ĐẠI HỌC QUỐC GIA HÀ NỘI
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ

PHẠM HÙNG

HƯỚNG TIẾP CẬN DỰA TRÊN HỌC MÁY CHO BÀI TOÁN TRÍCH XUẤT THÔNG TIN QUAN ĐIỂM

Ngành: Công nghệ thông tin
Chuyên ngành: Kỹ thuật phần mềm
Mã số: 60480103

LUẬN VĂN THẠC SĨ CÔNG NGHỆ THÔNG TIN

NGƯỜI HƯỚNG DẪN KHOA HỌC: TS. NGUYỄN VĂN VINH

HÀ NỘI - 2017

LỜI CAM ĐOAN

Tôi là Phạm Hùng, học viên lớp Kỹ Thuật Phần Mềm K21 xin cam đoan báo cáo luận văn này được viết bởi tôi dưới sự hướng dẫn của thầy giáo, tiến sĩ Nguyễn Văn Vinh. Tất cả các kết quả đạt được trong luận văn này là quá trình tìm hiểu, nghiên cứu của riêng tôi. Trong toàn bộ nội dung của luận văn, những điều được trình bày là kết quả của cá nhân tôi hoặc là được tổng hợp từ nhiều nguồn tài liệu khác. Các tài liệu tham khảo đều có xuất xứ rõ ràng và được trích dẫn hợp pháp.

Tôi xin hoàn toàn chịu trách nhiệm và chịu mọi hình thức kỷ luật theo quy định cho lời cam đoan của mình.

Hà Nội, ngày ….. tháng …… năm 2017

Người cam đoan

Phạm Hùng

MỤC LỤC

BẢNG CÁC TỪ VIẾT TẮT

Viết tắt	Đầy đủ	Ý nghĩa
RNN	Recurrent Neural Network	Mạng neural hồi quy
ANN	Artificial Neural Network	Mạng neural nhân tạo
NLP	Natural Language Processing	Xử lý ngôn ngữ tự nhiên
LSTM	Long short-term memory	Mạng neural cải tiến giải quyết vấn đề phụ thuộc từ quá dài
CNN	Convolutional Neural network	Mạng neural tích chập
SVM	Support Vector Machine	Máy vector hỗ trợ

TÓM TẮT NỘI DUNG

Mạng neural hồi quy RNN được áp dụng rất rộng rãi trong các bài toán xử lý ngôn ngữ tự nhiên NLP. Do mạng hồi quy RNN mô hình hóa được bản chất của dữ liệu trong NLP như đặc tính chuỗi và sự phụ thuộc lẫn nhau giữa các thành phần theo thứ tự. Ngoài ra, do năng lực tính toán của máy tính ngày càng mạnh mẽ nên đã thực hiện hóa được việc huấn luyện mạng neural hồi quy nhiều tham số vốn yêu cầu nhiều bước tính toán hơn so với mạng neural thông thường. Do đó, việc áp dụng mạng RNN có thể coi là một bước đột phá trong xử lý ngôn ngữ.

Luận văn sẽ trình bày về lý thuyết mạng neural RNN và cải tiến của nó là LSTM cùng với một số thuật toán học máy quan trọng trong quá trình xử lý dữ liệu ngôn ngữ. Cuối cùng, luận văn sẽ mô tả việc áp dụng và kết quả khi sử dụng mô hình LSTM trong bài toán trích xuất thông tin quan điểm. Thuật toán sẽ được đánh giá dựa trên hai tập dữ liệu tiếng Anh và tiếng Việt.

MỞ ĐẦU

Trong thời đại hiện nay, nhằm phục vụ cho nhu cầu cuộc sống ngày càng cao của con người, các sản phẩm và dịch vụ cũng có bước phát triển rất mạnh mẽ. Có thể kể đến từ những sản phẩm đáp ứng nhu cầu thường ngày của con người như quần áo, sách, tạp chí, đồ dùng cá nhân cho đến những nhu cầu cao hơn về thị hiếu, du lịch, thẩm mĩ. Với mỗi loại sản phẩm và dịch vụ hiện tại cũng rất phong phú về chủng loại, chất lượng, cạnh tranh về giá cả tới từ nhiều nhà cung cấp khác nhau. Do đó, việc duy trì phát triển một sản phẩm dịch vụ có được mạng lưới người sử dụng rộng rãi đòi hỏi rất nhiều công sức. Một trong những phương pháp cơ bản và hiệu quả nhất là lắng nghe ý kiến phản hồi của khách hàng về sản phẩm dịch vụ. Dựa trên những ý kiến phản hồi này, nhà cung cấp sản phẩm dịch vụ có thể đánh giá được thị hiếu của sản phẩm, hiệu quả của chiến lược marketing quảng bá sản phẩm hay điều chỉnh sản phẩm phù hợp để đạt được hiệu quả kinh doanh tốt nhất. Công việc trên có tên gọi là trích xuất thông tin quan điểm của người dùng. Đây là bài toán cơ bản nhưng có ứng dụng rất lớn trong cuộc sống.

Cùng với sự phát triển của thiết bị di động và mạng internet, người dùng có rất nhiều kênh để tương tác với nhà cung cấp dịch vụ. Có thể kể đến các kênh truyền thống như email, điện thoại, fax cho đến các hình thức mới hơn như viết phản hồi trên các trang mạng xã hội, viết bài review sản phẩm, phản hồi ngay trên trang giới thiệu sản phẩm hay trên các diễn đàn. Từ các nguồn kể trên, dữ liệu được thu thập lại dưới dạng văn bản. Từ dữ liệu dạng văn bản, luận văn sẽ trình bày phương pháp áp dụng học máy để xử lý thông tin văn bản nhằm trích xuất được thông tin quan điểm của người dùng.

Luận văn của tôi được chia thành các phần sau:

Chương 1: Trình bày tổng quan về bài toán trích xuất thông tin quan điểm và một số khái niệm liên quan. Đồng thời, tôi trình bày những thách thức của việc trích xuất thông tin quan điểm sử dụng mô hình học máy.

Chương 2: Trình bày các phương pháp và một số thuật toán sử dụng cho bài toán trích xuất thông tin quan điểm. Trong đó, tôi sẽ trình bày kỹ về mô hình mạng Recurrent Neural Network (RNN), mô hình tiên tiến đang được áp dụng cho việc xử lý thông tin dạng chuỗi như văn bản.

