TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM
TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG
KHOA CÔNG NGHỆ THÔNG TIN

ĐỒ ÁN CUỐI KÌ MÔN

NHẬP MÔN XỬ LÝ NGÔN NGỮ TỰ NHIÊN

Người hướng dẫn: **PGS.TS LÊ ANH CƯỜNG**

Người thực hiện: **LÊ GIA BẢO – 52000629**

NGUYỄN THÀNH CHẤT – 52000629

PHAN QUỐC TOÀN - 52000814

Nhóm **: 05**

Khoá **: 24**

THÀNH PHỐ HỒ CHÍ MINH, NĂM 2024

TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM
TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG
KHOA CÔNG NGHỆ THÔNG TIN

ĐỒ ÁN CUỐI KÌ MÔN

NHẬP MÔN XỬ LÝ NGÔN NGỮ TỰ NHIÊN

Người hướng dẫn: **PGS.TS LÊ ANH CƯỜNG**

Người thực hiện: **LÊ GIA BẢO - 52000629**

NGUYỄN THÀNH CHẤT - 52000636

PHAN QUỐC TOÀN - 52000814

Nhóm **: 05**

Khoá **: 24**

THÀNH PHỐ HỒ CHÍ MINH, NĂM 2024

LỜI CẢM ƠN

Chúng em xin gửi lời cảm ơn chân thành đến PGS.TS Lê Anh Cường vì sự giúp đỡ quý báu của Thầy trong suốt quá trình chúng em hoàn thành môn học này tại trường Đại học Tôn Đức Thắng. Những kiến thức và kinh nghiệm mà Thầy đã truyền đạt không chỉ giúp chúng em hiểu rõ hơn về môn học mà còn mang lại những bài học quý giá trong cuộc sống và công việc sau này. Sự nhiệt tình và tâm huyết của Thầy luôn là nguồn động lực lớn lao để chúng em cố gắng học tập và phấn đấu. Chúng em rất biết ơn vì sự hỗ trợ tận tâm của Thầy và hy vọng sẽ tiếp tục nhận được sự chỉ dẫn của Thầy trong tương lai.Kính chúc Thầy luôn dồi dào sức khỏe và thành công trong sự nghiệp.

ĐỒ ÁN ĐƯỢC HOÀN THÀNH
TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG

Tôi xin cam đoan đây là sản phẩm đồ án cuối kì của chúng tôi và được sự hướng dẫn của PGS.TS Lê Anh Cường;. 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 cuối kì 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 20 tháng 5 năm 2024

Tác giả

(ký tên và ghi rõ họ tên)

Lê Gia Bảo

Nguyễn Thành Chất

Phan Quốc Toàn

PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN

Phần xác nhận của GV hướng dẫn

_____

_____

_____

_____

_____

_____

_____

Tp. Hồ Chí Minh, ngày tháng năm
(kí và ghi họ tên)

Phần đánh giá của GV chấm bài

_____

_____

_____

_____

_____

_____

_____

Tp. Hồ Chí Minh, ngày tháng năm
(kí và ghi họ tên)

TÓM TẮT

Đồ án nghiên cứu tập trung vào việc khai thác và ứng dụng các kỹ thuật trong lĩnh vực xử lý ngôn ngữ tự nhiên (NLP). Trong chương đầu tiên, đề tài sẽ giới thiệu chi tiết về Byte Pair Encoding (BPE) - một phương pháp nén dữ liệu đơn giản nhưng hiệu quả, được sử dụng rộng rãi trong việc giảm kích thước từ vựng và cải thiện hiệu suất của các mô hình NLP. Chương này sẽ giải thích nguyên lý hoạt động của BPE, các bước thực hiện và những lợi ích mà BPE mang lại trong việc xử lý ngôn ngữ tự nhiên. Chương thứ hai sẽ tập trung vào việc xây dựng một mô hình chatbot chăm sóc khách hàng, ứng dụng BPE trong quá trình tiền xử lý dữ liệu. Chatbot này được thiết kế để tự động hóa và nâng cao chất lượng dịch vụ khách hàng, giúp giải quyết các thắc mắc và yêu cầu của khách hàng một cách nhanh chóng và hiệu quả.

