. Có khá nhiều các công cụ mã nguồn mở cài đặt mô hình CRF cho bài toán gán nhãn chuỗi như CRF++ , CRF Suite , Mallet ,…

Gần đây, các mô hình mạng neural hồi quy (Recurrent Neural Networks) được áp dụng khá nhiều cho bài toán gán nhãn chuỗi. Mô hình mạng neural hồi quy tỏ ra hiệu quả với dữ liệu dạng chuỗi vì nó mô hình mối quan hệ phụ thuộc giữa các từ trong câu. Ví dụ, mạng neural hồi quy được áp dụng cho bài toán gán nhán từ loại (POS Tagging), bài toán xác định thực thể tên gọi

**3. Quản lý hội thoại**

Trong các phiên trao đổi dài (long conversation) giữa người và chatbot, chatbot sẽ cần ghi nhớ những thông tin về ngữ cảnh (context) hay quản lý các trạng thái hội thoại (dialog state). Vấn đề quản lý hội thoại (dialoge management) khi đó là quan trọng để đảm bảo việc trao đổi giữa người và máy là thông suốt.

Chức năng của thành phần quản lý hội thoại là nhận đầu vào từ thành phần NLU, quản lý các trạng thái hội thoại (dialogue state), ngữ cảnh hội thoại (dialogue context), và truyền đầu ra cho thành phần sinh ngôn ngữ (Natural Language Generation, viết tắt là NLG). Ví dụ module quản lý dialogue trong một chatbot phục vụ đặt vé máy bay cần biết khi nào người dùng đã cung cấp đủ thông tin cho việc đặt vé để tạo một ticket tới hệ thống hoặc khi nào cần phải xác nhận lại thông tin do người dùng đưa vào. Hiện nay, các sản phẩm chatbot thường dùng mô hình máy trạng thái hữu hạn (Finite State Automata – FSA), mô hình Frame-based (Slot Filling), hoặc kết hợp hai mô hình này.

Hình 3: Minh hoạ quản lý hội thoại theo mô hình máy trạng thái hữu hạn FSA

FSA là mô hình quản lý hội thoại đơn giản nhất. Ví dụ, hãy tưởng tượng một hệ thống chăm sóc khách hàng của một công ty viễn thông, phục vụ cho những khách hàng than phiền về vấn đề mạng chậm. Nhiệm vụ của chatbot là hỏi tên khách hàng, số điện thoại, tên gói Internet khách hàng đang dùng, tốc độ Internet thực tế của khách hàng.  Hình vẽ 3 minh hoạ một mô hình quản lý hội thoại cho chatbot chăm sóc khách hàng. Các trạng thái của FSA tương ứng với các câu hỏi mà dialogue manager hỏi người dùng. Các cung nối giữa các trạng thái tương ứng với các hành động của chatbot sẽ thực hiện. Các hành động này phụ thuộc phản hồi của người dùng cho các câu hỏi. Trong mô hình FSA, chatbot là phía định hướng người sử dụng trong cuộc hội thoại.

Ưu điểm của mô hình FSA là đơn giản và chatbot sẽ định trước dạng câu trả lời mong muốn từ phía người dùng. Tuy nhiên, mô hình FSA không thực sự phù hợp cho các hệ thống chatbot phức tạp hoặc khi người dùng đưa ra nhiều thông tin khác nhau trong cùng một câu hội thoại. Trong ví dụ chatbot ở trên, khi người dùng đồng thời cung cấp cả tên và số điện thoại, nếu chatbot tiếp tục hỏi số điện thoại, người dùng có thể cảm thấy khó chịu.

Mô hình Frame-based (hoặc tên khác là Form-based) có thể giải quyết vấn đề mà mô hình FSA gặp phải. Mô hình Frame-based dựa trên các frame định sẵn để định hướng cuộc hội thoại. Mỗi frame sẽ bao gồm các thông tin (slot) cần điền và các câu hỏi tương ứng mà dialogue manager hỏi người dùng. Mô hình này cho phép người dùng điền thông tin vào nhiều slot khác nhau trong frame. Hình vẽ 4 là một ví dụ về một frame cho chatbot ở trên.

Hình 4: Frame cho chatbot hỏi thông tin khách hàng cho tình huống mạng chậm

Thành phần quản lý dialogue theo mô hình Frame-based sẽ đưa ra câu hỏi cho khách hàng, điền thông tin vào các slot dựa trên thông tin khách hàng cung cấp cho đến khi có đủ thông tin cần thiết. Khi người dùng trả lời nhiều câu hỏi cùng lúc, hệ thống sẽ phải điền vào các slot tương ứng và ghi nhớ để không hỏi lại những câu hỏi đã có câu trả lời.

Trong các miền ứng dụng phức tạp, một cuộc hội thoại có thể có nhiều frame khác nhau. Vấn đề đặt ra cho người phát triển chatbot khi đó là làm sao để biết khi nào cần chuyển đổi giữa các frame. Cách tiếp cận thường dùng để quản lý việc chuyển điều khiển giữa các frame là định nghĩa các luật (production rule). Các luật này dựa trên một số các thành tố như câu hội thoại hoặc câu hỏi gần nhất mà người dùng đưa ra.

**Những thách thức**

Mặc dù lĩnh vực NLP và Machine Learning đã và đang có rất nhiều bước tiến mạnh mẽ, trong phát triển chatbot, vẫn còn tồn tại nhiều thách thức mà các nhà nghiên cứu xử lý ngôn ngữ tự nhiên và học máy cần vượt qua. Chúng tôi liệt kê hai vấn đề ở dưới đây.

Vấn đề thứ nhất là vấn đề đồng tham chiếu (coreference). Trong nói và viết, thông thường chúng ta thường dùng những cách gọi rút gọn cho những đối tượng mà chúng ta đã đề cập trước đó. Ví dụ trong tiếng Anh, người nói và viết có thể dùng các đại từ như “it”, “they”, “he”,… Nếu không có thông tin ngữ cảnh và bộ phân tích đồng tham chiếu, sẽ là rất khó để chatbot biết các từ này tham chiếu đến đối tượng nào. Việc không xác định được đúng đối tượng mà từ thay thế tham chiếu đến có thể khiến chatbot hiểu không chính xác câu hội thoại của người dùng. Thách thức này là khá rõ ràng trong các đoạn hội thoại dài.

Vấn đề thứ hai là làm sao để giảm bớt công sức làm dữ liệu khi phát triển chatbot. Theo những cách tiếp cận ở trên, khi phát triển một ứng dụng chatbot, nhà phát triển cần gán nhãn dữ liệu huấn luyện cho bộ phân lớp intent và nhận dạng thực thể tên gọi. Trong những miền ứng dụng phức tạp (chẳng hạn như y tế), công sức để tạo ra những bộ dữ liệu đó là khá đắt đỏ. Vì thế, việc phát triển các phương pháp để tận dụng những nguồn dữ liệu sẵn có trong doanh nghiệp để giảm lượng dữ liệu cần gán nhãn, trong khi vẫn đảm bảo được độ chính xác của các mô hình xử lý ngôn ngữ tự nhiên là cần thiết.