Chương 3: Trình bày việc áp dụng mô hình RNN cho bài toán phân tích quan điểm.

Chương 4: Kết quả một số thử nghiệm.

Chương 5: Kết luận.

CHƯƠNG 1: TỔNG QUAN VỀ BÀI TOÁN

1.1 Khái niệm quan điểm

1.2 Bài toán trích xuất thông tin quan điểm

Bài toán trích xuất thông tin quan điểm dựa trên các thông tin phản hồi của người sử dụng nhằm phân loại phản hồi đó là tích cực hay tiêu cực. Thông tin phản hồi của người dùng được tổng hợp dưới dạng văn bản từ nhiều nguồn khác nhau như trên trang bán hàng, Facebook, hệ thống chợ của Google hay Apple. Dựa trên đánh giá của người dùng, kết quả của chiến lược marketing hay quảng bá sản phẩm được xác định là có hiệu quả hay không.

Bài toán trích xuất thông tin quan điểm (sentiment analysis) là một lĩnh vực nghiên cứu về các ý kiến, quan điểm, đánh giá, thái độ và cảm xúc của con người về một đối tượng. Trích xuất thông tin quan điểm thu hút được sự quan tâm lớn của cộng đồng nghiên cứu nói chung và cộng đồng xử lý ngôn ngữ tự nhiên nói riêng bởi hai yếu tố:

Thứ nhất, do sự bùng nổ thông tin và mạng xã hội nên con người có thể tự do chia sẻ ý kiến cảm nghĩ. Trong lịch sử loài người, đây là thời điểm lượng thông tin nói chung và thông tin về ý kiến quan điểm nói riêng phát triển rất nhanh và mạnh. Lượng thông tin chia sẻ trên mạng xã hội là khổng lồ . Nhận thấy rằng nếu có thể khai thác thông tin từ lượng dữ liệu khổng lồ này thì sẽ cho phép khai phá rất nhiều thông tin quan trọng giúp xác định và giải quyết nhiều vấn đề. Đơn cử như có thể dự đoán, định hướng xu thế của công nghệ, thời trang, tiêu dùng của xã hội.

Thứ hai, sự đa dạng và kết quả có thể thấy rõ khi áp dụng nó vào một số lĩnh vực như phân tích tâm lý người dùng, nghiên cứu thị trường. Ví dụ như trong kinh doanh, việc phân tích và nắm được các ý kiến phản hồi của người sử dụng, khách hàng sẽ giúp tổ chức, cá nhân nhận ra những điểm hạn chế của sản phẩm, dịch vụ mình cung cấp. Họ sẽ kịp thời có giải pháp khắc phục để đáp ứng được nhu cầu sử dụng của thị trường, nâng cao kết quả kinh doanh nhờ nắm bắt được thị hiếu và kênh chăm sóc khách hàng hiệu quả.

Quan điểm được chia làm chủ yếu là hai loại là tích cực (positive) và tiêu cực (negative). Ngoài ra trong một số trường hợp xét tới cả loại thứ ba là trung lập (neural).

1.3 Các hướng tiếp cận và giải quyết bài toán

1.3.1 Mô hình Support Vector Machine

1.3.2 K-nearest neighbors

CHƯƠNG 2: MẠNG NEURAL VÀ RNN

2.1 Mạng neural nhân tạo ANN

2.1.1 Mạng nơ-ron sinh học

2.1.2 Kiến trúc tổng quát của mạng neural nhân tạo

2.2 Mạng neural hồi quy RNN

Các mạng ANN không thể làm được điều này vì bản chất nó không mô phỏng khía cạnh thời gian. Giả sử bạn muốn phân loại sự kiện nào sẽ xảy ra ở một thời điểm trong bộ phim. Mạng ANN khó có thể được vận dụng để dự đoán được sự kiện xảy ra ở thời điểm cần xét mà không căn cứ vào những sự kiện trước trong phim. Mạng ANN cho các neural thành phần của lớp đầu vào, lớp ẩn và lớp đầu ra là độc lập về mặt thời gian. Trong khi đó, tính chất thời gian trước sau lại là đặc trưng của ngôn ngữ văn bản hay xử lý ngôn ngữ tự nhiên. .

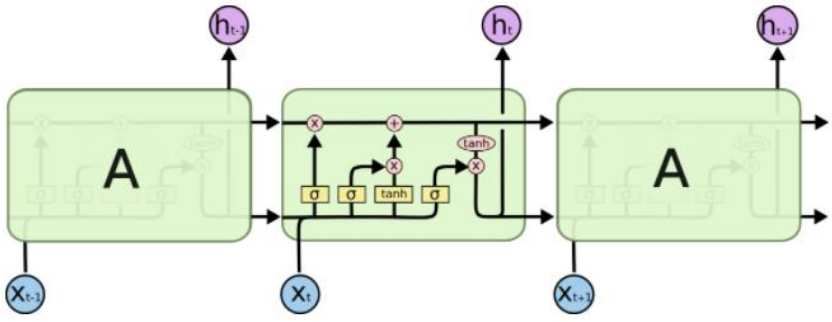
2.3 Vấn đề lưu trữ thông tin ngữ cảnh phụ thuộc lâu dài.

Trên lý thuyết, mạng RNN có thể phát sinh bộ nhớ đủ để xử lý vấn đề lưu trữ phụ thuộc dài. Tuy nhiên, trong thực tế thì không phải vậy. Vấn đề này đã được Hochreiter (1991) đưa ra như thách thức của mạng RNN. Và mạng Long short-term memory (LSTM) được phát biểu năm 1997 đã giải quyết được vấn đề này.

2.4. Mạng Long short-term memory

Long short term memory là cải tiến của mạng RNN nhằm giải quyết vấn đề học, lưu trữ thông tin ngữ cảnh phụ thuộc dài. tôi cùng xem xét cách LSTM [9] cải tiến hơn so với mạng RNN. Trong mô hình RNN, tại thời điểm t thì giá trị của vector ẩn ht chỉ được tính bằng một hàm tanh

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. Christopher Olah [10] đã có cách giải thích rất cụ thể về cách hoạt động của RNN.