MỤC LỤC

DANH MỤC CÁC BẢNG BIỂU, HÌNH VẼ, ĐỒ THỊ

DANH MỤC HÌNH

DANH MỤC BẢNG

CHƯƠNG 1 – GIỚI THIỆU BPE

1.1 Phương pháp BPE (Byte-Pair Encoding).

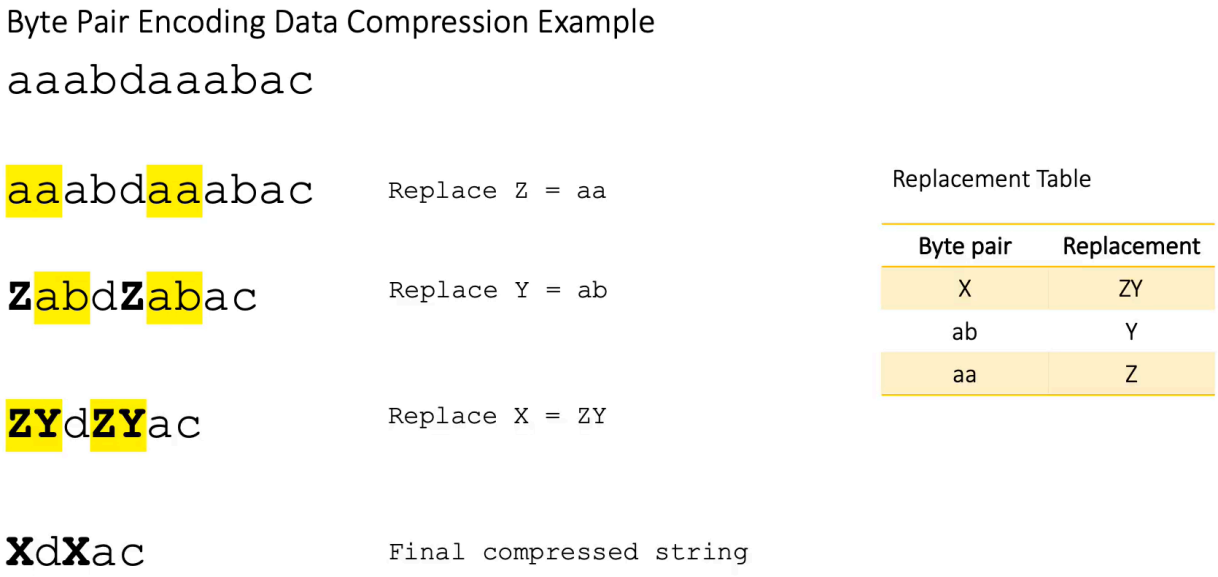
Byte Pair Encoding là một thuật toán nén dữ liệu được giới thiệu lần đầu tiên vào năm 1994, giúp tăng hiệu quả của tất cả các mô hình **NLP** tiên tiến hiện nay (bao gồm cả BERT). Mặc dù vậy, thuật toán này không phải ai cũng biết.

Những năm gần đây có thể nói là thời điểm vàng để tham gia vào lĩnh vực **NLP.** Các không gian véc tơ từ được huấn luyện như Word2vec và GloVe đã đặt nền tảng cho những thành công để máy tính có thể hiểu được ý nghĩa của các từ. Trong nhiều năm, chúng là cách biểu diễn đáng tin cậy trong việc huấn luyện các mô hình học máy trong **NLP** khi không có nhiều dữ liệu. Mặc dù vậy, chúng không phải là công cụ toàn năng khi đối mặt với những từ hiếm xuất hiện. Các từ này được thay thế bởi tokens <unk> khi cài đặt mô hình.

Để giải quyết các từ hiếm, chúng ta có giải pháp là biểu diễn văn bản dưới dạng tập hợp các ký tự. Các từ hiếm xét cho cùng vẫn được tạo nên từ những ký tự "không hiếm". Mặc dù vậy, các ký tự không có được sự mô tả ngữ nghĩa trọn vẹn như các từ. Đi tìm một cách biểu diễn dữ liệu trung hòa giữa biểu diễn dữ liệu bằng ký tự và từ, bước đột phá thực sự đầu tiên trong việc giải quyết vấn đề từ hiếm được thực hiện bởi các nhà nghiên cứu tại Đại học Edinburgh bằng cách sử dụng thành phần từ với Bye Pair Encoding (BPE). Ngày nay, phương pháp mã hóa này và các biến thể của nó trở thành chuẩn mực trong hầu hết các mô hình tiên tiến bao gồm cả BERT, GPT-2, RoBERTa, v.v.

