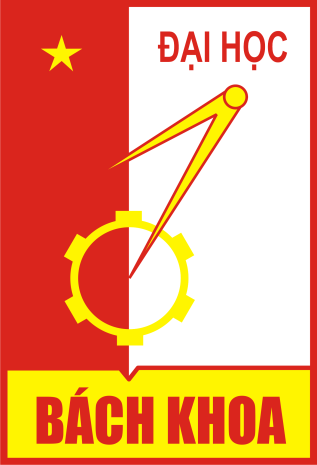


**ĐẠI HỌC BÁCH KHOA HÀ NỘI**

VIỆN CÔNG NGHỆ THÔNG TIN & TRUYỀN THÔNG

-----🙞🙜🕮🙞-----



**BÁO CÁO HỌC PHẦN**

Khai phá web

ĐỀ TÀI: Nhận diện cảm xúc

Nhóm 7

*Sinh viên thực hiện :*

**Nguyễn Thượng Hiền - 20151397**

**Nguyễn Đức Tùng - 20154253**

**Phạm Khắc Hòa - 20146314**

**Nguyễn Anh Tuấn - 20144898**

**Hà Nội, tháng 12/2018**

Mục lục

[**LỜI NÓI ĐẦU** 3](#_Toc532965641)

[1.Tổng quan đề tài 4](#_Toc532965642)

[2.Giới thiệu mô hình sử dụng: 4](#_Toc532965643)

[a.Mạng hồi quy RNN: 4](#_Toc532965644)

[b.Mạng LSTM 6](#_Toc532965645)

[c. Mạng Bidirectional LSTM (BiLSTM) 10](#_Toc532965646)

[3. Triển khai mô hình 10](#_Toc532965647)

[3.1.Thu thập dữ liệu 10](#_Toc532965648)

[3.2.Chuẩn bị dữ liệu cho mô hình 12](#_Toc532965649)

[a. Tách từ cho tiếng Việt 12](#_Toc532965650)

[b. Biểu diễn từ trong không gian (word embeding) 12](#_Toc532965651)

[3.3.Mạng LSTM 14](#_Toc532965652)

[3.4.Mạng Bi-LSTM 15](#_Toc532965653)

[3.5. Xây dựng nhận diện cảm xúc 15](#_Toc532965654)

[3.6. Xây dựng nhận diện khía cạnh 15](#_Toc532965655)

[3.7. Xây dựng nhận diện khía cạnh và cảm xúc song song 15](#_Toc532965656)

[3. Các khó khăn gặp phải trong quá trình thực hiện đồ án và cách thức giải quyết 16](#_Toc532965657)

[4. Kết luận và các đề cử cho việc tiếp tục phát triển và cải tiến trong tương lai 17](#_Toc532965658)

[**KẾT LUẬN** 18](#_Toc532965659)

LỜI NÓI ĐẦU

Hiện nay, sự phát triển của học máy đã làm cho các lĩnh vực khác nhau có những đột phá. Nhiều lĩnh vực trong cuộc sống đã ứng dụng các kỹ thuật học máy và đạt được nhiều kết quả tốt ,cùng với đó là những hướng đi mới hiệu quả cho những bài toán khác nhau . Và 1 trong những bài toán, vấn đề trong NLP là nhận diện cảm xúc , khía cạnh cũng đã có được bước đi mới, hiệu quả và chính xác hơn.Nhằm tìm hiểu và đánh giá một số mô hình học máy đối với việc xử lý ngôn ngữ tự nhiên, nhóm chúng em đã thực hiện đề tài :Nhận diện cảm xúc trong tiếng Việt.

**Github:**

https://github.com/NguyenThuongHien/Sentiment-Analysis-Foody

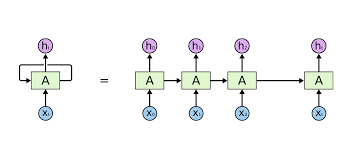
# 1.Tổng quan đề tài

Nhận diện cảm xúc là 1 bài toán không phải là xa lạ trong xử lý ngôn ngữ tự nhiên, nhưng với mỗi ngôn ngữ lại có những nét đặc trưng riêng.Đối với tiếng Việt , nhận diện cảm xúc thuộc dạng bài toán học máy có giám sát, gán nhán cho dữ liệu đầu vào từ đó xác định được cảm xúc trong bình luận của người dùng .Có nhiều phương pháp để giải quyết vấn đề này và 1 trong số đó là sử dụng Deep Learning ,và mô hình chúng em lựa chọn là LSTM,sử dụng mạng

# 2.Giới thiệu mô hình sử dụng:

## a.**Mạng hồi quy RNN:**

Recurrent Neural Network (RNN) là một loại Neural Network trong đó mỗi node trong các lớp ẩn có một kết nối với chính bản thân. Chính kết nối này tạo ra các trạng thái nội tại của kiến trúc mạng cho phép mô hình hoá các chuỗi với độ dài bất kỳ.