Hình 2.1 Module lặp của mạng LSTM

RNN CHO BÀI TOÁN TRÍCH XUẤT QUAN ĐIỂM

3.1 Bài toán trích xuất thông tin quan điểm sử dụng RNN

3.2 Một số phương pháp vector hóa từ

3.2.1 Bag of Words

3.2.2 TF-IDF

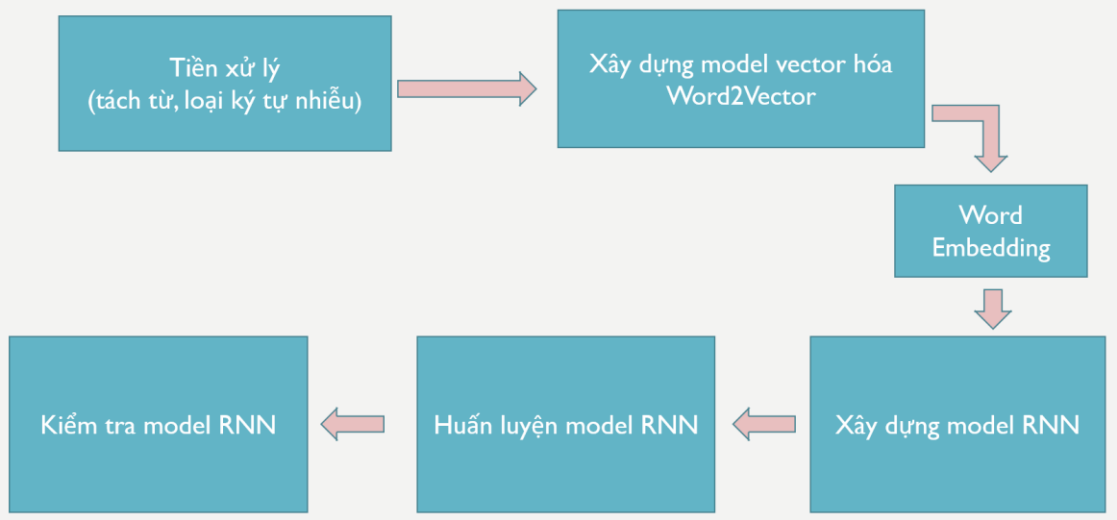
3.2.3 Word2vec

Giới thiệu

Chi tiết cách thực hiện

3.3. Áp dụng LSTM trong bài toán trích xuất thông tin quan điểm

Việc giải bài toán trích xuất thông tin quan điểm sẽ bao gồm việc giải quyết một chuỗi các bài toán nhỏ hơn. Chuỗi các bài toán nhỏ hơn này được gọi là pipeline của mô hình học máy.



Hình 2.2 Pipeline của bài toán trích xuất thông tin quan điểm sử dụng RNN

- Tiền xử lý kho ngữ liệu
- Xây dựng model vector hóa Word2vec cho tập ngữ liệu
- Word Embedding sử dụng mô hình kết quả của Word2vec để vector từng câu trong tập ngữ liệu
- Áp dụng mạng RNN để giải quyết bài toàn bao gồm các bước nhỏ: xây dựng model RNN, huấn luyện model RNN, kiểm tra model RNN

CHƯƠNG 3: KẾT QUẢ THỰC NGHIỆM

4.1 Bộ ngữ liệu

Luận văn sử dụng hai bộ ngữ liệu một tiếng Anh và một tiếng Việt được thu thập từ đánh giá của người dùng. Các kết quả thử nghiệm bao gồm việc turning các hyper parameter trong mô hình LSTM và cuối cùng là so sánh kết quả của LSTM với các thuật toán state-of-art sử dụng cả hai bộ ngữ liệu tiếng Việt và tiếng Anh.

4.1.1 Bộ ngữ liệu tiếng Anh (Food Reviews)

Bộ ngữ liệu tiếng Anh là bộ Food Reviews lấy dữ liệu từ Amazon [17]. Dữ liệu được thu thập trong 10 năm, bao gồm 568.454 đánh giá về sản phẩm đồ ăn trên trang thương mại điện từ Amazon. Dữ liệu bao gồm cả thông tin sản phẩm, thông tin người dùng, xếp hạng ưa thích và phần dữ liệu văn bản ghi lại đánh giá của người dùng.

Dataset statistics	
Number of reviews	568,454
Number of users	256,059
Number of products	74,258
Users with > 50 reviews	260
Median no. of words per review	56
Timespan	Oct 1999 - Oct 2012

Hình 3.1 Bộ ngữ liệu tiếng Anh

	Positive	**Neural**	**Negative**
Review/score	4-5	3	0-2
Số lượng đánh giá	443.777	42.640	82.037

Hình 3.2 Phân bố loại câu trong ngữ liệu tiếng Anh

Làm một vài khảo sát đối với tập dữ liệu này tôi có một số thông tin như sau: câu dài nhất là 1103 từ; trong đó độ dài câu gồm 13 từ có số lượng câu lớn nhất là 19166 câu. Tính được độ dài câu có mean = 35.29 và sigma = 31.76.

4.1.2 Bộ ngữ liệu tiếng Việt

Bộ ngữ liệu tiếng Việt gồm 5.100 nhận xét về sản phẩm tin học bao gồm 1.700 nhận của tích cực, tiêu cực và trung tính mỗi loại. Tập test bao gồm 1.050 nhận xét trong đó gồm 350 nhận xét mỗi loại. Câu dài nhất là có 2.716 từ và câu ngắn nhất có 1 từ. Trung bình số từ trên câu là 28,4 từ.

