**Bài 1 (5đ):**

*Tìm hiểu và trình bày về các biểu diễn token theo phương pháp BPE (Byte-Pair Encoding). Cho các ví dụ về các mô hình có sử dụng tokenizer theo BPE theo format sau:*

- Định nghĩa:

- Các khái niệm liên quan đến BPE:

- Các bước hoạt động BPE:

- Ưu điểm so với các phương pháp biểu diễn token truyền thống

- 2 ví dụ về các mô hình có sử dụng tokenizer theo BPE

- Dẫn tài liệu tham khảo theo format IEEE

*So sánh 2 mô hình sử dụng BPE và không sử dụng BPE sử dụng ngôn ngữ python trên jupter. Phân mô hình không sử dụng BPE tôi đã xây dựng xong. Lưu ý 2 mô hình này phải được train và test trên cùng một bộ dữ liệu kaggle theo format sau:*

- Mô tả bài toán

- Xây dựng bài toán (chi tiết)

- Đánh giá so sánh 2 mô hình

*So sánh 2 mô hình trên một bài toán cụ thể nào đó có sử dụng BPE và không sử dụng BPE sử dụng ngôn ngữ python trên jupter. Lưu ý 2 mô hình này phải được train và test trên cùng một bộ dữ liệu kaggle theo format sau:*

*1.4.1 Mô tả bài toán*

*Bài toán được sử dụng để so sánh hiệu suất của hai mô hình NLP là Phân loại tình cảm (Sentiment Analysis) trên tập dữ liệu "Sentiment Analysis from IMDB Reviews" được cung cấp bởi Kaggle. Mục tiêu của bài toán là phân loại đánh giá phim trong IMDB thành hai loại: tích cực và tiêu cực.*

*1.4.2. Tiền xử lí dữ liệu*

# Cài đặt thư viện Kaggle

!pip install kaggle

# Tải lên tệp kaggle.json của bạn chứa thông tin xác thực API

from google.colab import files

files.upload()

# Di chuyển tệp đã tải lên vào thư mục cần thiết

!mkdir -p ~/.kaggle

!mv kaggle.json ~/.kaggle/

# Thiết lập quyền truy cập

!chmod 600 ~/.kaggle/kaggle.json

# Bây giờ bạn có thể tải tập dữ liệu bằng lệnh API Kaggle

!kaggle datasets download -d lakshmi25npathi/imdb-dataset-of-50k-movie-reviews

# Giải nén tập dữ liệu

!unzip imdb-dataset-of-50k-movie-reviews.zip

# Tải tập dữ liệu từ Kaggle

df = pd.read\_csv("/content/IMDB Dataset.csv")

df.head()

df['sentiment'].value\_counts()

data = df.head(1000)

data["sentiment"].value\_counts()

from bs4 import BeautifulSoup

import nltk

nltk.download('stopwords')

# Tải stopwords của NLTK

stop\_words = set(stopwords.words('english'))

def preprocess\_text(text):

    # Loại bỏ thẻ HTML

    text = BeautifulSoup(text, "html.parser").get\_text()

    # Loại bỏ ký tự đặc biệt và số

    text = re.sub(r'[^a-zA-Z]', ' ', text)

    # Chuyển đổi thành chữ thường

    text = text.lower()

    # Tách từ

    tokens = word\_tokenize(text)

    # Loại bỏ stopwords

    tokens = [word for word in tokens if word not in stop\_words]

    # Kết hợp các từ lại thành một câu

    clean\_text = ' '.join(tokens)

    return clean\_text

import nltk

nltk.download('punkt')

# Áp dụng tiền xử lý vào cột 'review'

data['review'] = df['review'].apply(preprocess\_text)

*1.4.3 Xây dựng mô hình*

from sklearn.feature\_extraction.text import TfidfVectorizer #

vectorizer = TfidfVectorizer(max\_features=5000)

X = vectorizer.fit\_transform(data['review']).toarray()

y = data['sentiment']

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

data

*1.4.3.1 Mô hình 1: Không sử dụng BPE (sử dụng word2vec)*

from sklearn.svm import SVC

from sklearn.model\_selection import GridSearchCV

# Thiết lập các siêu tham số tốt nhất thủ công

best\_svm\_classifier = SVC(C=1, kernel='rbf')

# Huấn luyện mô hình với dữ liệu huấn luyện của bạn

best\_svm\_classifier.fit(X\_train, y\_train)

# Dự đoán sử dụng mô hình tốt nhất

predictions = best\_svm\_classifier.predict(X\_test)

accuracy = accuracy\_score(y\_test, predictions)

print(f"Độ chính xác sử dụng phương pháp Bag of Words: {accuracy}")