Hình vẽ trên mô tả một đoạn của mạng nơ-ron hồi quy AA với đầu vào là x\_txt và đầu ra là h\_tht . Một vòng lặp cho phép thông tin có thể được truyền từ bước này qua bước này qua bước khác của mạng nơ-ron.

Các vòng lặp này khiến cho mạng nơ-ron hồi quy trông có vẻ khó hiểu. Tuy nhiên, nó không khác mấy so với các mạng nơ-ron thuần. Một mạng nơ-ron hồi quy có thể được coi là nhiều bản sao chép của cùng một mạng, trong đó mỗi đầu ra của mạng này là đầu vào của một mạng sao chép khác.

Ý tưởng chính của RNN (Recurrent Neural Network) là sử dụng chuỗi các thông tin. Trong các mạng nơ-ron truyền thống tất cả các đầu vào và cả đầu ra là độc lập với nhau. Tức là chúng không liên kết thành chuỗi với nhau. Nhưng các mô hình này không phù hợp trong rất nhiều bài toán. Ví dụ, nếu muốn đoán từ tiếp theo có thể xuất hiện trong một câu thì ta cũng cần biết các từ trước đó xuất hiện lần lượt thế nào chứ nhỉ? RNN được gọi là hồi quy (Recurrent) bởi lẽ chúng thực hiện cùng một tác vụ cho tất cả các phần tử của một chuỗi với đầu ra phụ thuộc vào cả các phép tính trước đó. Nói cách khác, RNN có khả năng nhớ các thông tin được tính toán trước đó. Trên lý thuyết, RNN có thể sử dụng được thông tin của một văn bản rất dài, tuy nhiên thực tế thì nó chỉ có thể nhớ được một vài bước trước đó (ta cùng bàn cụ thể vấn đề này sau) mà thôi.

Mô hình trên mô tả phép triển khai nội dung của một RNN. Triển khai ở đây có thể hiểu đơn giản là ta vẽ ra một mạng nơ-ron chuỗi tuần tự. Ví dụ ta có một câu gồm 5 chữ “Đẹp trai lắm gái theo”, thì mạng nơ-ron được triển khai sẽ gồm 5 tầng nơ-ron tương ứng với mỗi chữ một tầng. Lúc đó việc tính toán bên trong RNN được thực hiện như sau:

\color{blue}x\_txt là đầu vào tại bước \color{blue}tt. Ví dụ, \color{deeppink}x\_1x1 là một vec-tơ one-hot tương ứng với từ thứ 2 của câu (trai).

\color{blue}s\_tst là trạng thái ẩn tại bước \color{blue}tt. Nó chính là bộ nhớ của mạng. \color{blue}s\_tst được tính toán dựa trên cả các trạng thái ẩn phía trước và đầu vào tại bước đó: \color{blue}s\_t = f(U x\_t + W s\_{t-1} )st =f(Uxt +Wst−1 ). Hàm \color{blue}ff thường là một hàm phi tuyến tính như [tang hyperbolic (tanh)](https://vi.wikipedia.org/wiki/H%C3%A0m_hypebolic) hay [ReLu](https://en.wikipedia.org/wiki/Rectifier_(neural_networks)). Để làm phép toán cho phần tử ẩn đầu tiên ta cần khởi tạo thêm \color{deeppink}s\_{-1}s−1 , thường giá trị khởi tạo được gắn bằng 0.

￼ot là đầu ra tại bước ￼t. Ví dụ, ta muốn dự đoán từ tiếp theo có thể xuất hiện trong câu thì ￼ot chính là một vec-tơ xác xuất các từ trong danh sách từ vựng của ta: ￼

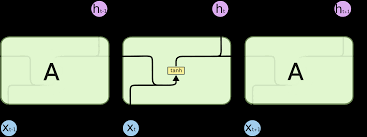
Trong vài năm gần đây, việc ứng dụng RNN đã đưa ra được nhiều kết quả không thể tin nổi trong nhiều lĩnh vực: nhận dạng giọng nói, mô hình hóa ngôn ngữ, dịch máy, mô tả ảnh,… Danh sách vẫn còn đang được mở rộng tiếp.