Tích cực	**Trung tính**	**Tiêu cực**
1.700	1.700	1.700

4.2 Cài đặt và thử nghiệm

Các thử nghiệm được cài đặt sử dụng ngôn ngữ python [16] trên môi trường python 3.6. Một số thư viện của python sử dụng trong thực nghiệm gồm:

Thư viện	
Numpy	Thư viện xử lý mảng, ma trận thực hiện các phép tính như nhân ma trận, tính ma trận chuyển vị …
Re	Thư viện về biểu thức chính quy Regular Expression
Pandas	Đọc dữ liệu lớn
Sklearn	Thư viện hỗ trợ cài đặt các thuật toán cơ bản như SVM, ANN
Gensim	Thư viện hỗ trợ cài đặt mô hình Word2vec
TensorFlow	Thư viện rất mạnh cho học máy hỗ trợ cài đặt mô hình, huấn luyện và kiểm thử mô hình
Matplotlib	Thư viện vẽ các loại đồ thị và hình

4.2.1 Bước tiền xử lý

Tiền xử lý là bước quan trọng không kém so với các bước xây dựng mô hình toán. Theo Andrew Ng [8] tiền xử lý tốt mang lại kết quả tốt không ngờ cho toàn mô hình. Tại bước tiền xử lý, tôi chủ yếu thực hiện việc loại bỏ những ký tự HTML, những ký tự không phải là chữ cái. Hàm loại bỏ các ký tự nhiễu đầu vào là một phản hồi khách hàng và đầu ra là phản hồi đã được làm mịn. Mã python của hàm loại bỏ ký tự nhiễu có dạng:

```
def clean_sentence(sentence):
    # Remove HTML
    review_text = BeautifulSoup(sentence).text

    # Remove non-letters
    letters_only = re.sub("[^a-zA-Z]", " ", review_text)
    return letters_only
```

Tiếp đó, tôi thực hiện loại bỏ những từ stopword trong phản hồi

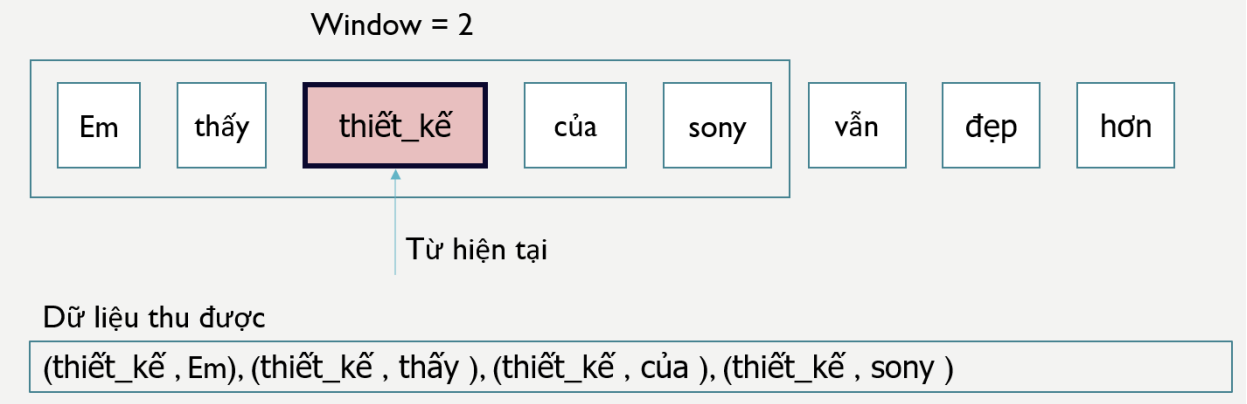
```
def review_to_words(review):
    """
    Function to convert a raw review to a string of words
    :param review
    :return: meaningful_words
    """
    # 1. Convert to lower case, split into individual words
    words = review.lower().split()
    #
    # 2. In Python, searching a set is much faster than searching
    #   a list, so convert the stop words to a set
    stops = set(stopwords.words("english"))
    #
    # 3. Remove stop words
    meaningful_words = [w for w in words if not w in stops]
    #
    # 4. Join the words back into one string separated by space,
    # and return the result.
    return " ".join(meaningful_words)
```

Đối với bộ ngữ liệu tiếng Việt cần thêm bước tách từ, ở đây có thể dùng một số công cụ tách từ có sẵn như Đông Du [3] của tác giả Lưu Tuấn Anh.

4.2.2 Xây dựng model Word2vec

Từ mảng các phản hồi đã được tiền xử lý, thực hiện xây dựng mô hình Word2vec. Mô hình Word2vec xây dựng một từ điển các từ và giá trị ánh xạ vector cho từ đó.

Khi đưa một câu vào, dựa trên giá trị window tôi sẽ tách được các cặp từ mô tả sự xuất hiện của từ hiện tại với từ xung quanh. Giả sử đối với câu “Em thấy thiết kế của sony vẫn đẹp hơn”, hình dưới đây mô tả việc lấy các cặp từ để đưa vào huấn luyện khi từ hiện tại là “thiết kế”.



Hình 3.3 Cách lấy cặp từ đưa vào huấn luyện Word2vec

Bản chất huấn luyện Word2vec sẽ dựa vào tần suất xuất hiện của các cặp từ để dự đoán từ tiếp theo trong câu. Từ đó, tính toán tối ưu hàm mất mát và cập nhật các tham số feature của từ. Xây dựng model word2vec sử dụng thư viện Gensim như sau.