Một số người gọi BERT là sự khởi đầu của một kỷ nguyên mới. Tuy nhiên, BPE cũng xứng đáng được tôn vinh mặc dù không nhiều người biết đến nó.
Về nguồn gốc, Byte Pair Encoding lần đầu tiên được Philip Gage giới thiệu trong bài viết "**A New Algorithm for Data Compression**", ấn bản tháng 2/1994 của C Users Journal.

Đây là là một kỹ thuật nén dữ liệu hoạt động bằng cách thay thế các cặp byte liên tiếp có tần suất lớn bằng một byte không tồn tại trong dữ liệu. Hình dưới là một ví dụ về cách hoạt động của BPE.



Hình 1.1: Ví dụ về Byte Pair Encoding (BPE)

Để thực hiện mã hóa sử dụng Subword, BPE được sửa đổi một chút. Các cặp Subword thường xuyên xuất hiện được hợp nhất với nhau chứ không bị thay thế như BPE với mục đích nén dữ liệu. Về cơ bản, các từ hiếm gặp sẽ được chia thành các Subword phổ biến hơn. Xuất phát từ việc biểu diễn tất cả các từ bằng các ký tự. BPE sẽ thống kê toàn bộ các cặp ký tự xuất hiện nhiều nhất và bổ sung Subword vào trong tập từ vựng. Quá trình này được lặp lại đến khi kích thước của tập từ vựng đạt đến kích thước mong muốn (đây là một siêu tham số).

BPE với mã hóa Subword là cách biểu diễn kết hợp được điểm mạnh của cả hai phương pháp mã hóa phổ biến trước đó. Với các Subword, chúng ta không gặp phải vấn đề

Out-of-Vocabulary và cũng không làm mất mát quá nhiều ngữ nghĩa như việc sử dụng mã hóa cấp ký tự.

Mã hóa theo cặp (BPE) tiền thân được phát triển như một thuật toán để nén văn bản, sau đó được OpenAI sử dụng để tokenize khi huấn luyện trước mô hình GPT. Nó được sử dụng bởi rất nhiều mô hình Transformer, bao gồm GPT, GPT-2, RoBERTa, BART và DeBERTa.

Huấn luyện BPE bắt đầu bằng cách tính toán tập hợp các từ duy nhất được sử dụng trong kho ngữ liệu (sau khi hoàn thành các bước chuẩn hóa và pre-tokenization), sau đó xây dựng từ vựng bằng cách lấy tất cả các ký hiệu được sử dụng để viết những từ đó. Ví dụ rất đơn giản, giả sử kho dữ liệu của chúng ta sử dụng năm từ sau:

`"hug", "pug", "pun", "bun", "hugs"`

Từ vựng cơ sở khi đó sẽ là ["b", "g", "h", "n", "p", "s", "u"]. Đối với các trường hợp trong thực tế, từ vựng cơ sở đó sẽ chứa tất cả các ký tự ASCII, ít nhất và có thể là một số ký tự Unicode. Nếu một mẫu bạn đang tokenize sử dụng một ký tự không có trong kho dữ liệu huấn luyện, thì ký tự đó sẽ được chuyển đổi thành token không xác định. Đó là một lý do tại sao nhiều mô hình NLP rất kém trong việc phân tích nội dung bằng biểu tượng cảm xúc.

GPT-2 và RoBERTa tokenizer (khá giống nhau) có một cách thông minh để giải quyết vấn đề này: chúng không xem các từ được viết bằng các ký tự Unicode mà là các byte. Bằng cách này, từ vựng cơ sở có kích thước nhỏ (256), nhưng mọi ký tự bạn có thể nghĩ đến sẽ vẫn được bao gồm và không bị chuyển đổi thành token không xác định. Thủ thuật này được gọi là *BPE cấp byte*.

Sau khi có được bộ từ vựng cơ bản này, chúng ta thêm các token mới cho đến khi đạt được kích thước từ vựng mong muốn bằng cách học *hợp nhất*, đây là các quy tắc để hợp nhất hai yếu tố của từ vựng hiện có với nhau thành một từ mới. Vì vậy, lúc đầu sự hợp nhất này sẽ tạo ra các token có hai ký tự và sau đó, khi quá trình huấn luyện tiến triển, các từ phụ sẽ dài hơn.