## b.Mạng LSTM

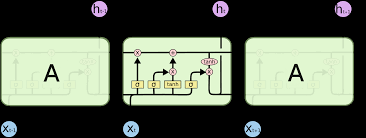
Mạng bộ nhớ dài-ngắn (Long Short Term Memory networks), thường được gọi là LSTM - là một dạng đặc biệt của RNN, nó có khả năng học được các phụ thuộc xa. LSTM được giới thiệu bởi [Hochreiter & Schmidhuber (1997)](http://deeplearning.cs.cmu.edu/pdfs/Hochreiter97_lstm.pdf), và sau đó đã được cải tiến và phổ biến bởi rất nhiều người trong ngành. Chúng hoạt động cực kì hiệu quả trên nhiều bài toán khác nhau nên dần đã trở nên phổ biến như hiện nay.

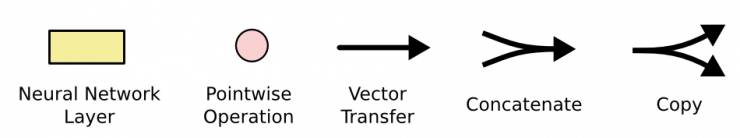
LSTM được thiết kế để tránh được vấn đề phụ thuộc xa (long-term dependency). Việc nhớ thông tin trong suốt thời gian dài là đặc tính mặc định của chúng, chứ ta không cần phải huấn luyện nó để có thể nhớ được. Tức là ngay nội tại của nó đã có thể ghi nhớ được mà không cần bất kì can thiệp nào.

Mọi mạng hồi quy đều có dạng là một chuỗi các mô-đun lặp đi lặp lại của mạng nơ-ron. Với mạng RNN chuẩn, các mô-dun này có cấu trúc rất đơn giản, thường là một tầng *￼tanh*.



LSTM cũng có kiến trúc dạng chuỗi như vậy, nhưng các mô-đun trong nó có cấu trúc khác với mạng RNN chuẩn. Thay vì chỉ có một tầng mạng nơ-ron, chúng có tới 4 tầng tương tác với nhau một cách rất đặc biệt.





Ở sơ đồ trên, mỗi một đường mang một véc-tơ từ đầu ra của một nút tới đầu vào của một nút khác. Các hình trong màu hồng biểu diễn các phép toán như phép cộng véc-tơ chẳng hạn, còn các ô màu vàng được sử dụng để học trong các từng mạng nơ-ron. Các đường hợp nhau kí hiệu việc kết hợp, còn các đường rẽ nhánh ám chỉ nội dung của nó được sao chép và chuyển tới các nơi khác nhau

**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.



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

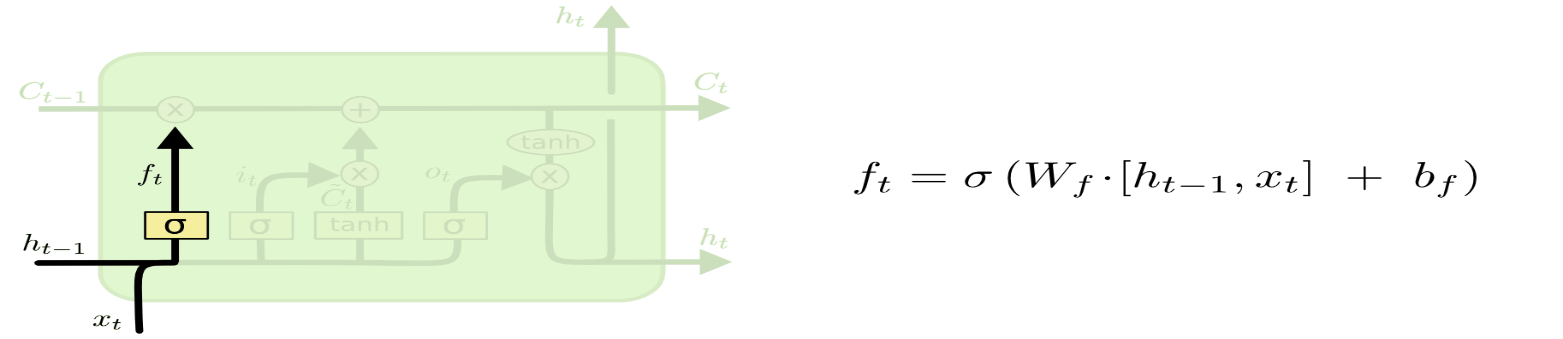


Tầng sigmoid sẽ cho đầu ra là một số trong khoản *[0, 1]*[0,1], mô tả có bao nhiêu thông tin có thể được thông qua. Khi đầu ra là *0*0 thì có nghĩa là không cho thông tin nào qua cả, còn khi là *1*1 thì có nghĩa là cho tất cả các thông tin đi qua nó.