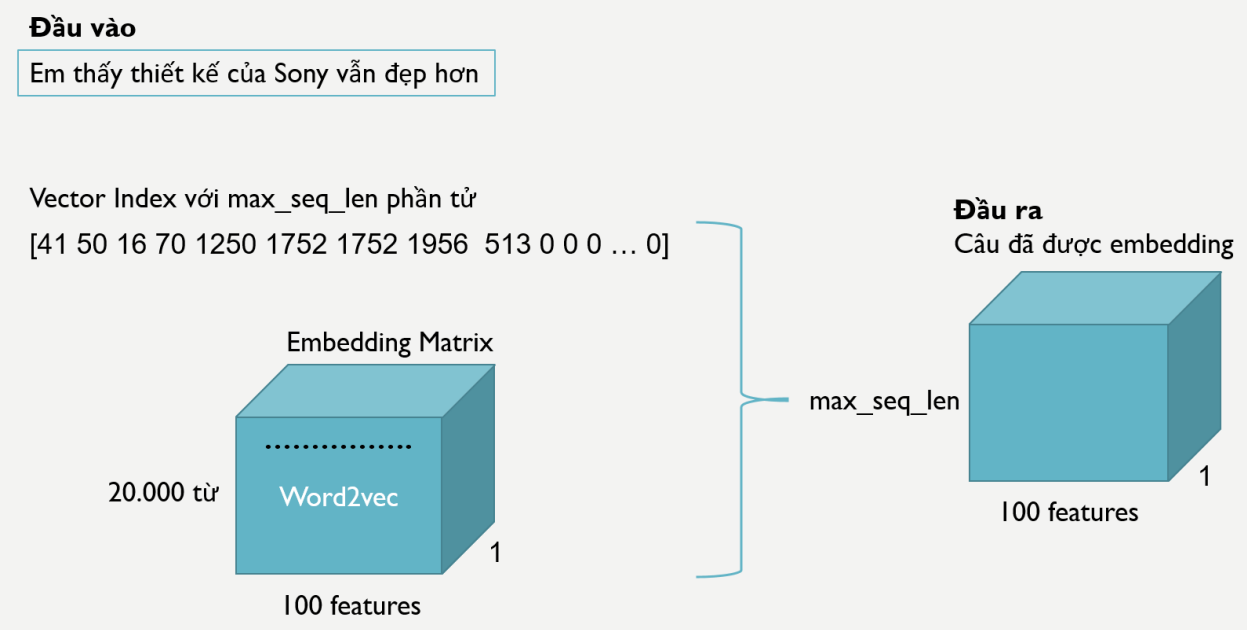
```
from gensim.models import Word2vec
model = Word2vec(doc, size=100, window=10, min_count=3, workers=4, sg=1);
model.save("food.w2v")
```

- min_count: giá trị ngưỡng của từ. Những từ có tần suất xuất hiện lớn hơn min_count mới được đưa vào mô hình word2vec
- Window: giá trị của cửa sổ từ. Tại vị trí hiện tại của từ đang xét sẽ ghi nhận giá trị window từ đứng trước và đứng sau từ hiện tại.
- Size: số lượng feature mong muốn
- Sg: sử dụng thuật toán CBOW hoặc skip-model để huấn luyện

4.2.3 Word Embedding

Word Embedding là quá trình đưa các từ trong câu về dạng để mô hình toán có thể hiểu được. Cụ thể là từ dạng text, các từ sẽ được chuyển về dạng vector đặc trưng để đưa vào mô hình LSTM. Trước khi đưa về dạng vector các câu cần được chuẩn hóa về độ dài. Chọn max_seq_len là độ dài của câu, khi đó tất cả các câu trong tập huấn luyện đều được cắt hoặc nối để có độ dài max_seq_len.

Khi một câu được đưa vào, trước tiên nó sẽ được embedding theo số index tương ứng của nó trong từ điển. Sau đó, dựa trên từ điển và kết quả word2vec thu được tôi embedding toàn bộ câu dưới dạng ma trận như hình dưới đây.



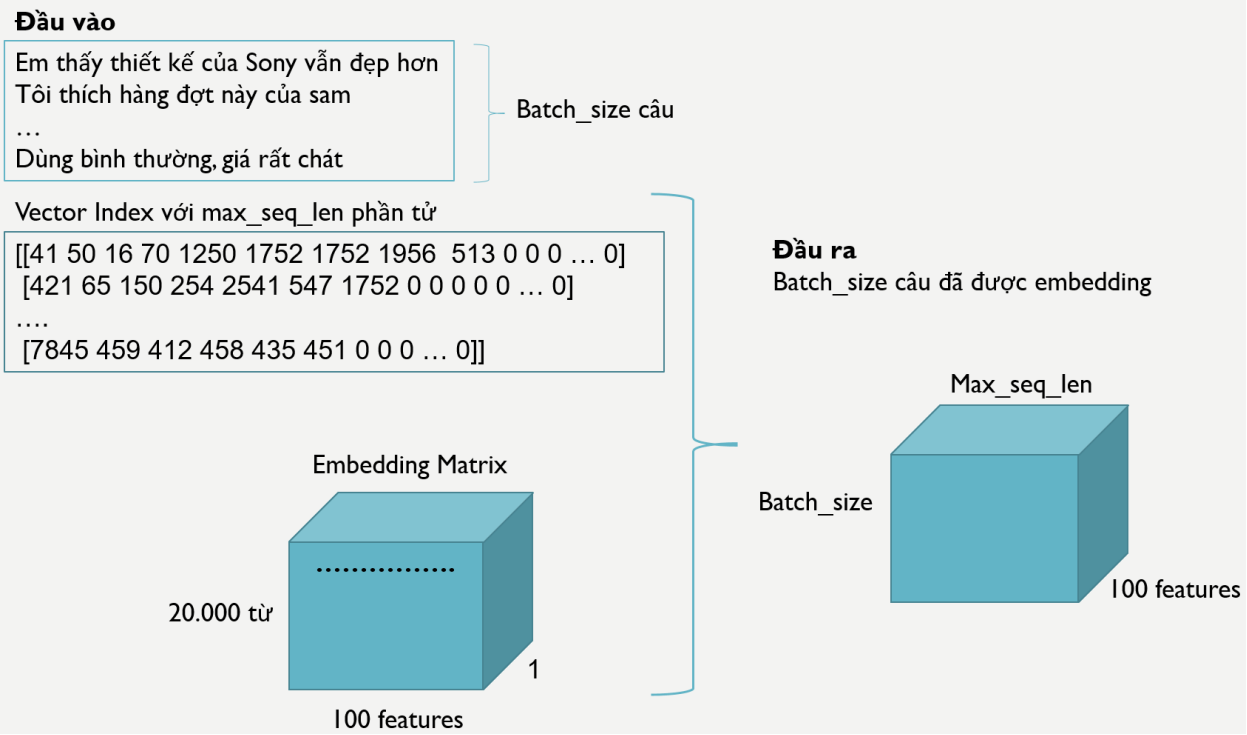
Hình 3.4 Quá trình word embedding của 1 câu

Tương ứng nhãn của câu cũng được embedding theo bảng sau

Tích cực	[1,0,0]
Trung tính	[0,1,0]
Tiêu cực	[0,0,1]

4.2.4 Huấn luyện mô hình LSTM

Huấn luyện mô hình tôi sẽ đưa vào mô hình batch_size số câu trong một lượt huấn luyện. Cách đưa vào batch_size chứ không đưa toàn bộ mô hình dựa trên tư tưởng của thuật toán Mini-batch Gradient Decent. Thuật toán sẽ lấy ngẫu nhiên và không lặp lại batch_size bộ dữ liệu từ tập huấn luyện. Mô tả quá trình word embedding với batch_size câu như sau.