# Vector semantics sử dụng Word2Vec

from gensim.models import Word2Vec, KeyedVectors

#'review' là cột chứa các đánh giá đã được xử lý trước

data['tokenized'] = data['review'].apply(lambda x: word\_tokenize(x))

data

# Loại bỏ các câu đã được mã hóa thành các từ riêng lẻ

tokenized\_sentences = data['tokenized'][data['tokenized'].apply(len) > 0]

# Huấn luyện mô hình Word2Vec

word2vec\_model = Word2Vec(sentences=tokenized\_sentences, vector\_size=100, window=5, min\_count=1, workers=4)

#Chuyển đổi đoạn văn thành vectors sử dụng Word2Vec (trung bình của các vectors từng từ)

def get\_vector(sentence):

    vectors = [word2vec\_model.wv[word] for word in sentence if word in word2vec\_model.wv]

    if not vectors:

        return np.zeros(word2vec\_model.vector\_size)

    return np.mean(vectors, axis=0)

# Giả sử tokenized\_sentences là một cột trong DataFrame của bạn

data['word2vec'] = data['tokenized'].apply(get\_vector)

data.head()

# Chia dữ liệu thành các tập huấn luyện và kiểm tra

X = np.vstack(data['word2vec'].to\_numpy())

y = data['sentiment']

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Khởi tạo bộ phân loại SVM

svm\_classifier\_word2vec = SVC(C=1, kernel='rbf')

# Huấn luyện mô hình với dữ liệu huấn luyện của bạn

svm\_classifier\_word2vec.fit(X\_train, y\_train)

# Dự đoán sử dụng mô hình đã huấn luyện

predictions\_word2vec = svm\_classifier\_word2vec.predict(X\_test)

# Tính toán độ chính xác

accuracy\_word2vec = accuracy\_score(y\_test, predictions\_word2vec)

print(f"Độ chính xác sử dụng các vectors Word2Vec: {accuracy\_word2vec}")

from gensim.models import Word2Vec #

from nltk.tokenize import word\_tokenize

import numpy as np

# Hàm tiền xử lý câu nhập để chuyển đổi thành vectơ đại diện

def preprocess\_input(sentence):

    tokenized\_sentence = word\_tokenize(sentence)

    vector = get\_vector(tokenized\_sentence) # Sử dụng hàm get\_vector đã định nghĩa trong đoạn mã trước

    return vector.reshape(1, -1)

# Nhập câu từ bàn phím

input\_sentence = input("Nhập câu: ")

# Tiền xử lý câu nhập để chuyển đổi thành vectơ đại diện

input\_vector = preprocess\_input(input\_sentence)

# Dự đoán nhãn của câu nhập sử dụng mô hình SVM đã huấn luyện

predicted\_label = svm\_classifier\_word2vec.predict(input\_vector)

# Khởi tạo bộ phân loại SVM với probability=True

svm\_classifier\_word2vec = SVC(C=1, kernel='rbf', probability=True)

# Huấn luyện mô hình với dữ liệu huấn luyện của bạn

svm\_classifier\_word2vec.fit(X\_train, y\_train)

# Dự đoán xác suất của từng nhãn cho câu nhập

predicted\_proba = svm\_classifier\_word2vec.predict\_proba(input\_vector)

# In nhãn dự đoán và xác suất tương ứng

print("Nhãn dự đoán cho câu nhập là:", predicted\_label[0])

print("Xác suất dự đoán:", predicted\_proba)

*1.4.3.2 Mô hình 2: Sử dụng BPE (sử dụng BERT)*

*1.4.3.3 So sánh hiệu suất 2 mô hình*

*Yêu cầu 1: Sử dụng các chỉ số đánh giá như độ chính xác (accuracy), độ nhạy (sensitivity), độ đặc hiệu (specificity), F1-score để so sánh hiệu suất của hai mô hình.*

*Yêu cầu 2: Khi nhập 1 câu từ bàn phím, hãy đánh giá đó là nhãn positive hay negative bằng cả 2 mô hình. Phân tích kết quả để xác định mô hình nào có hiệu suất tốt hơn trong việc phân loại đánh giá phim IMDB*

*(hãy xóa những dòng code dư thừa)*

**Bài làm:**

**1. Định nghĩa**

Xử lý ngôn ngữ tự nhiên (NLP) là một lĩnh vực con của trí tuệ nhân tạo mang lại khả năng hiểu máy và xử lý ngôn ngữ của con người. Tokenization là quá trình chia văn bản thành một tập hợp các token từ một chuỗi văn bản. Token hóa thường là bước đầu tiên trong các tác vụ xử lý ngôn ngữ tự nhiên như phân loại văn bản, nhận dạng thực thể được đặt tên và phân tích tình cảm. Các mã thông báo kết quả thường được sử dụng làm đầu vào cho các bước xử lý tiếp theo, chẳng hạn như vectơ hóa, trong đó các mã thông báo được chuyển đổi thành các biểu diễn số cho các mô hình học máy để sử dụng.