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

**Bên trong LSTM:**

Bước đầu tiên của LSTM là quyết định xem thông tin nào cần bỏ đi từ trạng thái tế bào. Quyết định này được đưa ra bởi tầng sigmoid - gọi là “tầng cổng quên” (forget gate layer). Nó sẽ lấy đầu vào là *h\_{t-1}ht*−1 và *x\_txt* rồi đưa ra kết quả là một số trong khoảng *[0, 1]*[0,1] cho mỗi số trong trạng thái tế bào *C\_{t-1}Ct*−1 . Đẩu ra là *1*1thể hiện rằng nó giữ toàn bộ thông tin lại, còn *0*0 chỉ rằng taonf bộ thông tin sẽ bị bỏ đi.



Bước tiếp theo là quyết định xem thông tin mới nào ta sẽ lưu vào trạng thái tế bào. Việc này gồm 2 phần. Đầu tiên là sử dụng một tầng sigmoid được gọi là “tầng cổng vào” (input gate layer) để quyết định giá trị nào ta sẽ cập nhập. Tiếp theo là một tầng *tanhtanh* tạo ra một véc-tơ cho giá trị mới *\tilde{C\_t}Ct* ~ nhằm thêm vào cho trạng thái. Trong bước tiếp theo, ta sẽ kết hợp 2 giá trị đó lại để tạo ra một cập nhập cho trạng thái.

Chẳng hạn với ví dụ mô hình ngôn ngữ của ta, ta sẽ muốn thêm giới tính của nhân vật mới này vào trạng thái tế bào và thay thế giới tính của nhân vật trước đó.

Cuối cùng, ta cần quyết định xem ta muốn đầu ra là gì. Giá trị đầu ra sẽ dựa vào trạng thái tế bào, nhưng sẽ được tiếp tục sàng lọc. Đầu tiên, ta chạy một tầng sigmoid để quyết định phần nào của trạng thái tế bào ta muốn xuất ra. Sau đó, ta đưa nó trạng thái tế bảo qua một hàm *tanhtanh* để co giá trị nó về khoảng *[-1, 1]*[−1,1], và nhân nó với đầu ra của cổng sigmoid để được giá trị đầu ra ta mong muốn.

Với ví dụ về mô hình ngôn ngữ, chỉ cần xem chủ thể mà ta có thể đưa ra thông tin về một trạng từ đi sau đó. Ví dụ, nếu đầu ra của chủ thể là số ít hoặc số nhiều thì ta có thể biết được dạng của trạng từ đi theo sau nó phải như thế nào.

## c. Mạng Bidirectional LSTM (BiLSTM)

**Bidirectional LSTM (BiLSTM)** đã được tạo ra để khắc phục điểm yếu trên. Một kiến trúc BiLSTM thường chứa 2 mạng LSTM đơn được sử dụng đồng thời và độc lập để mô hình hoá chuỗi đầu vào theo 2 hướng: từ trái sang phải (forward LSTM) và từ phải sang trái (backward LSTM)

# 3. Triển khai mô hình

## 3.1.Thu thập dữ liệu

Bộ thu thập dữ liệu : Selenium

Crawl là một vấn đề hay gặp trong quá trình làm software. Ví dụ lấy tin tức, tin giảm giá, vé xem phim... là những dạng của crawl. Một cách khá đơn giản đó là phân tích HTML, đọc các thẻ và rút trích dữ liệu

Tuy nhiên việc crawl một trang bằng đọc HTML thuần sẽ không work được trong một số trường hợp như: dữ liệu được load bằng ajax(lúc đọc HTML sẽ chỉ thấy wrapper chứ không thấy dữ liệu) hay muốn vào được trang cần crawl thì phải qua bước login,…

Selenium hỗ trợ việc thu thập dữ liệu cả trong trường hợp như thế

Ngoài ra Selenium còn hỗ trợ các phần tử cấu trúc file HTML để truy nhập vào các phần tử HTML và thu thập dữ liệu hay thực hiện những sự kiện như chúng ta thao tác bằng tay 1 cách tự động như click, kéo chuột xuống , đúp chuột … để có thể thu thập dữ liệu 1 cách tự động và thu thập được nhiều dữ liệu hơn