Hình 3.5 Đưa batch_size câu vào mô hình huấn luyện

Để xây dựng mô hình LSTM tôi sử dụng thư viện TensorFlow [18], một mã nguồn mở rất mạnh trong học máy hiện đang được nhiều hãng lớn như Google sử dụng trong các sản phẩm thương mại. Trước tiên, tôi cần tạo TensorFlow graph. Để xây dựng TensorFlow graph, tôi định nghĩa một số siêu tham số (hyperparameter) như batch_size, số lượng LSTM units, số lượng vòng lặp khi train.

```
vocab_size = 20000
batch_size = 512
lstm_units = 64
iterations = 100000
```

Đối với TensorFlow graph, tôi định nghĩa 2 placeholders dữ liệu và nhãn dựa trên số chiều của ma trận tương ứng.

```
import TensorFlow as tf
tf.reset_default_graph()

labels = tf.placeholder(tf.float32, [batch_size, numClasses])
input_data = tf.placeholder(tf.int32, [batch_size, max_seq_len])
data = tf.Variable(tf.zeros([batch_size, max_seq_len, num_feature]),dtype=tf.float32)
data = tf.nn.embedding_lookup(wordVectors,input_data)
```

Sử dụng hàm embedding_lookup cho việc embedding batch_size câu đầu vào. Số chiều của data sẽ là (batch_size x max_seq_len x num_feature). tôi đưa data vào mô hình

LSTM bằng việc sử dụng hàm tf.nn.rnn_cell.BasicLSTMCell. Hàm BasicLSTMCell đầu vào là 1 siêu tham số lstm_units là số lượng units trong layer của LSTM. Tham số này phải được tinh chỉnh phù hợp đối với mỗi tập dữ liệu để đạt kết quả tốt nhất. Ngoài ra, khi huấn luyện mô hình mạng neural, tôi nên dropout bớt các tham số để tránh mô hình bị overfitting.

```
lstmCell = tf.contrib.rnn.BasicLSTMCell(lstm_units)
lstmCell = tf.contrib.rnn.DropoutWrapper(cell=lstmCell, output_keep_prob=0.75)
value, _ = tf.nn.dynamic_rnn(lstmCell, data, dtype=tf.float32)
```

Việc mô hình hóa LSTM tôi có nhiều cách để xây dựng. tôi có thế xếp chồng nhiều lớp LSTM lên nhau, khi đó vector ẩn cuối cùng của lớp LSTM thứ nhất sẽ là đầu vào của lớp LSTM thứ 2. Việc xếp chồng nhiều lớp LSTM lên nhau được coi là cách rất tốt để lưu giữ phụ thuộc ngữ cảnh xa lâu dài. Tuy nhiên vì thế số lượng tham số sẽ tăng gấp số lớp lần, đồng thời cũng tăng thời gian huấn luyện, cần thêm dữ liệu và dễ bị overfitting. Trong khuôn khổ của các tập dữ liệu thu thập được trong luận văn, tôi sẽ không xếp chồng các lớp LSTM vì những thử nghiệm với nhiều lớp LSTM không hiệu quả và gây overfitting. Đầu ra của mô hình LSTM là một vector ẩn cuối cùng, vector này được thay đổi để tương ứng với dạng vector kết quả đầu ra bằng cách nhân với ma trận trọng số.

```
weight = tf.Variable(tf.truncated_normal([lstm_units, numClasses]))
bias = tf.Variable(tf.constant(0.1, shape=[numClasses]))
value = tf.transpose(value, [1, 0, 2])
last = tf.gather(value, int(value.get_shape()[0]) - 1)
prediction = (tf.matmul(last, weight) + bias)
```

Tính toán độ chính xác (accuracy) dựa trên kết quả dự đoán của mô hình và nhãn. Kết quả dự đoán mô hình càng giống với kết quả nhãn thực tế thì mô hình càng có độ chính xác cao.

```
correctPred = tf.equal(tf.argmax(prediction,1), tf.argmax(labels,1))
accuracy = tf.reduce_mean(tf.cast(correctPred, tf.float32))
```

Kết quả dự đoán của mô hình không phải luôn luôn giống nhãn, đó gọi la lỗi. Để huấn luyện mô hình tôi cần tối thiểu hóa giá trị lỗi này. Định nghĩa một hàm tính lỗi cross entropy và một layer softmax sử dụng thuật toán tối ưu Adam với learning_rate được lựa chọn như một siêu tham số.

```
loss = tf.reduce_mean(tf.nn.softmax_cross_entropy_with_logits(logits=prediction,
labels=labels))
optimizer = tf.train.AdamOptimizer(learning_rate=0.0001).minimize(loss)
```

Lưu trữ độ chính xác và giá trị hàm lỗi qua từng vòng lặp khi huấn luyện sửa dụng tensorboard.

```
sess = tf.InteractiveSession()
saver = tf.train.Saver()
tf.summary.scalar('Loss', loss)
tf.summary.scalar('Accuracy', accuracy)

logdir = "tensorboard/" +"dict="+str(vocab_size) + "_maxSeq=" + str(maxSeqLength) +
"_batch=" + str(batchSize) + "_dimens=" + str(numDimensions) + "/"
writer = tf.summary.FileWriter(logdir, sess.graph)
merged = tf.summary.merge_all()
```

Thực hiện các thử nghiệm với mô hình LSTM có rất nhiều loại tham số cần turning thay đổi đối với mỗi tập dữ liệu. Ví dụ như lựa chọn giá trị learning_rate, lựa chọn hàm tối ưu, số lượng units LSTM, kích thước từ điển, số lượng đặc trưng của từ, số vòng lặp thực hiện huấn luyện LSTM … Dựa trên rất nhiều thử nghiệm, tôi sẽ rút ra được một số tham số ảnh hưởng nhiều hay ít đến kết quả thực hiện huấn luyện. Từ đó, tôi có thể rút ra được nhiều kết luận bổ ích của thực nghiệm.