### Sequence to sequence chatbots

Sequence to sequence là một bài toán đang được giải quyết một cách khá mạnh mẽ bởi các mạng **Deep Learning** bây giờ. Với đầu vào là 1 câu, dựa trên tập dữ liệu của chúng ta, chúng ta sẽ có thể sinh ra câu trả lời dựa vào **Deep Learning**. Bài toán này gần tương tự như bài toán dịch tự động (**Auto Translation**), chỉ khác ở chỗ, trong trường hợp này của chatbots, ngôn ngữ nguồn và ngôn ngữ đích sẽ cùng là một ngôn ngữ.

Một số từ khóa các bạn có thể tìm hiểu theo hướng giải quyết này đó là: **Recurrent Neural Network**, **LSTM**, **GRU**, **TCN** và **Transformer**.

## ****Các phương pháp tạo lập chatbot gợi ý****

Để đáp ứng và nâng cao trải nghiệm người dùng, AI chatbot là một lựa chọn tuyệt vời của các doanh nghiệp, tổ chức. Trong bài viết này giới thiệu một số định hướng xây dựng AI chatbot phổ biến, được đề xuất sử dụng rộng rãi bởi các nhà nghiên cứu.

Những nghiên cứu gần đây về chatbot thường phân theo 2 hướng lớn: (1) phương pháp pipeline và (2) phương pháp end-to-end.

**Trong phương pháp pipeline,** toàn bộ hệ thống chatbot được chia thành nhiều công đoạn, và thành phần khác nhau, chẳng hạn như đọc hiểu ngôn ngữ tự nhiên (Natural Language Understanding – NLU), quản lý hội thoại (Dialog Management) bao gồm theo dõi trạng thái hội thoại (Dialog State Tracking – DST) học các nguyên lý hội thoại (Policy Learning); sinh ngôn ngữ tự nhiên (Natural Language Generation – NLG). Ngược lại với các phương pháp tiếp cận theo pipeline, các phương pháp end-to-end xây dựng một hệ thống hội thoại bằng cách sử dụng một mô hình duy nhất, trong đó ngữ cảnh ngôn ngữ tự nhiên được lấy làm đầu vào và phản hồi ngôn ngữ tự nhiên được tạo ra làm đầu ra.

Rõ ràng, các hệ thống pipeline với cấu trúc mô-đun dễ hiểu và ổn định hơn, do đó được ứng dụng trong thương mại hóa nhiều hơn. Tuy nhiên, với việc yêu cầu ít chú thích hơn, các hệ thống end-to-end có thể trở thành một giải pháp thay thế đầy hứa hẹn trong tươn

Chi tiết quy trình nghiên cứu theo cả 2 phương pháp pipeline và end-to-end như sau:

### **Tạo lập chatbot bằng phương pháp Pipeline**

**Quy trình trong phương pháp Pipelines như sau:**

* **Hiểu ngôn ngữ tự nhiên (Natural Language Understanding – NLU)**

Ở công đoạn này, NLU có nhiệm vụ phân tích cú pháp của người dùng và đưa chúng thành một biểu diễn ngữ nghĩa có cấu trúc, thường bao gồm các cặp ý định (intent) và các cặp  thuộc tính – giá trị (slot: value). Ý định (intent) ở đây chỉ ra chức năng của lời nói, ví dụ: truy vấn hoặc cung cấp thông tin. Các cặp thuộc tính – giá trị là các yếu tố ngữ nghĩa được đề cập trong câu nói.

Ví dụ: trong câu nói “Bạn có thể giới thiệu một nhà hàng Trung Quốc ở Manhattan không?”, Các cặp thuộc tính – giá trị có thể là (“ẩm thực” : “Trung Quốc”) và (“địa điểm” : “Manhattan”). Ý định ở đây là “cung cấp thông tin”.

Phát hiện ý định (intent detection) và trích xuất thuộc tính – giá trị (slot-value extraction) có thể được giải quyết bằng cách sử dụng Mạng nơ-ron hồi quy (RNN), Mạng nơ-ron tích chập (CNN), Mạng nơ-ron đệ quy, CRF (conditional random fields) hoặc mô hình BERT bên cạnh việc kết hợp cơ chế attention..

* **Theo dõi trạng thái hội thoại (Dialog state tracking – DST)**

Ở bước này của quy trình, trình theo dõi trạng thái hộp thoại ước tính mục tiêu của người dùng bằng cách lấy toàn bộ ngữ cảnh hộp thoại làm đầu vào. Trong hầu hết các nghiên cứu gần đây, mục tiêu của người dùng được thể hiện bằng các cặp  thuộc tính – giá trị.

Năm 2013, Henderson từ Đại học Cambridge đã giới thiệu phương pháp học sâu để theo dõi trạng thái hộp thoại, trong đó sử dụng cửa sổ trượt để xuất ra một chuỗi phân phối xác suất trên một số giá trị tùy ý. Tại một số vị trí cố định cho phép áp dụng phân loại để dự đoán giá trị. Đối với các vị trí tự do có thể tạo ra giá trị trực tiếp hoặc dự đoán khoảng giá trị trong lời thoại.

Theo dõi trạng thái hộp thoại cũng có thể được thực hiện bằng cách sử dụng Mạng nơ-ron hồi quy (RNN) và Neural Belief Tracker (NBT), kết hợp một số các phương pháp tiếp cận khác.

* **Học nguyên lý hội thoại (Dialog policy learning – DPL)**

Sau khi xác định trạng thái hội thoại, bước tiếp theo là tiến hành đưa ra quyết định hành động tiếp theo của hệ thống. DPL sẽ học các nguyên lý hội thoại của con người bằng các phương thức như học có giám sát hoặc học tăng cường. Ngoài ra, phương pháp tiếp cận dựa trên quy tắc (rule-based) được sử dụng đầu tiên để khởi động hệ thống.

Hiện nay, để học nguyên lý hội thoại hầu như đều dựa vào học tăng cường (RL). Tuy nhiên, việc đào tạo một chính sách học tăng cường đòi hỏi nhiều tương tác với người dùng, điều này tốn nhiều thời gian và chi phí. Do đó, nhiều phương thức tiếp cận khác nhau đã được đề xuất để giải quyết vấn đề này, bao gồm cả việc sử dụng trình mô phỏng người dùng.

* **Sinh ngôn ngữ tự nhiên (Natural language generation – NLG)**

Ở công đoạn này; chúng ta sẽ dùng hành động được quyết định từ nguyên lý đối thoại tạo thành câu nói tự nhiên để phản hồi lại người dùng. Phản hồi tương ứng với một lời thoại của máy tính phải đảm bảo độ chính xác về mặt thông tin, cụ thể và tự nhiên.

Công đoạn này có thể được tiếp cận như một nhiệm vụ xử lý ngôn ngữ có điều kiện, bắt đầu bằng cách đào tạo trước một GPT với kho ngữ liệu quy mô lớn, và sau đó tinh chỉnh (fine-tune) mô hình về các nhiệm vụ sản sinh ngôn ngữ tự nhiên mục tiêu với một số lượng nhỏ các mẫu đã đào tạo.