Với dữ liệu trong bài này nhóm sử dụng là bình luận của người dùng về món ăn trên trang foody.vn

Thông qua selenium sử dụng xpath để truy cập tới các phần tử cấu trúc trong file HTML thực hiện việc thu thập

Để thu được dữ liệu huấn luyện cần phải phân ra 2 loại ứng với 2 nhãn “tích cực” và “ tiêu cực” ta tiến hành thu thập như sau:

* Sử dụng hàm xpath() cùng với 1 xâu cú pháp chỉ ra cấu trúc đi tới và chọn được thẻ html chứa dữ liệu của binh luận và thu thập chúng
* Trong Foody ứng với mỗi bình luận sẽ là phần điểm người dùng cho món ăn đó, dựa đó để thu thập dữ liệu và chia chúng thành 2 file ứng với 2 nhãn cụ thể vd: nếu điểm <4 thì bình luận đó sẽ là tiêu cực, điểm >8 bình luận đó là tích cực.Cũng có nhiều bình luận mà có điểm dưới 8 là tích cực nhưng cũng có 1 vài bình luận giả sử có điểm 6 là tiêu cực , vì thu thập là tự động lên chúng ta sẽ chọn ngưỡng <4 và>8 là ngưỡng đủ an toàn
* Trong foody để chánh phải tải 1 lúc nhiều bình luận lên trang html mà họ đã để 1 button loadmore , vì vậy nếu chỉ lấy dữ liệu ở các thẻ html chứa phần bình luận ta sẽ thu thập rất chậm vì thế nên ở đây ta sử dụng sự kiện click trong selenium để click button này và thu thập toàn bộ chúng

## 3.2.Chuẩn bị dữ liệu cho mô hình

Trước khi đi vào xây dựng lên mô hình của các mạng ( phần sau) ta phải tiến hành embeding cho 1 câu bình luận mà chúng ta sẽ đưa vào huấn luyện trong mô hình

### a. Tách từ cho tiếng Việt

Vì trong tiếng Việt có tồn tại từ ghép , mà trong đó có những loại từ ghép mà khi tách các từ đơn của nó ra sẽ không có mang nghĩa gì cả hoặc nghĩa bị lệch lạc đi

Có nhiều cách giải quyết được vấn đề này như thuật toán Longest Matching, CRF,…

Và rất may mắn đã có thư viện tách từ tiếng Việt pyvi của thầy Trần Việt Trung

### b. Biểu diễn từ trong không gian (word embeding)

Định nghĩa word embeddings (tạm dịch là **tập nhúng từ**):

**Tập nhúng từ** là tên chung cho một tập hợp các mô hình ngôn ngữ và các phương pháp học đặc trưng trong xử lý ngôn ngữ tự nhiên (NLP), nơi các từ hoặc cụm từ từ vựng được ánh xạ tới vectơ số thực. Về mặt khái niệm, nó liên quan đến việc nhúng toán học từ một không gian với một chiều cho mỗi từ vào một không gian vectơ liên tục với kích thước thấp hơn nhiều

Cũng có nhiều phương pháp để biểu diễn từ dưới dạng vector nhưng nổi trội thì có word2vec