Tại bất kỳ bước nào trong quá trình huấn luyện token, thuật toán BPE sẽ tìm kiếm cặp token hiện có thường xuyên nhất (theo "cặp", ở đây có nghĩa là hai token liên tiếp trong một từ). Cặp thường xuyên nhất đó là cặp sẽ được hợp nhất, và chúng ta xả và lặp lại cho bước tiếp theo.

Quay trở lại ví dụ trước, giả sử các từ có tần số như sau:

("hug", 10), ("pug", 5), ("pun", 12), ("bun", 4), ("hugs", 5)

nghĩa là "hug" có mặt 10 lần trong kho ngữ liệu, "pug" 5 lần, "pun" 12 lần, "bun" 4 lần và "hug" 5 lần. Chúng ta bắt đầu huấn luyện bằng cách tách từng từ thành các ký tự (những ký tự hình thành từ vựng ban đầu của chúng ta) để có thể xem mỗi từ như một danh sách các token:

("h" "u" "g", 10), ("p" "u" "g", 5), ("p" "u" "n", 12), ("b" "u" "n", 4), ("h" "u" "g" "s", 5)

Sau đó, chúng ta xem xét các cặp. Cặp ("h", "u") có trong các từ "hug" và "hugs", vì vậy tổng cộng là 15 lần trong ngữ liệu. Tuy nhiên, đây không phải là cặp thường xuyên nhất: vinh dự đó thuộc về ("u", "g"), có trong "hug", "pug", và "hugs", với tổng cộng 20 lần xuất hiện trong bộ từ vựng.

Do đó, quy tắc hợp nhất đầu tiên được học bởi tokenizer là ("u", "g") -> "ug", có nghĩa là "ug" sẽ được thêm vào từ vựng và cặp này sẽ được hợp nhất trong tất cả các từ của ngữ liệu. Vào cuối giai đoạn này, từ vựng và ngữ liệu sẽ giống như sau:

Vocabulary: ["b", "g", "h", "n", "p", "s", "u", "ug"]

Corpus: ("h" "ug", 10), ("p" "ug", 5), ("p" "u" "n", 12), ("b" "u" "n", 4), ("h" "ug" "s", 5)

Bây giờ chúng ta có một số cặp dẫn đến một token dài hơn hai ký tự: ví dụ: cặp ("h", "ug"), (hiện diện 15 lần trong kho ngữ liệu). Cặp thường gặp nhất ở giai đoạn này là ("u", "n"), xuất hiện 16 lần trong kho ngữ liệu, vì vậy quy tắc hợp nhất thứ hai đã học là ("u", "n") -> "un". Thêm nó vào bộ từ vựng và hợp nhất tất cả các lần xuất hiện hiện có sẽ dẫn chúng ta đến:

Vocabulary: ["b", "g", "h", "n", "p", "s", "u", "ug", "un"]

Corpus: ("h" "ug", 10), ("p" "ug", 5), ("p" "un", 12), ("b" "un", 4), ("h" "ug" "s", 5)

Giờ thì cặp xuất hiện nhiều nhất là ("h", "ug"), nên chúng ta hợp nhất ("h", "ug") -> "hug", trả về cho chúng ta token gồm ba kí tự đầu tiên. Sau sự hợp nhất này, kho ngữ liệu sẽ như sau:

Vocabulary: ["b", "g", "h", "n", "p", "s", "u", "ug", "un", "hug"]

Corpus: ("hug", 10), ("p" "ug", 5), ("p" "un", 12), ("b" "un", 4), ("hug" "s", 5)

Và chúng ta tiếp tục làm vậy cho đến khi chúng ta chạm đến kích thước bộ tự điển ta mong muốn.