### ****Tạo lập chatbot bằng phương pháp End-to-end****

Phương pháp End-to-end được đề xuất khi người ta nhận thấy một số hạn chế của phương pháp Pipeline. Đầu tiên, liên quan tới vấn đề chỉ định, với phương pháp pipeline thường khó có thể chỉ định các phản hồi của người dùng với mô-đun cụ thể tương ứng. Ngoài ra, các công đoạn có sự phụ thuộc và tương quan lẫn nhau, trong đó bất kỳ thay đổi về dữ liệu hoặc huấn luyện lại một thành phần nào của hệ thống pipeline cũng yêu cầu tất cả các thành phần khác phải được điều chỉnh cho phù hợp.

Những vấn đề này có thể được giải quyết bằng cách xây dựng một mô hình sinh (generative models) mạng nơ-ron cho các hệ thống hội thoại. Hầu hết các phương pháp end-to-end này sử dụng mô hình seq2seq (sequence to sequence).

Tuy nhiên, cách tiếp cận truyền thống của end-to-end yêu cầu  dữ liệu hội thoại rất nhiều và đa dạng  để học các hành vi đối thoại cơ bản của con người.

**Phương pháp đo độ tương tự ngữ nghĩa giữa văn bản tiếng việt**

**Giới thiệu**

Đánh giá độ tương tự ngữ nghĩa giữa các văn bản đóng vai trò quan trọng trong phát triển các ứng dụng xử lý ngôn ngữ tự nhiên như: mô hình vector trong các bài toán tìm kiếm thông tin, phân lớp văn bản, dịch máy tự động, đánh giá tính chặt chẽ của văn bản, ... Trong một số trường hợp, phương pháp đánh giáđộ tương tự giữa hai đoạn văn bản sử dụng việc so khớp từ đơn giản, dựa trên số đơn vị từ vựng xuất hiện ở cả hai đoạn văn bản đầu vào. Một số cải tiến với phương pháp đơn giản này đã được đề xuất để nâng cao hiệu quả đánh giá: xem xét đến gốc từ (stemming), loại bỏ các từ dừng, gán nhãn từ loại (part-of-speech tagging), so khớp tập con dài nhất, .... Tuy nhiên, sử dụng độ tương tự từ vựng không phải lúc nào cũng xác định được độ tương tự ngữ nghĩa văn bản.Trong khi đó có rất nhiều phương pháp đánh giá độ tương tự ngữ nghĩa giữa từ với từ, hoặc dựa trên cơ sở tri thức (knowledge-based), hoặc dựa trên kho ngữ liệu (corpus-based). Các độ đo này được áp dụng thành công trong một số bài toán xử lý ngôn ngữ như phát hiện từ dùng sai nghĩa, nhận dạng từ đồng nghĩa, .... Từ đó, một số nghiên cứu đưa ra các phương pháp đánh giá độ tương tự ngữ nghĩa văn bản dựa trên việc khai thác các thông tin có được từ độ tương tự ngữ nghĩa của các từ thành phần trong văn bản

**Word2vector**

**Giới thiêu word2vec:**

Word2vec là một mô hình đơn giản và nổi tiếng giúp tạo ra các biểu diễn embedding của từ trong một không gian có số chiều thấp hơn nhiều lần so với số từ trong từ điển.

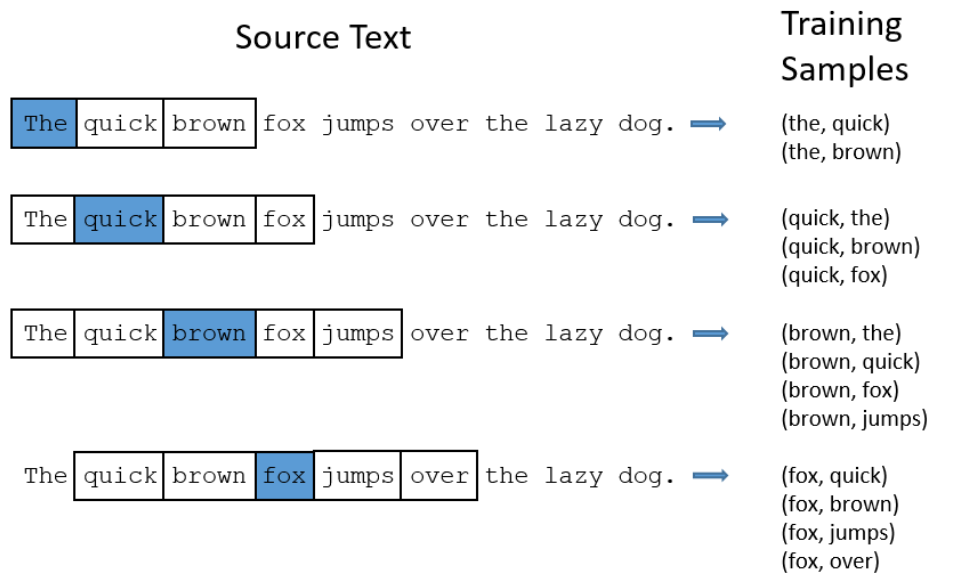
ý tưởng cơ bản:

* Hai từ xuất hiện trong những văn cảnh giống nhau thường có ý nghĩa gần với nhau.
* ta có thể đoán được một từ nếu biết các từ xung quanh nó trong câu.

**Một vài định nghĩa:**

với mỗi từ đích trong một câu của cơ sở dữ liệu, các từ ngữ cảnh được định nghĩa là các từ trong cùng câu có vị trí cách từ địch một khoảng không qúa C/2 với C là một số tự nhiên dương. Như vậy, với mỗi từ đích, ta sẽ có một bộ không quá C từ ngữ cảnh.  
ví dụ: với câu tiếng anh “ “The quick brown fox jump over the lazy dog” với C=4.

Khi “the” là từ đích, ta có cặp dữ liệu huấn luyện là (the, quick) và (the, brown). Khi “brown” là từ đích, ta có cặp dữ liệu huấn luyện là (brown, the), (brown, quick), (brown, fox) và (brown, jumps).



Word2vec định nghĩa hai embedding vector cùng chiều cho mỗi từ w trong từ điển. Khi nó là một từ đích, embedding vector của nó là u; khi nó là một từ ngữ cảnh, embedding của nó là v. Ta cần hai embedding khác nhau vì ý nghĩa của từ đó khi nó là từ đích và từ ngữ cảnh là khác nhau. Tương ứng với đó, ta có hai ma trận embedding U và V cho các từ đích và các từ ngữ cảnh.