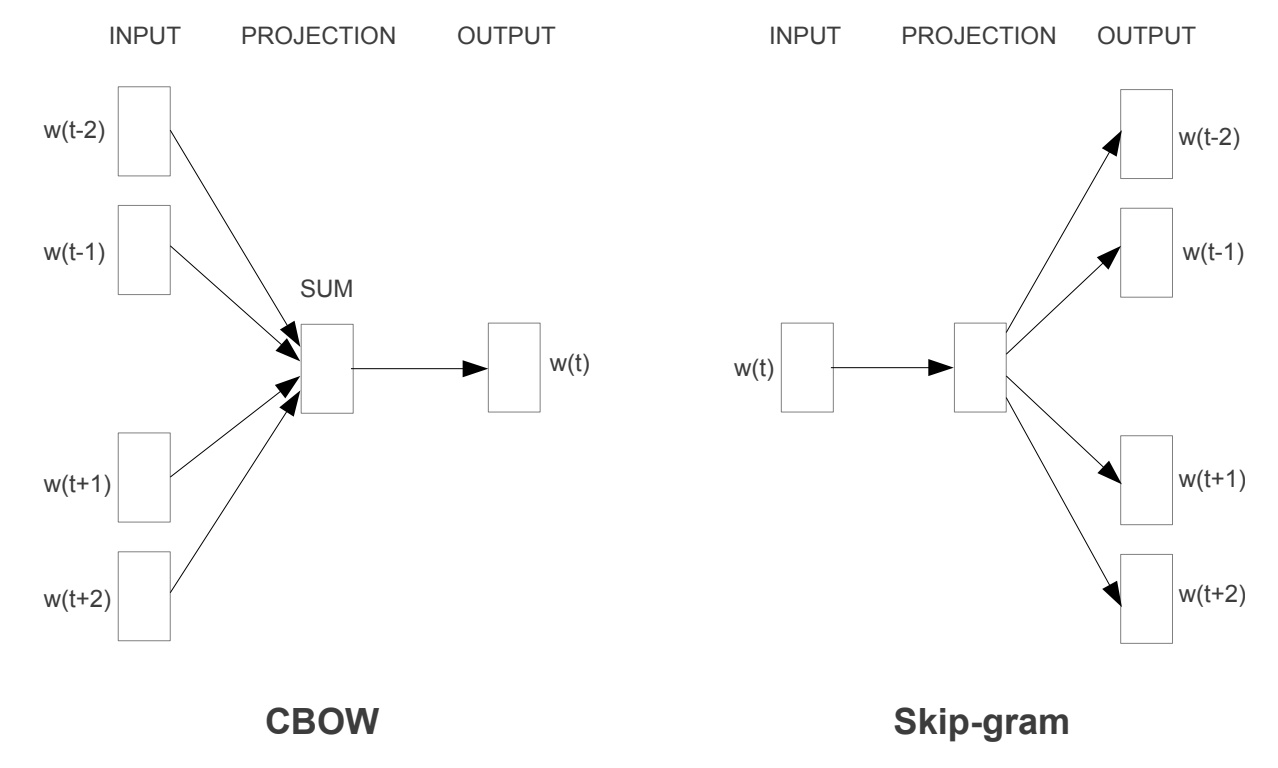
Ý tưởng chính của **word2vec** là:

* Thay vì lưu thông tin xuất hiện của các từ bằng cách đếm trực tiếp như **ma trận đồng xuất hiện**, word2vec học để đoán từ lân cận của tất cả các từ.
* Các giải pháp sau đó như **Glove** cũng tương tự word2vec được đề xuất bởi nhóm Pennington năm 2014.
* Tính toán nhanh hơn và dễ dàng thêm dữ liệu mới vào trong mô hình

Phương pháp:

1. Đoán các từ lân cận trong cửa số m của mỗi từ 1.1. Với mỗi từ t = 1 … T 1.2. Đoán các từ trong cửa sổ bán kính m của tất cả các từ
2. Hàm mục tiêu (objective function): tối ưu hợp lý hoá cực đại của bất kỳ **từ** ngữ cảnh (context word) đối với một **từ** đang xét hiện tại (center word). J(θ)=−1T∏Tt=1∏−m≤j≤m,j≠0p(wt+j|wt;θ)

Các kiến trúc khác nhau: (1) cho ngữ cảnh đoán từ hiện tại (CBoW), và (2) cho từ hiện tại đoán ngữ cảnh (Skip-gram).



**Biểu diễn 1 câu bình luận**: Dựa vào model pretrained của word2vec ta có được biểu diễn của từng từ trong 1 câu ( các từ này là từ đã được tách như phần trước đã đề cập),ta quy định 1 độ dài tối đa cho 1 câu giả sử là max\_length(quy định để cho các ma trận đồng nhất nhau về các cột) , vậy ứng với 1 câu ta thu được 1 ma trận max\_length x số chiều của vector biểu diễn từ. max\_length là số từ tối đa trong 1 câu mà ta chọn. Nếu câu dài hơn ta sẽ cắt đi, nếu câu ngắn hơn ta sẽ thêm vào vector 0. Dữ liệu features X gồm n câu (sentences) sẽ được biểu diễn thành 1 tensor 3 chiều n x max\_length x d; với n = số mẫu (hay số câu bình luận), max\_length = chiều dài 1 câu (nhớ chọn là 1 số cố định), d = số chiều không gian vector biểu diễn 1 từ.

Khi đã đưa được mỗi từ (word) thành một vector, mỗi câu thành một ma trận, dữ liệu đầu vào trở thành 1 tensor 3 chiều, thì chúng ta có thể sử dụng các mô hình deeplearning để classify.

## 3.3.Mạng LSTM

**Xây dựng model LSTM:**

Để đơn giản, chúng ta sử dụng thư viện Keras, bạn chỉ cần chọn các hyper-parameters như số layer, số neuron trên 1 layer, kỹ thuật Dropout, hay chọn thuật toán tối ưu…

Với Keras, chúng ta có thể dễ dàng lưu và load model để sử dụng lại nhiều lần.