Mã hóa cặp byte (BPE) là một thuật toán nén được sử dụng trong Xử lý ngôn ngữ tự nhiên (NLP) để thể hiện vốn từ vựng lớn với một tập hợp nhỏ các đơn vị từ con. Nó được giới thiệu bởi Sennrich et al. vào năm 2016 và đã được sử dụng rộng rãi trong các tác vụ NLP khác nhau như dịch máy, phân loại văn bản và tạo văn bản. Ý tưởng cơ bản của BPE là hợp nhất lặp đi lặp lại cặp byte hoặc ký tự liên tiếp thường xuyên nhất trong một kho văn bản cho đến khi đạt được kích thước từ vựng được xác định trước. Các đơn vị từ con kết quả có thể được sử dụng để đại diện cho văn bản gốc một cách nhỏ gọn và hiệu quả hơn.

**2. Các khái niệm**

Các khái niệm liên quan đến BPE:

Vocabulary: Một tập hợp các đơn vị từ con có thể được sử dụng để đại diện cho một kho văn bản.

Byte: Một đơn vị thông tin kỹ thuật số thường bao gồm tám bit.

Nhân vật: Một biểu tượng đại diện cho một chữ cái hoặc chữ số bằng văn bản hoặc in.

Frequency (Tần số): Số lần một byte hoặc ký tự xảy ra trong một kho văn bản.

Merge: Quá trình kết hợp hai byte hoặc ký tự liên tiếp để tạo ra một đơn vị từ con mới.

**3. Các bước liên quan đến BPE**

- Bước 1: Khởi tạo từ vựng với tất cả các byte hoặc ký tự trong kho văn bản

- Bước 2: Tính tần suất của từng byte hoặc ký tự trong kho văn bản.

- Bước 3: Lặp lại các bước sau cho đến khi đạt được kích thước từ vựng mong muốn:

* Tìm cặp byte hoặc ký tự liên tiếp thường xuyên nhất trong kho văn bản
* Hợp nhất cặp để tạo một đơn vị từ con mới.
* Cập nhật số tần suất của tất cả các byte hoặc ký tự có chứa cặp đã phối.
* Thêm đơn vị từ con mới vào từ vựng.

- Bước 4: Đại diện cho kho văn bản bằng cách sử dụng các đơn vị từ con trong từ vựng.

**4. Cách hoạt động BPE**

**Bài 2 (5đ):** Chọn 1 trong 2 bài sau:

Bài 2a:

Tìm hiểu và xây dựng bài toán phát hiện lỗi và sửa lỗi chính tả tiếng Việt theo tiếp cận học sâu. Các nhiệm vụ cần thực hiện:

a) Chọn mô hình: mô hình chỉ có Encoder; hoặc mô hình sinh Encoder-Decoder; hoặc mô hình chỉ sử dụng Decoder

b) Xây dựng, sinh dữ liệu huấn luyện dựa trên các loại lỗi chính tả tiếng Việt thường gặp.

c) Huấn luyện mô hình. Có thể sử dụng Pre-trained model và thực hiện fine-tune hoặc tự train từ đầu.

d) Hãy đánh giá độ chính xác của mô hình

Bài 2b: Hãy xây dựng mô hình để tạo một trợ lý ảo (agent) làm nhiệm vụ chăm sóc khách hàng trong một lĩnh vực nào đó tự bạn chọn, ví dụ:

- Agent của một siêu thị cho cho giới thiệu các mặt hàng và trả lời câu hỏi của khách hàng

- Agent chăm sóc khách hàng cho khách sạn

- Agent chăm sóc khách hàng cho công ty du lịch

- … (tự bạn chọn lĩnh vực)

Các nhiệm vụ cần làm:

- Thu thập và xây dựng dữ liệu huấn luyện

- Lựa chọn mô hình huấn luyện

- Đánh giá độ chính xác của mô hình

Nộp bài: Đối với cả bài 1 và bài 2

- File Word chứa trình bày về lý thuyết

- File codes