**Phương Pháp xây dựng mô hình:**

1. ***Skip-gram:***

1.1. xây dựng hàm mất mát:

Hàm mất mát tổng cộng sẽ là tổng của hàm mất mát tại mỗi từ ngữ cảnh. Việc tối ưu hàm mất mát có thể được thực hiện thông qua Gradient Descent trên từng từ ngữ cảnh hoặc một batch các từ ngữ cảnh.

ví dụ: với từ đích là “fox” và các từ ngữ cảnh là “quick”, “brown”, “jumps” và “over”. Việc dự đoán xác suất xảy ra các từ ngữ cảnh khi biết từ đích được mô hình hóa bởi:

P("quick","brown","jumps","over"|"fox")

Ta có thể giả sử rằng sự xuất hiện của một từ ngữ cảnh khi biết từ đích *độc lập* với các từ ngữ cảnh khác để xấp xỉ xác suất trên đây bởi:

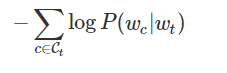
P("quick"|"fox")P("brown"|"fox")P("jumps"|"fox")P("over"|"fox")

Giả sử từ đích có chỉ số t trong từ điển v và tập hợp các chỉ số của các từ ngữ cảnh tương ứng là Ct. Số lượng phần tử của Ct dao động từ C/2 ( nếu wt đứng đầu hoặc cuối câu) tới C (nếu wt đứng ở giữa câu và có đủ C/2 từ ngữ cảnh mỗi phía).

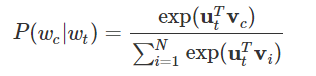
Từ dữ liệu đã có, ta cần một mô hình sao cho xác suất dưới đây càng lớn càng tốt với mỗi từ ngữ cảnh wt:



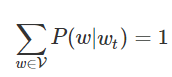
Để tránh các sai số tính toán khi nhân các số nhỏ hơn 1 với nhau, bài toán tối ưu này thường được đưa về bài toán tối thiểu đối số của log (thường được gọi là *negative log loss*):



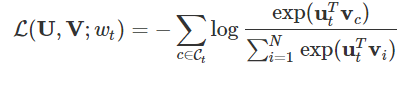
xác suất có điều kiện P(wc|wt) được định nghĩa bởi:

 (

Biểu thức này càng cao thì xác suất thu được càng lớn. Việc định nghĩa xác xuất như biểu thức như trên đảm bảo rằng:

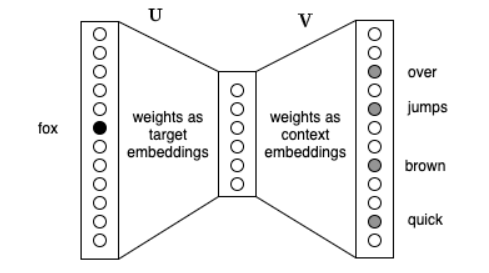


Tóm lại, hàm mất mát ứng với từ đích wt theo U,V được cho bởi:



1.2. biểu diễn dưới dạng neural:

skip-gram word2vec là một mạng neural vô cùng đơn giản với chỉ một tầng ẩn không có hàm kích hoạt.

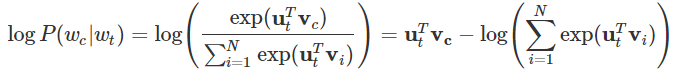


Kiến trúc đơn giản này giúp word2vec hoạt động tốt ngay cả khi số lượng từ trong từ điển là cực lớn (có thể lên tới nhiều triệu từ). Lưu ý rằng kích thước đầu vào và đầu ra của mạng word2vec này bằng với số lượng từ trong từ điển.

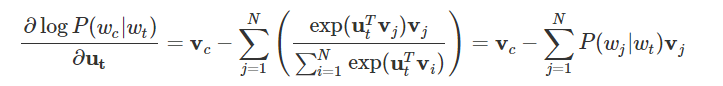
1.3. Tối ưu hàm mất mát:

Việc tối ưu hai ma trận trọng số U và V được thực hiện thông qua các thuật toán Gradient Descent. Các thuật toán tối ưu dạng này yêu cầu tính gradient cho từng ma trận.

Xét riêng số hang:

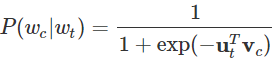


Đạo hàm theo u:



1.4. Xấp xỉ hàm mất mát và lấy mẫu âm:

Để tránh việc cập nhật rất nhiều tham số này trong một lượt, một phương pháp xấp xỉ được đề xuất giúp cải thiện tốc độ tính toán đáng kể.Mỗi xác suất P(wc|wt) được mô hình bởi một hàm sigmoid.



Bản chất của bài toán tối ưu ban đầu là xây dựng mô hình sao cho với mỗi từ đích, xác suất của một từ ngữ cảnh xảy ra là cao trong khi xác suất của *toàn bộ* các từ ngoài ngữ cảnh đó là thấp – việc này được thể hiện trong hàm softmax. Để hạn chế tính toán, trong phương pháp này ta chỉ lấy mẫu ngẫu nhiên một vài từ ngoài ngữ cảnh đó để tối ưu. Các từ trong ngữ cảnh được gọi là “từ dương”, các từ ngoài ngữ cảnh được gọi là “từ âm”; vì vậy phương pháp này còn có tên gọi khác là “lấy mẫu âm” (*negative sampling*).

Khi đó, với mỗi từ đích, ta có một bộ các từ ngữ cảnh với nhãn là 1 và 0 tương ứng với các từ ngữ cảnh ban đầu (gọi là *ngữ cảnh dương*) và các từ *ngữ cảnh âm* được lấy mẫu từ ngoài tập ngữ cảnh dương đó.

Với các từ ngữ cảnh dương, log(P(wc|wt))tương tự với hàm mất mát trog truy hồi logistic với nhãn bằng 1.

tương tự, log⁡(1−P(wc|wt)) như là hàm mất mát cho các từ ngữ cảnh âm với nhãn bằng 0.

1. ***continuous bag of words (CBOW):***

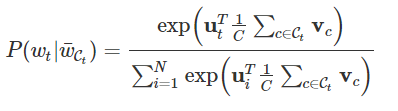
Continuous bag of Words đi tìm xác suất xảy ra từ đích khi biết các từ ngữ cảnh xung quanh. Ta cần mô hình hóa dữ liệu sao cho xác suất sau đây đạt giá trị lớn:

P("fox"|"quick","brown","jumps","over")

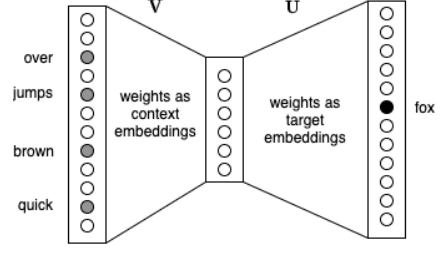
Vì có nhiều từ ngữ cảnh trong điều kiện, chúng thường được đơn giản hóa bằng cách lấy một từ “trung bình” làm đại diện.



với w¯Ct là trung bình cộng của các từ trong ngữ cảnh của từ đích wt. Embedding của từ trung bình này là trung bình của embedding các từ ngữ cảnh. Xác suất này cũng được định nghĩa tương tự như trong Skip-gram:



Biểu diễn mạng neural cho CBOW:



Lưu ý rằng giá trị tại tầng ẩn là trung bình cộng của các embedding của các từ ngữ cảnh.

Kỹ thuật tối ưu likelihood này cũng tương tự như trong Skip-gram và phương pháp lấy mẫu âm với các từ đích cũng có thể được sử dụng một cách tương tự.