1.2 Các ví dụ về các mô hình có sử dụng tokenizer theo BPE.

1.2.1 Thuật toán tokenize

Tokenize tuân thủ chặt chẽ quá trình huấn luyện, theo nghĩa là các đầu vào mới được tokenize bằng cách áp dụng các bước sau:

1. Chuẩn hoá
2. Pre-tokenization
3. Tách các từ thành các ký tự riêng lẻ
4. Áp dụng các quy tắc hợp nhất đã học theo thứ tự trên các phần tách đó

Lấy ví dụ mà ta đã sử dụng trong quá trình huấn luyện, với ba quy tắc hợp nhất đã học:

```
("u", "g") -> "ug"
("u", "n") -> "un"

("h", "ug") -> "hug"
```

Từ "bug" sẽ được tokenize thành ["b", "ug"]. "mug", tuy nhiên, sẽ tokenizer thành ["[UNK]", "ug"] vì kí tự "m" không có trong bộ từ vựng gốc. Tương tự, từ "thug" sẽ được tokenize thành ["[UNK]", "hug"]: kí tự "t" không có trong bộ từ vựng gốc, và áp dụng quy tắc hợp nhất ở "u" và "g" và sau đó "hu" và "g".

1.2.2 Triển khai BPE

Hãy cùng xem các thuật toán BPE được triển khai. Đây không phải là phiên bản tối ưu mà bạn có thể thực sự sử dụng cho một kho ngữ liệu lớn; chúng tôi chỉ muốn cho bạn xem đoạn mã để bạn có thể hiểu thuật toán này tốt hơn.

Đầu tiên chúng ta cần một kho ngữ liệu, vậy nên hay tạo ra một bản đơn giản với một vài câu:

```
corpus = [
    "This is the Hugging Face Course.",
    "This chapter is about tokenization.",
    "This section shows several tokenizer algorithms.",
      "Hopefully, you will be able to understand how they are trained and generate tokens.",

]
```

Tiếp theo, ta cần tiền tokenize kho ngữ liệu này thành các từ. Vì ta đang sao chép một bản BPE tokenize (như GPT-2), ta vẫn có thể sử dụng gpt2 tokenize cho bước pre-tokenization:

```
from transformers import AutoTokenizer

tokenizer = AutoTokenizer.from_pretrained("gpt2")
```

Sau đó ta tính tần suất của từng từ trong kho ngữ liệu như khi làm với pre-tokenization:

```
from collections import defaultdict

word_freqs = defaultdict(int)

for text in corpus:
words_with_offsets=
tokenizer.backend_tokenizer.pre_tokenizer.pre_tokenize_str(text)
    new_words = [word for word, offset in words_with_offsets]
```

```
        for word in new_words:
            word_freqs[word] += 1


    print(word_freqs)
```

```
    defaultdict(int, {'This': 3, 'Ġis': 2, 'Ġthe': 1, 'ĠHugging': 1,
'ĠFace': 1, 'ĠCourse': 1, '.': 4, 'Ġchapter': 1,
     'Ġabout': 1, 'Ġtokenization': 1, 'Ġsection': 1, 'Ġshows': 1,
'Ġseveral': 1, 'Ġtokenizer': 1, 'Ġalgorithms': 1,
      'Hopefully': 1, ',': 1, 'Ġyou': 1, 'Ġwill': 1, 'Ġbe': 1,
'Ġable': 1, 'Ġto': 1, 'Ġunderstand': 1, 'Ġhow': 1,

    'Ġthey': 1, 'Ġare': 1, 'Ġtrained': 1, 'Ġand': 1, 'Ġgenerate': 1,
'Ġtokens': 1})
```

Tiếp theo chúng ta sẽ tính bộ từ vựng cơ sở từ các kí tự sử dụng trong kho ngữ liệu:

```
    alphabet = []

    for word in word_freqs.keys():
        for letter in word:
            if letter not in alphabet:
                alphabet.append(letter)
    alphabet.sort()


print(alphabet)
```

```
    [ ',', '.', 'C', 'F', 'H', 'T', 'a', 'b', 'c', 'd', 'e', 'f', 'g',
'h', 'i', 'k', 'l', 'm', 'n', 'o', 'p', 'r', 's',

  't', 'u', 'v', 'w', 'y', 'z', 'Ġ']
```

Ta cũng có thể thêm các token đặc biệt từ mô hình ở đầu của bộ từ vựng. Trong trường hợp của GPT-2, token đặc biệt duy nhất đó là "<|endoftext|>":

```
vocab = ["<|endoftext|>"] + alphabet.copy()
```