4.2.5 Cài đặt một số phương pháp học có giám sát kinh điển

Việc cài đặt một số thuật toán như SVM, KNN có vai trò so sánh kết quả đối với thuật toán LSTM mà tôi đã xây dựng. Để cài đặt các thuật toán này, tôi có thể sử dụng thư viện sklearn [20] rất dễ dàng sau khi dữ liệu đã được word embedding.

4.3 Kết quả trích xuất thông tin quan điểm

4.3.1 Một số thử nghiệm và kết quả trên bộ ngữ liệu tiếng Anh

Việc huấn luyện mô hình LSTM cho kết quả đầu ra phụ thuộc vào nhiều yếu tố như các siêu tham số. Khi thay đổi các tham số để tối ưu cho mô hình, tôi sẽ phải làm rất nhiều các thử nghiệm. Để đánh giá được một hay vài tham số có ý nghĩa hơn so với các tham số khác tôi sẽ thực hiện tinh chỉnh và căn cứ vào đường học (Learning Curve) để đánh giá. Những thử nghiệm trong luận văn, tôi đã lựa chọn những tham số có ý nghĩa về mặt ngôn ngữ để đánh giá. Chi tiết tôi chia bộ dữ liệu tiếng Anh làm 2 tập train và test theo tỉ lệ 60/40 và thực hiện các thử nghiệm như sau.

Thử nghiệm 1: Giữ số lượng từ vựng bằng 20000 (vocab_size = 20000)

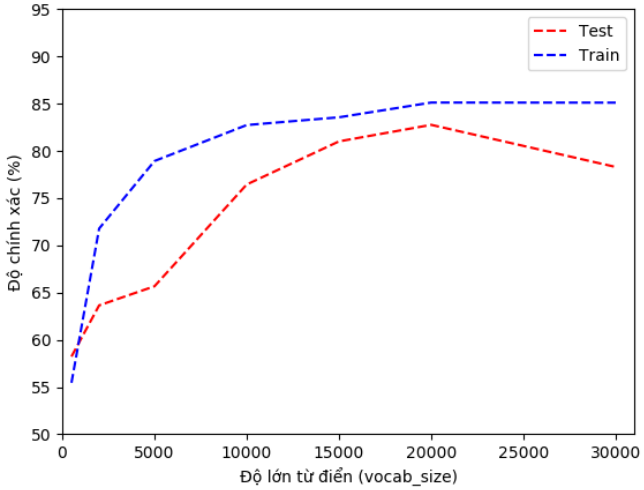
Số lượng từ của tập ngữ liệu được tính toán ở trên là 50.538, tuy nhiên tôi thử chọn 20.000 từ được sử dụng nhiều nhất để làm từ điển. Thay đổi độ dài cho phép của câu đầu vào (max_seq_len). Max_seq_len có tác dụng truncate chuỗi các câu đầu vào thành câu có độ dài là max_seq_len, trong đó những câu có độ dài nhỏ hơn được điền tiếp 1 số ký tự đặc biệt và câu có độ dài lớn hơn thì được cắt đi chỉ còn độ dài max_seq_len

Max_seq_len	**Độ chính xác (Train)**	**Độ chính xác (Test)**
25	84.23 %	75.57 %
50	85.12 %	82.76 %
80	82.11 %	80.82 %
110	81.31 %	78.23 %
140	77.57 %	79.85 %

Nhận xét, số lượng từ vựng không đổi thì max_seq_len cho kết quả tốt nhất với độ dài bằng 50 từ. Với số từ bằng 50 tương ứng với trên 80% câu trong tập mẫu do đó tôi thấy giá trị này đại diện khá tốt cho độ dài của câu.

Thử nghiệm 2: Giữ độ dài từ mỗi câu là 50 từ

Giữ max_seq_len = 50, thay đổi độ lớn của từ điển. Thay đổi độ lớn của từ điển ảnh hưởng khá lớn đến kết quả bởi nếu số lượng từ nhỏ sẽ có quá nhiều từ trong tập mẫu sẽ không có trong từ điển; nếu số lượng lớn thì số lượng từ được nhận ra sẽ nhiều khi sử dụng word2vec với số lượng đặc trưng lớn (khoảng 300) thì độ phức tạp tính toán sẽ tăng lên rất nhiều.



Hình 3.6 Thử nghiệm với độ dài câu bằng 50 từ

Thử nghiệm 3: So sánh với một số phương pháp khác

Các phương pháp được so sánh gồm KNN, SVM, Gaussian, ANN. Kết quả cho thấy sử dụng LSTM cho kết quả khá khả quan.

Thuật toán	**Độ chính xác**	
	Train	Test
Nearest Neighbors accuracy	74.63%	78.32%
Linear SVM accuracy	79.55%	81.82%
Gaussian Process accuracy	79.52%	79.68%
Neural Net accuracy	79.52%	79.12%
LSTM	85.12%	82.76%

Hình 3.7 Kết quả trên bộ ngữ liệu tiếng Anh

4.3.2 Một số thử nghiệm và kết quả trên bộ ngữ liệu tiếng Việt

Thuật toán	**Độ chính xác**	
	Train	Test
Nearest Neighbors accuracy	55.7%	38.5%
Linear SVM accuracy	56.9%	40.5%
Gaussian Process accuracy	62.3%	42.9%
Neural Net accuracy	73.3%	41.3%
LSTM	87.83%	43.7%

Hình 3.8 Kết quả trên bộ ngữ liệu tiếng Việt

Bộ ngữ liệu tiếng Việt hiện tại có số lượng câu còn ít, ngoài ra có rất nhiều từ bị viết tắt, viết sai theo các cách khác nhau. Ví dụ như để chỉ "không" – tập dữ liệu có các từ "ko","k","khog". Khi áp dụng những thuật toán như word2vec để tính toán word embedding thường cho số lượng tham số lớn dễ gây hiện tượng overfitting.

Kết quả tốt nhất hiện ghi nhận sử dụng vocab_size = 2000, max_seq_len = 20, số feature của word2vec bằng 50, tuy nhiên vẫn bị overfitting.