Có hai cách mô hình hóa dữ liệu là Skip-gram và CBOW. Skip-gram giả sử rằng từ một từ đích ta có thể suy ra các từ ngữ cảnh. CBOW giả sử rằng từ các từ ngữ cảnh, ta có thể đoán được từ đích.

* Skip-gram làm việc tốt với dữ liệu nhỏ, nó có khả năng biểu diễn tốt những từ có tần suất thấp. Việc này hợp lý vì ta xây dựng được nhiều mẫu huấn luyện xung quanh từ có tần suất thấp này.
* CBOW phù hợp với các bộ dữ liệu lớn khi mà số mẫu huấn luyện được tạo ra từ mỗi ngữ cảnh (chỉ là một) ít hơn nhiều so với Skip-gram (tỉ lệ với kích thước cửa số ngữ cảnh). CBOW biểu diễn tốt hơn các từ xảy ra thường xuyên.

Word2vec không chỉ có thể sử dụng để tạo embedding cho các từ mà còn có thể áp dụng cho các bộ dữ liệu khác mà sự xuất hiện của một đối tượng phụ thuộc vào các đối tượng khác trong cùng văn cảnh.

**Độ tương tự giữa 2 văn bản dựa vào mô hình Word2Vec**

Với mô hìnhWord2Vec, dựa vào độ đo Cosine giữa 2 vector, ta có thể lựa chọn ra rất nhiều từ tương tự với từ đã cho (độ đo Cosine càng lớn, độ tương tự càng lớn). Tuy nhiên trong nghiên cứu này, chúng tôi chỉ lấy những từ có khoảng cách Cosine lớn hơn một ngưỡng xác định, ngưỡng này được lựa chọn thông qua thử nghiệm được trình bày ở mục 3.

Ta gọi đó là các từ gần gũi (thường là các từ đồng nghĩa) với từ đã cho.Khi có tập từ gần gũi, ta sẽ cải tiến thuật toán đo độ tương tự về từ vựng giữa 2 câu để thu được độ tương tự ngữ nghĩa giữa 2 câu, cụ thể như sau:

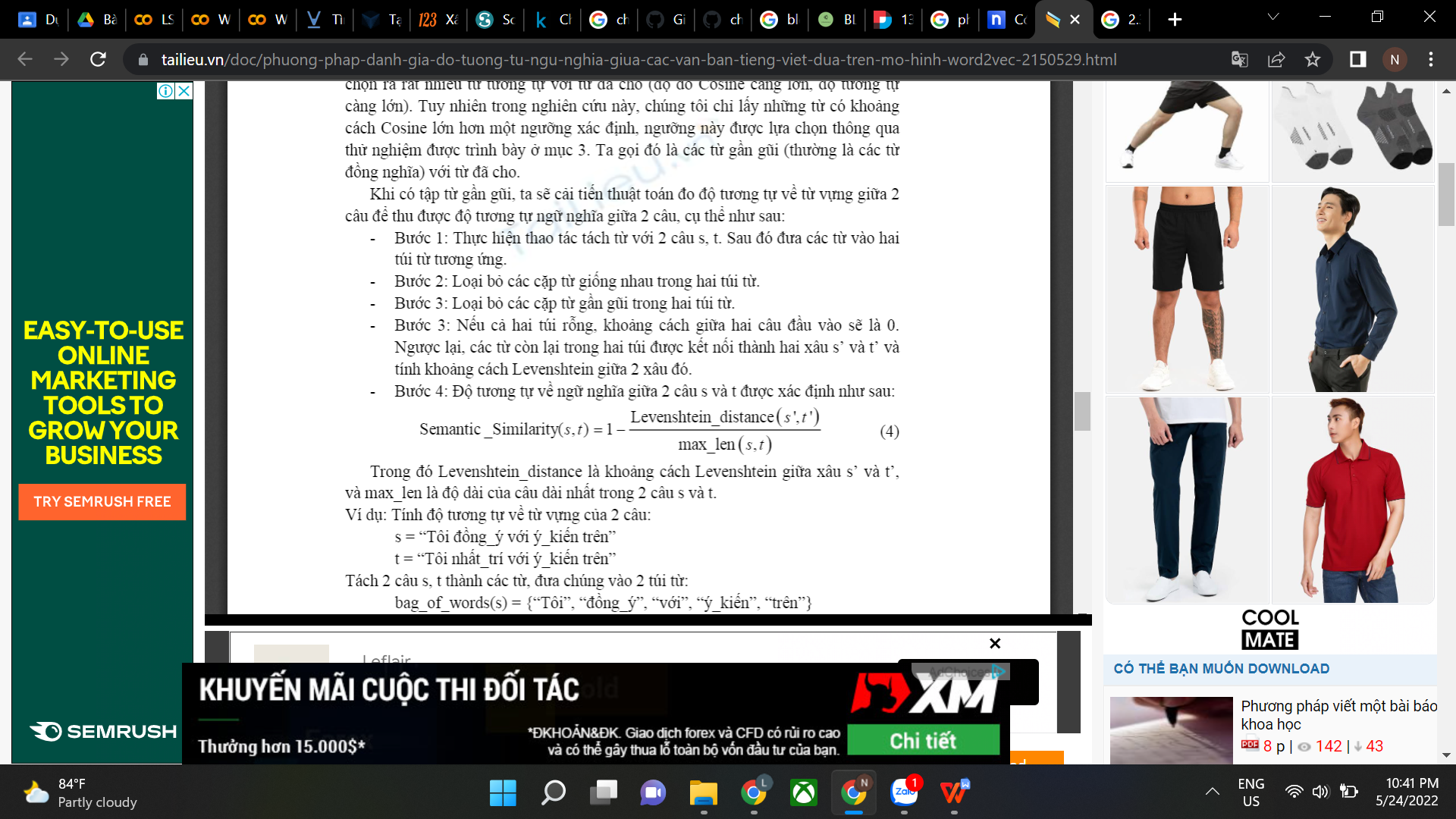
-Bước 1: Thực hiện thao tác tách từ với 2 câu s, t. Sau đó đưa các từ vào hai túi từ tương ứng

.-Bước 2: Loại bỏ các cặp từ giống nhau trong hai túi từ.

-Bước 3: Loại bỏ các cặp từ gần gũi trong hai túi từ.

-Bước 3: Nếu cả hai túi rỗng, khoảng cách giữa hai câu đầu vào sẽ là 0. Ngược lại, các từ còn lại trong hai túi được kết nối thành hai xâu s’ và t’ và tính khoảng cách Levenshtein giữa 2 xâu đó.