Ta giờ cần phải chia mỗi từ thành các kí tự riêng lẻ để có thể bắt đầu huấn luyện

```
splits = {word: [c for c in word] for word in word_freqs.keys()}
```

Giờ ta đã sẵn sàng để huấn luyện, hãy cùng viết một hàm tính tần suất mỗi cặp. Ta sẽ cần sử dụng nó ở bước huấn luyện:

```
    def compute_pair_freqs(splits):
        pair_freqs = defaultdict(int)
        for word, freq in word_freqs.items():
            split = splits[word]
            if len(split) == 1:
                continue
            for i in range(len(split) - 1):
                pair = (split[i], split[i + 1])
                pair_freqs[pair] += freq

    return pair_freqs
```

Hãy nhìn vào một phần từ điển sau khi tách:

```
    pair_freqs = compute_pair_freqs(splits)

    for i, key in enumerate(pair_freqs.keys()):
        print(f"{key}: {pair_freqs[key]}")
        if i >= 5:
```

```
        break

    ('T', 'h'): 3
    ('h', 'i'): 3
    ('i', 's'): 5
    ('Ġ', 'i'): 2
    ('Ġ', 't'): 7

('t', 'h'): 3
```

Giờ thì, tìm xem cặp xuất hiện nhiều nhất bằng một vòng lặp nhanh:

```
    best_pair = ""
    max_freq = None

    for pair, freq in pair_freqs.items():
        if max_freq is None or max_freq < freq:
            best_pair = pair
            max_freq = freq

print(best_pair, max_freq)

('Ġ', 't') 7
```

Vậy phép hợp nhất đầu tiên là ('Ġ', 't') -> 'Ġt', và ta thêm 'Ġt' vào bộ từ vựng:

```
    merges = {("Ġ", "t"): "Ġt"}

vocab.append("Ġt")
```

Để tiếp tục, ta cần áp dụng sự hợp nhất ở từ điển splits. Hãy cùng viết một hàm khác cho nó:

```
def merge_pair(a, b, splits):
    for word in word_freqs:
        split = splits[word]
        if len(split) == 1:
            continue

        i = 0
        while i < len(split) - 1:
            if split[i] == a and split[i + 1] == b:
                split = split[:i] + [a + b] + split[i + 2 :]
            else:
                i += 1
        splits[word] = split

    return splits
```

Giờ ta có thể nhìn xem kết quả của lần hợp nhất đầu tiên:

```
splits = merge_pair("Ġ", "t", splits)
print(splits["Ġtrained"])
```

```
['Ġt', 'r', 'a', 'i', 'n', 'e', 'd']
```

Giờ thì ta có tất cả những gì mình cần để lặp cho đến khi ta học tất các các hợp nhất mà ta muốn. Hãy cùng nhắm tới bộ từ vựng có kích cỡ là 50:

```
vocab_size = 50
```

```
while len(vocab) < vocab_size:
    pair_freqs = compute_pair_freqs(splits)
    best_pair = ""
    max_freq = None
    for pair, freq in pair_freqs.items():
        if max_freq is None or max_freq < freq:
            best_pair = pair
            max_freq = freq
    splits = merge_pair(*best_pair, splits)
    merges[best_pair] = best_pair[0] + best_pair[1]

  vocab.append(best_pair[0] + best_pair[1])
```

Kết quả là, chúng ta đã học 19 quy tắc hợp nhất (bộ từ điển gốc có kích cỡ là 31 tương ứng 30 kí tự trong bảng chữ cái cùng một token đặt biệt):

```
print(merges)
```

```
{('Ġ', 't'): 'Ġt', ('i', 's'): 'is', ('e', 'r'): 'er', ('Ġ', 'a'): 'Ġa', ('Ġt', 'o'): 'Ġto', ('e', 'n'): 'en',
 ('T', 'h'): 'Th', ('Th', 'is'): 'This', ('o', 'u'): 'ou', ('s', 'e'): 'se', ('Ġto', 'k'): 'Ġtok',
 ('Ġtok', 'en'): 'Ġtoken', ('n', 'd'): 'nd', ('Ġ', 'is'): 'Ġis', ('Ġt', 'h'): 'Ġth', ('Ġth', 'e'): 'Ġthe',

 ('i', 'n'): 'in', ('Ġa', 'b'): 'Ġab', ('Ġtoken', 'i'): 'Ġtokeni'}
```