Đây là model LSTM xây dựng sau khi đã thử-sai và lựa chọn ra các tham số mô hình khả quan

model = Sequential()

model.add(LSTM(64,return\_sequences=**True**,input\_shape=input\_dim)) model.add(Dropout(0.2))

model.add(LSTM(32))

model.add(Dense(self.n\_class,activation="softmax")) model.compile(loss=keras.losses.categorical\_crossentropy, optimizer=keras.optimizers.Adadelta(), metrics=['accuracy']) **return** model

Mô hình là 1 mô hình đơn giản liên tiêp nhau gồm có:

* Lớp 1 là LSTM với 64 node
* Lớp thứ 2 Dropout để loại bớt đi sự tham gia của các node
* Lớp thư 3 là LSTM với 32 node
* Lớp cuối là Dense với activation softmax cho ra kết quả dự đoán

## 3.4.Mạng Bi-LSTM

Xây dựng tương tự LSTM ta được :

model = Sequential()

model.add(Bidirectional(LSTM(32,return\_sequences=True),input\_shape=

input\_dim))

model.add(Dropout(0.1))

model.add(Bidirectional(LSTM(16)))

model.add(Dense(self.no\_class, activation="softmax"))

model.compile(loss=keras.losses.categorical\_crossentropy,

optimizer=keras.optimizers.Adadelta(),

metrics=['accuracy'])

## 3.5. Xây dựng nhận diện cảm xúc

* Từ tập dữ liệu có được từ thu thập qua selenium xây dựng nên tập train
* Đưa tập train vao tách từ cho từng câu trong đó, và đưa từng từ được tách vào biểu diễn vector word2vec
* Sau đó đua chúng vao model để traning và sinh ra mô hình được huấn luyện sẵn

## 3.6. Xây dựng nhận diện khía cạnh

Tương tự như nhận diện cảm xúc nhưng gồm nhiều lớp cần gán nhãn hơn

## 3.7. Xây dựng nhận diện khía cạnh và cảm xúc song song

Trong 1 bình luận của Foody có nhắc tới rất nhiều khía cạnh và cảm xúc về mỗi khía cạnh đó là đa dạng, ta muốn xây dựng nhận diện cảm xúc cho từng khía cạnh

Khi đó việc nhận diện đơn thuần là khó khăn và dẫn tới sai lệch

**Giải pháp**: Cần có được tập các câu từ 1 bình luận, mà mỗi câu biểu thị 1 ý nhất định (mệnh đề) ,mục đích tách ra các câu nhỏ thuộc duy nhất 1 khía cạnh

* Sử dụng các dấu hiệu để tách câu theo thứ tự :

+Tách câu theo dấu chấm

+Tách câu theo dấu chấm phẩy

+Tách câu theo liên tư như : và , nhưng, mà,còn

* Sau đó sử dụng POS\_tag của mỗi câu đã được tách để kiểm tra xem mỗi câu đó có là mệnh đề không,nếu có thì thêm vào tập câu được tách, nếu không ghép lại với câu liền trước nó-> Ta thu được 1 tập các câu đơn mà nó chỉ thuộc 1 khía cạnh
* Đưa các câu này vào nhận diện khía cạnh nếu câu nào ở cùng khía cạnh thì ghép chúng lại với nhau thành 1 , sau đó ta thu được các câu, mà mỗi câu chỉ ở vao nhiều nhất 1 khía cạnh
* Đưa các câu được ghép đó vào nhận diện cảm xúc và ta thu được cảm xúc của khía cạnh trong bình luận của người dùng

# Kết quả đạt được

|  |  |  |
| --- | --- | --- |
|  | loss | accuracy |
| Sentiment | 0.03 | 0.9809 |
| Aspect | 0.05 | 0.9902 |

# Các khó khăn gặp phải trong quá trình thực hiện đồ án và cách thức giải quyết

Đầu tiên là khó khăn về mặt đội nhóm. Nhóm chúng em gồm 4 thành viên, nhưng có những bạn kỳ này mới quen nhau. Trong nhóm, các bạn có lịch học rất khác nhau, đến cuối tuần nhiều bạn phải đi làm thêm hoặc về quê phụ giúp ba mẹ việc nhà nên việc họp nhóm là khá khó khăn. Chúng em đã khắc phục khó khăn này bằng cách ứng dụng công nghệ thông tin trong việc họp nhóm, ví dụ như sử dụng Facebook để liên lạc, và github để chia sẻ mã nguồn cho nhau. Việc ứng dụng những công nghệ ấy đã giúp chúng em rất nhiều trong quá trình làm đồ án, để lại cho bọn em nhiều kinh nghiệm để tiếp tục làm đồ án của các môn học khác.