4.4 Nhận xét

Kết quả trên bộ ngữ liệu tiếng Anh là khá tốt, kết quả khi sử dụng model LSTM cho kết quả tốt hơn so với các thuật toán SVM, KNN, Gaussian hay ANN. Trong tập dữ liệu tiếng Anh đã chọn một số tham số như sau

- Số feature of vector = 128
- Dropout = 0.8
- Activation = 'softmax'
- Optimizer = 'adam'
- Learning_rate = 0.001

Kết quả bộ ngữ liệu tiếng Việt bị overfitting. Hiện tượng này xảy ra khi độ chính xác trên tập train tốt nhưng độ chính xác trên tập test lại rất thấp. Nguyên nhân được xác định là do bộ ngữ liệu tiếng Việt có số lượng mẫu ít, khi train trong mạng neural có nhiều tham số rất không tốt và hay dẫn đến overfitting. Việc này không thể cải thiện kể cả khi dropout thêm. Sau khi quan sát bộ ngữ liệu tiếng Việt thì thấy có rất nhiều từ là tên riêng (Ví dụ: iphone, asus) hay viết tắt (Ví dụ: k thay cho không) dù đã loại bỏ stopword. Đây thực sự là thách thức trong việc thu thập dữ liệu tự nhiên đặc biệt bằng tiếng Việt.

CHƯƠNG 4: KẾT LUẬN

Mạng neural LSTM có thể được sử dụng rộng rãi trong bài toán xử lý ngôn ngữ tự nhiên như sentiment analysis. Đặc biệt là có thể tận dụng được ưu điểm của việc xử lý dạng chuỗi và thứ tự các từ trong câu. Tuy nhiên, các nghiên cứu LSTM cho sentiment analysis chưa tận dụng được đầy đủ các tài nguyện về sentiment như Sentiment lexicon, từ phủ định hay từ chỉ mức độ.

Với việc định nghĩa max_seq_len thì cách làm này là chấp nhận được đối với tập ngữ liệu mà luận văn sử dụng. Tập ngữ liệu là tập phản hồi của người dùng có số lượng từ không lớn hơn 100. Do đó, có thể xem xét việc lấy max_seq_len số từ đưa vào LSTM để huấn luyện là có thể tổng quát hóa được câu cần xét. Tuy nhiên, đối với tập phản hồi có số từ lớn hơn thì tôi phải xem xét việc vector hóa mà không làm mất mát quá nhiều ý nghĩa của câu do việc chọn đại diện max_seq_len không là không đủ để đại diện cho câu. Một phương pháp thường được sử dụng là dùng TF-IDF kết hợp với một thuật toán giảm số chiều như LDA (Linear Discriminant Analysis).

LSTM là một mô hình kỹ thuật hiệu quả trong bài toán xử lý chuỗi và hiện đang được các nhà nghiên cứu sử dụng rất nhiều. Tuy nhiên, LSTM không phải là một kỹ thuật vạn năng mà cứ bài toán về NLP là lại áp dụng được. Nó còn căn cứ vào nhiều yếu tố như tập ngữ liệu, đặc tính của tập ngữ liệu. Vì đôi khi sử dụng một thuật toán ML lại cho kết quả tốt hơn như SVM, Decision Tree hay ANN.

Nhận thấy rằng, những nghiên cứu gần đây sử dụng các phương pháp học máy và Deep Learning giống như trận sóng thần áp đảo trong NLP. Tuy nhiên, người làm vẫn nên trú trọng bổ sung các kiến thức về ngôn ngữ học và semantic. Bởi ngoài việc trong một vài trường hợp, việc sử dụng một vài rule là cách giải quyết tối ưu nhất so với việc train một mô hình ngôn ngữ đồ sộ. Mà nhờ các kiến thức về ngôn ngữ học, người nghiên cứu có thể cân nhắc được mô hình NLP tốt nhất có thể giải quyết bài toán cũng như biểu diễn đầu vào bằng những đặc trưng có ý nghĩa.

TÀI LIỆU THAM KHẢO

Tiếng Việt

[1] Bùi Công Cường, Nguyễn Doãn Phước (2001). Hệ mờ, mạng nơ-ron và ứng dụng. Nhà xuất bản Khoa học và kỹ thuật. Hà Nội.

[2] Vũ Hữu Tiệp, Blog Machine Learning Cơ bản tại địa chỉ https://machinelearningcoban.com/

[3] Lưu Tuấn Anh (2012), Bộ tách từ Đông Du https://github.com/rockkhuya/DongDu

Tiếng Anh

[4] Hochreiter and Schmidhuber (1997), Long short-term memory

[5] B. Liu (2009), Handbook Chapter: Sentiment Analysis and Subjectivity. Handbook of Natural Language Processing, Handbook of Natural Language Processing. Marcel Dekker, Inc. New York, NY, USA.

[6] B.Liu (2015), Sentiment analysis: mining sentiments, opinions and emotions, Cambridge University Press, ISBN 9781107017894

[7] Tomas Mikolov, Kai Chen, Greg Corrado, Jeffrey Dean (2013), Efficient Estimation of Word Representations in Vector Space In Proceedings of Workshop at ICLR.

[8] Andrew Ng, Machine Learning course on Coursera

[9] Christopher Olah (2015), Understanding LSTM networks in Colah's blog

[10] Andrej Karpathy (2015), The Unreasonable Effectiveness of Recurrent Neural Network at Andrej Karpathy blog

[11] McCormick, C. (2016). Word2vec Tutorial - The Skip-Gram Model.

[12] Google (2013), Word2vec model https://code.google.com/archive/p/word2vec/

[13] J. McAuley and J. Leskovec (2013), From Amateurs to Connoisseurs: Modeling the Evolution of User Expertise through Online Reviews

[14] The statistic of social media usage (2014) http://thesocialskinny.com/103-crazy-social-media-statistics-to-kick-off-2014/

[15] Kishori K. Pawar, Pukhraj P Shrishrimal, R. R. Deshmukh (2015) Twitter Sentiment Analysis: A Review ISSN 2229-5518

[16] Python Programming Language https://www.python.org/

[17] Jure Leskovec, Web data Amazon Fine Foods reviews (2014) https://snap.stanford.edu/data/web-FineFoods.html

[18] TensorFlow https://www.TensorFlow.org/

[19] Scikit Learn http://scikit-learn.org/stable/