Và bộ từ vựng cấu thành bởi token đặc biệt, các kí tự trong bảng chữ cái, và tất cả kết quả từ các quy tắc hợp nhất:

```
print(vocab)
```

```
['<|endoftext|>', ',', '.', 'C', 'F', 'H', 'T', 'a', 'b', 'c',
'd', 'e', 'f', 'g', 'h', 'i', 'k', 'l', 'm', 'n', 'o',
 'p', 'r', 's', 't', 'u', 'v', 'w', 'y', 'z', 'Ġ', 'Ġt', 'is',
'er', 'Ġa', 'Ġto', 'en', 'Th', 'This', 'ou', 'se',
 'Ġtok', 'Ġtoken', 'nd', 'Ġis', 'Ġth', 'Ġthe', 'in', 'Ġab', 'Ġtokeni']
```

Sử dụng train_new_from_iterator() trên cùng kho ngữ liệu sẽ không mang về kết quả kho ngữ liệu y hệt. Đó là bởi khi có sự lựa chọn về cặp có tần suất cao nhất, ta đã chọn cái đầu tiên xuất hiện, trong khi thư viện Tokenizers chọn cái đầu tiên dựa trên ID bên trong của nó.

Để tokenize văn bản mới, chúng ta tiền tokenize nó, tách ra, rồi áp dụng quy tắc hợp nhất được học:

```
def tokenize(text):
pre_tokenize_result= tokenizer._tokenizer.pre_tokenizer.pre_tokenize_str(text)
  pre_tokenized_text = [word for word, offset in pre_tokenize_result]
  splits = [[l for l in word] for word in pre_tokenized_text]
  for pair, merge in merges.items():
    for idx, split in enumerate(splits):
      i = 0
      while i < len(split) - 1:
        if split[i] == pair[0] and split[i + 1] == pair[1]:
```

```
                split = split[:i] + [merge] + split[i + 2 :]
            else:
                i += 1
        splits[idx] = split

    return sum(splits, [])
```

Ta có thể thử các này với bất kì đoạn văn nào khác được tạo thành từ các kí tự trong bảng chữ cái:

```
tokenize("This is not a token.")
```

```
['This', 'Ġis', 'Ġ', 'n', 'o', 't', 'Ġa', 'Ġtoken', '.']
```

1.3 So sánh 2 mô hình trên một bài toán cụ thể nào đó có sử dụng BPE và không sử dụng BPE

1.3.1 Giới thiệu về BPE

BPE hoạt động bằng cách chia các từ thành các subword units (các đơn vị con từ), điều này giúp giảm kích thước từ vựng và cho phép mô hình học các mẫu từ các phần nhỏ hơn của từ. Nó đặc biệt hữu ích cho các ngôn ngữ có nhiều biến thể từ như tiếng Việt.

1.3.2. So sánh các mô hình

1.3.2.1 Model Bert

Epoch	Training Loss	Validation Loss	Accuracy	F1	Precision	Recall
1	0.319900	0.294484	0.872800	0.872574	0.875938	0.872800
2	0.204100	0.318293	0.895000	0.894996	0.895164	0.895000

Bảng 1.3.2.1 Bảng kết quả model BERT

1.3.2.2 Model GPT-2

Epoch	Training Loss	Validation Loss	Accuracy	F1	Precision	Recall
1	0.319800	0.284376	0.880600	0.880476	0.882567	0.880600
2	0.203500	0.283235	0.892200	0.892166	0.892920	0.892200

Bảng 1.3.2.2 Bảng kết quả model GPT-2

1.3.2.3 So sánh kết quả:

Model	Training Loss	Validation Loss	Accuracy	F1	Precision	Recall
GPT-2	0.203500	0.283235	0.892200	0.892166	0.892920	0.892200
BERT	0.204100	0.318293	0.895000	0.894996	0.895164	0.895000

1.3.2.3 Nhận xét:

- BERT có độ chính xác và các chỉ số đánh giá khác nhỉnh hơn GPT-2 trên cùng một bộ dữ liệu.