-Bước 4: Độ tương tự về ngữ nghĩa giữa 2 câu s và t được xác định như sau:



Trong đó Levenshtein\_distance là khoảng cách Levenshtein giữa xâu s’ và t’, và max\_len là độ dài của câu dài nhất trong 2 câu s và t

**Bleu score**

[BLEU](https://trituenhantao.io/tu-dien-thuat-ngu/bleu/" \t "https://trituenhantao.io/kien-thuc/bleu-phep-do-trong-dich-may/_blank) là viết tắt của **Bilingual Evaluation Understudy**, là phương pháp đánh giá một bản dịch dựa trên các bản dịch tham khảo, được giới thiệu trong paper [BLEU: a Method for Automatic Evaluation of Machine Translation](http://www.aclweb.org/anthology/P02-1040.pdf)). BLEU được thiết kế để sử dụng trong [dịch máy](https://trituenhantao.io/kien-thuc/ung-dung-cua-nlp-7-thanh-tu-noi-bat/) (Machine Translation), nhưng thực tế, phép đo này cũng được sử dụng trong các nhiệm vụ như tóm tắt văn bản, nhận dạng giọng nói, sinh nhãn ảnh v..v.. Bên cạnh đó phép đo này hoàn toàn có thể sử dụng để đánh giá chất lượng bản dịch của nhân viên.

Điều kiện tiên quyết để có thể sử dụng BLEU là bạn phải có một (hoặc nhiều) câu mẫu. Đối với bài toán dịch máy, câu mẫu chính là câu đầu ra của cặp câu trong tập dữ liệu. [BLEU](https://trituenhantao.io/tu-dien-thuat-ngu/bleu/" \t "https://trituenhantao.io/kien-thuc/bleu-phep-do-trong-dich-may/_blank) đánh giá một câu thông qua việc so khớp câu đó với các câu mẫu và cho thang điểm từ 0 (sai lệch tuyệt đối) đến 1 (khớp tuyệt đối).

BLEU được biết đến như một phương pháp đơn giản, dễ hiểu, chi phí tính toán thấp và tương đồng với cách đánh giá của con người. Mặc dù vậy, yếu tổ con người trong việc sinh câu mẫu làm cho BLEU không hoàn toàn khách quan. Ví dụ, cùng một câu có thể có nhiều bản dịch tốt và việc viết tất cả các bản dịch đó vào tập câu mẫu đôi khi bất khả thi.

Cách tính của [BLEU](https://trituenhantao.io/tu-dien-thuat-ngu/bleu/" \t "https://trituenhantao.io/kien-thuc/bleu-phep-do-trong-dich-may/_blank) là đếm số [n-gram](https://trituenhantao.io/tu-dien-thuat-ngu/n-gram/" \t "https://trituenhantao.io/kien-thuc/bleu-phep-do-trong-dich-may/_blank) khớp nhau giữa câu mẫu (R) và câu được đánh giá (C) sau đó chia cho số [token](https://trituenhantao.io/tu-dien-thuat-ngu/token/" \t "https://trituenhantao.io/kien-thuc/bleu-phep-do-trong-dich-may/_blank) của C. Việc chọn n phụ thuộc vào ngôn ngữ, nhiệm vụ và mục tiêu cụ thể. Đơn giản nhất ta có thể sử dụng [unigram](https://trituenhantao.io/tu-dien-thuat-ngu/unigram/" \t "https://trituenhantao.io/kien-thuc/bleu-phep-do-trong-dich-may/_blank) là n-gram chứa 1 token (n=1). Một cách trực quan, n càng lớn, câu văn càng mượt.

Việc so khớp này không phụ thuộc vào vị trí, do đó BLEU không thể đánh giá được thứ tự của từ. Đây vừa là ưu điểm vừa là hạn chế của BLEU. Trong ngôn ngữ, một câu có thể được biểu diễn bởi các thứ tự từ khác nhau nhưng vẫn phải tuân theo những quy tắc nhất định.

Ngoài ra, để tránh việc một bản dịch lặp đi lặp lại một từ mà vẫn được “chấm điểm” cao (ví dụ như ****“this this this this”**** được so với ****“this is a cat”****), [BLEU](https://trituenhantao.io/tu-dien-thuat-ngu/bleu/" \t "https://trituenhantao.io/kien-thuc/bleu-phep-do-trong-dich-may/_blank) tính đến số lần xuất hiện lớn nhất của mỗi n-gram trong toàn bộ các câu mẫu để giới hạn số lần khớp tối đa. Trong ví dụ trên, ****this**** sẽ chỉ được tính 1 lần”

IMG_256Số lần khớp tối đa được giới hạn bởi số lần xuất hiện tối đa của [n-gram](https://trituenhantao.io/tu-dien-thuat-ngu/n-gram/" \t "https://trituenhantao.io/kien-thuc/bleu-phep-do-trong-dich-may/_blank) trong các câu mẫu.

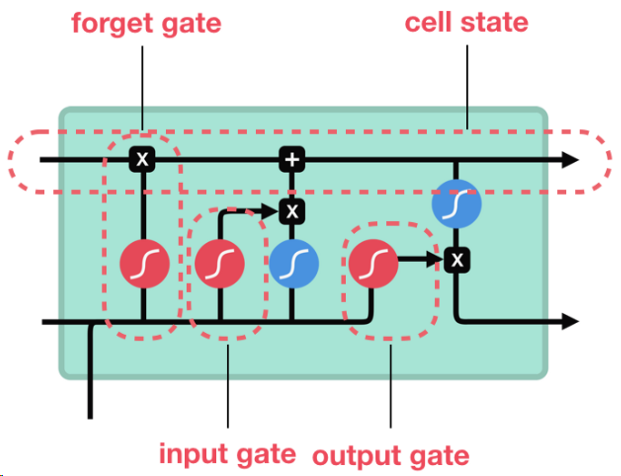
Trong Python, để tính điểm BLEU chúng ta có thể thực hiện dễ dàng với sự hỗ trợ của thư viện [NLTK](https://www.nltk.org/).

**LSTM**

**LSTM model**

Về cơ bản mô hình của LSTM không khác mô hình truyền thống của RNN, nhưng chúng sử dụng hàm tính toán khác ở các trạng thái ẩn. Bộ nhớ của LSTM được gọi là tế bào (Cell) và bạn có thể tưởng tượng rằng chúng là các hộp đen nhận đầu vào là trạng thái phía trước *ht*−1​ và đầu vào hiện tại *xt*​. Bên trong hộp đen này sẽ tự quyết định cái gì cần phải nhớ và cái gì sẽ xoá đi. Sau đó, chúng sẽ kết hợp với trạng thái phía trước, nhớ hiện tại và đầu vào hiện tại. Vì vậy mà ta ta có thể truy xuất được quan hệ của các từ phụ thuộc xa nhau rất hiệu quả.