Khó khăn tiếp theo là về vấn đề kiến thức. Vì đây là lần đầu bọn em được có cơ hội tìm hiểu những kiến thức về khai phá web, nên nhiều phần kiến thức bọn em còn chưa hiểu rõ hoặc áp dụng chưa đúng. Có nhiều những phần kiến thức mà đây là lần đầu chúng em sử dụng đến, như là crawl dữ liệu từ các trang web, lập trình bằng ngôn ngữ python, sử dụng các thư viện trong học máy như keras, tokenize, word2vec,… Thời gian làm thì có thể nhanh, số lượng code có thể ít thế nhưng thời gian chiếm đa phần chính là tìm hiểu và áp dụng chúng vào bài toán cụ thể. Kèm theo đó là trong quá trình cài đặt cũng như lập trình vướng mắc khá nhiều vào thời gian fix bug, sửa các lỗi liên quan đến mô hình, chỉnh sửa lại các thông số để đạt được kết quả đầu ra tốt nhất.

Khi bắt đầu làm bài, nhóm đã chọn đề tài đánh giá các comment trong các sản phẩm trên trang foody.vn. Khó khăn đầu tiên nhóm gặp phải là phải crawl comment trên trang foody. Các comment trên foody do người dùng đánh giá mà cho điểm, nhóm chúng em dự định sẽ lấy điểm để gán nhãn cho dữ liệu. Việc đánh giá mốc để xác định là tích cực (positive) hay tiêu cực (negative) cũng mất khá nhiều thời gian. Các comment trên foody thì có tỷ lệ positive:negative khá chênh lệch, vì vậy lượng dữ liệu cần crawl phải đủ lớn mới có thể đủ số lượng train được. Vấn đề tiếp theo nhóm gặp phải là chia khía cạnh cho từng comment. Mỗi comment lại có thể có nhiều khía cạnh khác nhau, vì vậy nhóm phải chia ra cho từng thành viên để có thể tách khía cạnh từ các comment đã crawl được.

# Kết luận và các đề cử cho việc tiếp tục phát triển và cải tiến trong tương lai

* 1. *Kết luận*
     1. *Ưu điểm*

Chương trình nhận diện khía cạnh khá chính xác.

Thời gian nhận diện tương đối nhanh đã có thể đưa ra kết quả.

* + 1. *Nhược điểm*

Những comment có nhiều khía cạnh chương trình chưa thể phát hiện và hiển thị được toàn bộ

* 1. *Các đề cử cho việc tiếp tục phát triển và cải tiến trong tương lai*

Dựa trên những ưu nhược điểm của chương trình hiện tại, nhóm em dự kiến tiếp tục phát triển chương trình trong tương lai. Mục tiêu của nhóm là:

* Áp dụng thuật toán xử lý ngôn ngữ tự nhiên (cụ thể là tiếng Việt), áp dụng trong nhận diện khía cạnh, để hệ thống có thể xác định được đối với các comment có nhiều khía cạnh
* Hệ thống hiện tại mới chỉ train được dữ liệu khá nhỏ, vì vậy kết quả đưa ra vẫn còn sai khác. Nếu có thêm thời gian, nhóm sẽ crawl/tìm được tập dữ liệu lớn hơn để train, giúp cải thiện kết quả đạt được.

**KẾT LUẬN**

Thông qua việc tìm hiểu và nghiên cứu đề tài này chúng em có cái nhìn toàn diện hơn trong việc đánh giá, thống kê các bình luận của người dùng nói về các sản phẩm trên trang foody.vn

Chúng em nhận thấy chương trình của mình còn rất nhiều thiếu sót, mong nhận được sự giup đỡ và đóng góp của thầy và các bạn để chương trình của chúng em được hoàn thiện hơn.

Bằng việc nghiên cứu về đề tài, chúng em thấy rằng khai phá web là một lĩnh vực khó nhưng rất hay và thú vị, cùng với đó việc áp dụng cả xử lý ngôn ngữ tự nhiên và deeplearning trong việc đánh giá bình luận sản phẩm giúp chúng em hoàn thiện bản thân hơn về những kiến thức mà mình có thể sử dụng trong tương lai.