Ngoài việc sử dụng hai dữ liệu đầu vào là Current input (xt) và Hidden state ht-1 như mạng RNN thông thường, LSTM còn sử dụng thêm một đầu vào nữa là Cell state (c t-1). Ở đây chúng ta có thể coi Cell state là ‘memory’ của mạng LSTM và nó có thể lưu trữ thông tin của các timestep đầu tiên, do đó hạn chế ảnh hưởng của short-term memory như trong Vanilla RNN

Cấu trúc Cell của LSTM được hình thành từ 3 gates: Forget gate, Input gate và Output gate như sau:

a) Forget gate: có nhiệm vụ quyết định liệu thông tin của Cell state ở timestep t-1 có cần được lưu trữ hay không. Thông tin từ Current input và Hidden state được chuyển qua Sigmoid function với output nằm trong khoảng [0, 1]. Do đó nếu giá trị output gần bằng 1, tức là thông tin cần được giữ lại còn nếu output gần bằng 0 thì phải loại bỏ thông tin.

b) Input gate: có nhiệm vụ cập nhật thông tin vào Cell state. Ở đây ta nhân output của Sigmoid với output của Tanh để quyết định thông tin của Current input và Hidden state có nên được cập nhật vào Cell state hay không.

c) Output gate: có nhiệm vụ tính giá trị của Hidden state cho timestep tiếp theo. Với việc sử dụng Forget gate và Input gate, ta có thể tính được giá trị mới của Cell state và từ đó kết hợp với Current input và Hidden state để tính giá trị của Hidden state tiếp theo. Ở đây giá trị của Hidden state mới này cũng chính là giá trị Prediction.

Như vậy, trong LSTM chúng ta có 3 input và 2 output là Cell state và Hidden state.

## Ý tưởng cốt lõi của LSTM

Chìa khóa của LSTM là trạng thái tế bào (cell state) - chính đường chạy thông ngang phía trên của sơ đồ hình vẽ.

Trạng thái tế bào là một dạng giống như băng truyền. Nó chạy xuyên suốt tất cả các mắt xích (các nút mạng) và chỉ tương tác tuyến tính đôi chút. Vì vậy mà các thông tin có thể dễ dàng truyền đi thông suốt mà không sợ bị thay đổi.

[](https://colah.github.io/posts/2015-08-Understanding-LSTMs/img/LSTM3-C-line.png)

LSTM có khả năng bỏ đi hoặc thêm vào các thông tin cần thiết cho trạng thái tế báo, chúng được điều chỉnh cẩn thận bởi các nhóm được gọi là cổng (gate).

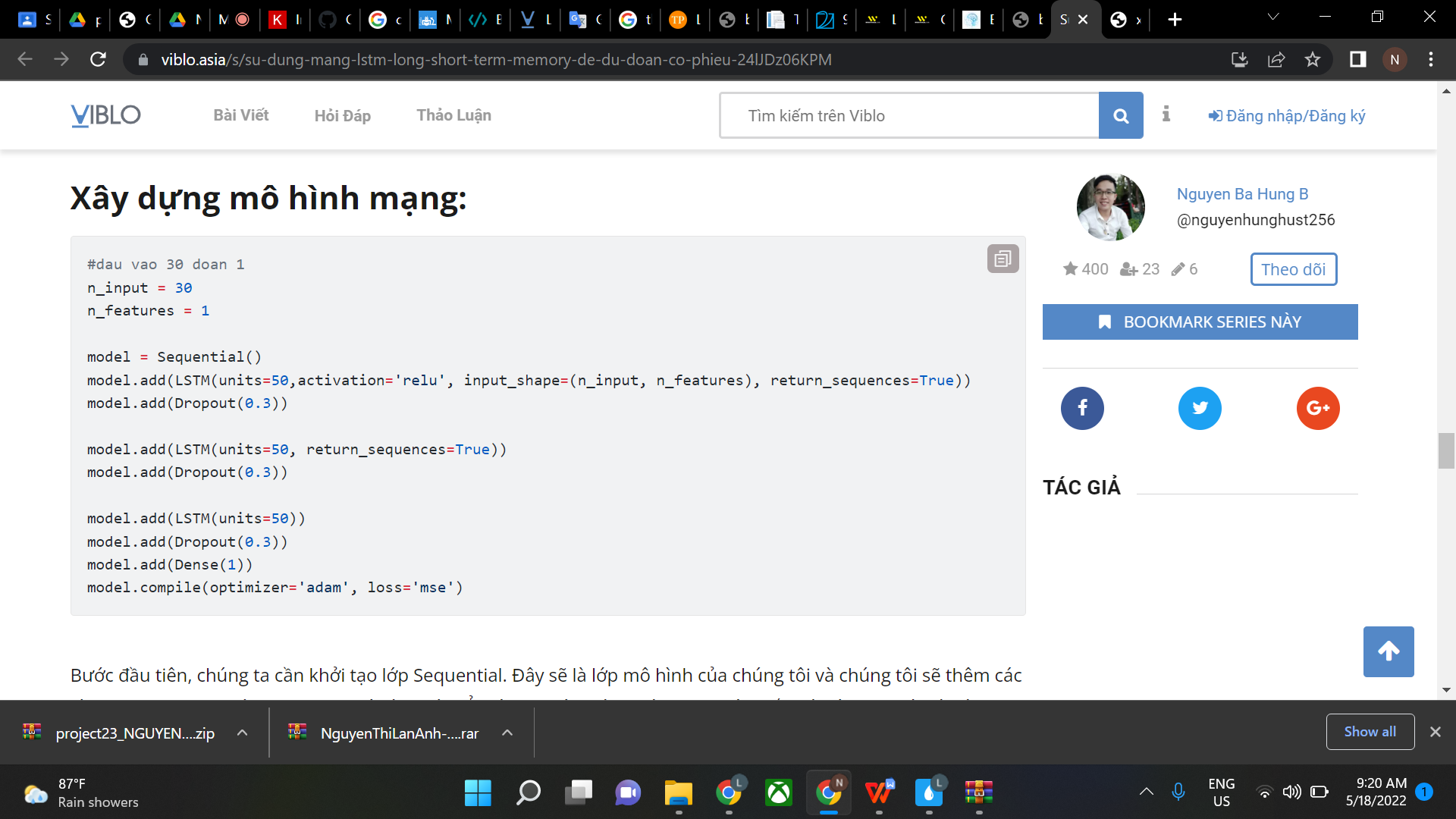
Các cổng là nơi sàng lọc thông tin đi qua nó, chúng được kết hợp bởi một tầng mạng sigmoid và một phép nhân.

[](https://colah.github.io/posts/2015-08-Understanding-LSTMs/img/LSTM3-gate.png)

Tầng sigmoid sẽ cho đầu ra là một số trong khoản [0, 1][0,1], mô tả có bao nhiêu thông tin có thể được thông qua. Khi đầu ra là 00 thì có nghĩa là không cho thông tin nào qua cả, còn khi là 11 thì có nghĩa là cho tất cả các thông tin đi qua nó.

Một LSTM gồm có 3 cổng như vậy để duy trì và điều hành trạng thái của tế bào.

**Code minh họa**



## 1.Lớp LSTM

Chúng tôi sẽ thiết lập một chức năng để xây dựng các lớp LSTM để xử lý số lượng các lớp và kích thước một cách linh hoạt. Dịch vụ sẽ lấy một danh sách các kích thước LSTM, có thể cho biết số lượng các lớp LSTM dựa trên độ dài của danh sách (ví dụ: ví dụ của chúng tôi sẽ sử dụng danh sách có độ dài 2, chứa các kích thước 128 và 64, cho biết mạng LSTM hai lớp trong đó kích thước lớp đầu tiên là 128 và lớp thứ hai có kích thước lớp ẩn là 64).

**2. Lớp Dense**

có thể được định nghĩa là lớp Mạng thần kinh chung được kết nối dày đặc. ****output = activation(dot(input, kernel) +bias)**** được thực thi bởi Lớp dense. Ở đây, một chức năng kích hoạt phần tử khôn ngoan đang được thực hiện bởi kích hoạt, để truyền một đối số kích hoạt, một ma trận trọng số được gọi là hạt nhân được xây dựng bởi lớp và thiên vị là một vectơ được tạo bởi lớp, chỉ có thể áp dụng nếu use\_bias là True

Dense ( ): Layer này cũng như một layer neural network bình thường, với các tham số sau:

* units : số chiều output, như số class sau khi train ( chó , mèo, lợn, gà).
* activation : chọn activation đơn giản với sigmoid thì output có 1 class.
* use\_bias : có sử dụng bias hay không (True or False)
* bias\_initializer: Nó có thể được định nghĩa như một bộ khởi tạo cho vectơ thiên vị mà Keras sử dụng bộ khởi tạo zero theo mặc định. Giả định rằng nó đặt vector thiên vị thành tất cả các số không.
* kernel\_regularizer: Nó có thể được gọi là một hàm điều chỉnh, được thực hiện trên ma trận trọng số hạt nhân .
* bias\_regularizer: Nó có thể được định nghĩa là một hàm điều chỉnh, được áp dụng cho vector thiên vị.
* Activity\_regualrizer: Nó liên quan đến một hàm điều chỉnh được áp dụng cho đầu ra của lớp (kích hoạt của nó).
* kernel\_constraint: Nó đề cập đến ràng buộc, được áp dụng cho ma trận trọng số hạt nhân.
* bias\_constraint: Nó có thể được định nghĩa là một ràng buộc, được áp dụng cho vector thiên vị.

****Input shape****

Lớp ****Input shape**** chấp nhận một tensor nD của hình dạng ( batch\_size,…, input\_dim) và đảm bảo rằng tình huống phổ biến nhất của nó sẽ phải là đầu vào 2D bao gồm một hình dạng (batch\_size, input\_dim) .

****Output shape****

Nó xuất ra một tensor nD của hình dạng ( batch\_size,…, units) . Ví dụ, trong đó đầu vào là dạng 2D có hình dạng (kích thước lô, đầu vào\_dim) , thì đầu ra tương ứng sẽ có hình dạng (kích thước lô\_ô, đơn vị) .

## 3.Dropout

****keras.layers.Dropout (rate, noise\_shape = None, seed = None)****

Việc bỏ qua được áp dụng cho đầu vào vì nó ngăn chặn việc trang bị quá mức bằng cách đặt ngẫu nhiên các đơn vị của tỷ lệ phân số thành 0 trong thời gian đào tạo ở mỗi lần cập nhật.

****Argument****

* rate: Nó đề cập đến một giá trị thực giữa 0 và 1, đại diện cho các đơn vị phân số được bỏ đi.
* noise\_shape: Nó đề cập đến số nguyên tensor một chiều mô phỏng hình dạng của mặt nạ bỏ học nhị phân, sẽ được sử dụng trong phép nhân với đầu vào. Nếu hình dạng đầu vào là (batch\_size, timesteps, features) và đối với tất cả các timeteps, bạn muốn mặt nạ bỏ qua của mình giống nhau, thì, trong trường hợp đó, noise\_shape = (batch\_size, 1, feature) có thể được sử dụng.
* seed: Nó chỉ ra một số nguyên python sẽ được sử dụng làm hạt giống ngẫu nhiên.

**4. Hàm compile:** Ở hàm này chúng ta sử dụng để training models như thuật toán train qua optimizer như Adam, SGD, RMSprop,..

* learning\_rate : dạng float , tốc độc học, chọn phù hợp để hàm số hội tụ nhanh.

****Transformer Neural Network****

Transformer là một mô hình học sâu được thiết kế để phục vụ giải quyết nhiều bài toán trong xử lý ngôn ngữ và tiếng nói, ví dụ như bài toán dịch tự động, bài toán sinh ngôn ngữ, phân loại, nhận dạng thực thể, nhận dạng tiếng nói, chuyển văn bản thành tiếng nói. Tuy nhiên, khác với RNNs, Transformer không xử lý các phần tử trong một chuỗi một cách tuần tự. Nếu dữ liệu đầu vào là một câu ngôn ngữ tự nhiên, Transformer không cần phải xử lý phần đầu câu trước rồi mới tới phần cuối câu. Do tính năng này, Transformer có thể tận dụng khả năng tính toán song song của GPU và giảm thời gian xử lý đáng kể.

Không sử dụng kiến trúc Recurrent (hồi quy) như RNNs mà Transformer sử dụng self-attention. Trong kiến trúc của mình, Transformer chứa 6 encoder và 6 decoder. Mỗi encoder chứa hai lớp: Self-attention và mạng truyền thẳng (FNN).

Self-Attention là cơ chế giúp encoder nhìn vào các từ khác trong lúc mã hóa một từ cụ thể, vì vậy,Transformers có thể hiểu được sự liên quan giữa các từ trong một câu, kể cả khi chúng có khoảng cách xa. Các decoder cũng có kiến trúc giống như vậy nhưng giữa chúng có một lớp attention để nó có thể tập trung vào các phần liên quan của đầu vào.

**Self-Attention có 4 bước:**

1. Tạo ra bộ 3 vectơ từ các vectơ đầu vào của encoder. Tại encoder đầu tiên, véctơ đầu vào là word embedding của từ. Như vậy với mỗi từ, ta sẽ có 3 vectơ Query, Key và Value. Các vectơ này được tạo nên bởi phép nhân ma trận giữa véctơ đầu vào và 3 ma trận trọng số tương ứng với query, key, value mà chúng ta sử dụng trong quá trình huấn luyện. 3 vectơ này đóng vai trò khác nhau và đều quan trọng đối với attention.
2. Tính điểm. Với mỗi từ, ta cần tính điểm của các từ khác trong câu đối với từ này. Giá trị này giúp quyết định từ nào cần được chú ý và chú ý bao nhiêu khi mã hóa một từ. Điểm được tính bằng tích vô hướng giữa véctơ Query của từ đang xét với lần lượt các vectơ Key của các từ trong câu. Ví dụ, khi ta tính self-attention trên từ có vị trí 1, điểm của nó với chính nó là q1.k1, điểm của nó với từ thứ hai là q1.k2, v..v..
3. Chuẩn hóa điểm. Trong bài báo gốc, điểm được chia cho 8 (căn bậc 2 của 64 – số chiều của vectơ Key). Điều này giúp cho độ dốc trở nên ổn định hơn. Tiếp theo, giá trị này được truyền qua hàm softmax để đảm bảo các giá trị điểm đều dương và có tổng không vượt quá 1.
4. Nhân vectơ Value với mỗi giá trị điểm đã tính phía trên rồi cộng lại với nhau. Ý đồ của việc này là bảo toàn giá trị vectơ của các từ cần được chú ý và loại bỏ vectơ của các từ không liên quan (bằng cách nhân nó với một số rất nhỏ, ví dụ như 0.001).