- Mặc dù BPE giúp giảm kích thước từ vựng và cải thiện khả năng hiểu các từ hiếm, WordPiece của BERT có thể tối ưu hơn trong việc xử lý các ngữ cảnh phức tạp của văn bản.
- Kết luận:

  Việc sử dụng BPE hay không sử dụng BPE phụ thuộc vào từng bài toán cụ thể. Trong bài toán phân loại cảm xúc, BERT với WordPiece có vẻ hiệu quả hơn GPT-2 với BPE. Tuy nhiên, các mô hình khác nhau có thể có kết quả khác nhau tùy thuộc vào cách chúng được thiết kế và huấn luyện.

CHƯƠNG 2 – XÂY DỰNG MÔ HÌNH CHATBOT CHĂM SÓC KHÁCH HÀNG

Chatbot là một ứng dụng được xây dựng nhằm tương tác với con người một cách tự động bằng cách sử dụng các kỹ thuật ngôn ngữ tự nhiên. Chương trình này đóng vai trò như một trợ lý ảo, trò chuyện với con người và khiến họ nghĩ rằng họ đang nói chuyện với một người thật. Trong bài báo này, chúng tôi sẽ phát triển một hệ thống chatbot hỗ trợ khách hàng trong lĩnh vực shop quần áo, tự động trả lời ngay lập tức tất cả các câu hỏi từ người dùng bất cứ lúc nào, ngay cả ngoài giờ hành chính. Tính năng quan trọng của một ứng dụng chatbot là hiểu câu hỏi của người dùng và đưa ra câu trả lời thích hợp. Vì vậy, chúng tôi đề xuất phương pháp xây dựng ứng dụng chatbot phù hợp với nhu cầu của các shop quần áo. Chúng tôi áp dụng mô hình VietAI/vit5-base, một biến thể của T5, để dự đoán câu trả lời từ câu hỏi đầu vào. Mô hình này đã được tinh chỉnh (fine-tuned) từ mô hình tiền huấn luyện với dữ liệu tiếng Việt để phù hợp với mục tiêu của dự án. Thử nghiệm cho thấy mô hình VietAI/vit5-base là lựa chọn phù hợp cho ứng dụng chatbot của chúng tôi vì khả năng xử lý ngôn ngữ tự nhiên và hiệu suất cao trên các chỉ số đánh giá như ROUGE.

2.1 Giới thiệu

Mục tiêu của dự án này là xây dựng một trợ lý ảo để hỗ trợ khách hàng trong việc mua sắm quần áo trực tuyến. Trợ lý ảo sẽ giúp giải đáp các thắc mắc về sản phẩm, tình trạng đơn hàng và chính sách đổi trả, từ đó nâng cao trải nghiệm khách hàng và tăng hiệu quả kinh doanh.

Trợ lý ảo tập trung vào việc xử lý các câu hỏi thường gặp của khách hàng về quần áo, bao gồm:

- Thông tin sản phẩm: Chất liệu, màu sắc, kích thước, giá cả, và tính năng đặc biệt.
- Tình trạng đơn hàng: Xác nhận đơn hàng, theo dõi vận chuyển, và thông báo giao hàng.

- Chính sách đổi trả và bảo hành: Quy định về đổi trả, thời hạn bảo hành, và quy trình khiếu nại.
- Tư vấn thời gian: gợi ý các loại quần áo và cách phối đồ phù hợp với phong cách khách hàng yêu cầu.

2.2 Thu thập và xây dựng dữ liệu huấn luyện

Dữ liệu được thu thập từ nguồn:

Các cuộc hội thoại thực tế với khách hàng trên các nền tảng trực tuyến. Câu hỏi thường gặp (FAQs) từ các trang web của các shop quần áo. Đánh giá sản phẩm và phản hồi của khách hàng trên các trang thương mại điện tử. Tổng cộng, chúng tôi thu thập được 1185 dòng dữ liệu, bao gồm câu hỏi và câu trả lời bằng tiếng Anh. Sau đó dùng google translate API để chuyển thành tiếng việt.

Các bước xử lý dữ liệu bao gồm:

- Làm sạch dữ liệu: Loại bỏ các ký tự không cần thiết, ký tự đặc biệt và khoảng trắng thừa.
- Chuẩn hóa văn bản: Chuyển đổi tất cả văn bản thành chữ thường, loại bỏ dấu câu và chuẩn hóa các từ viết tắt.
- Gán nhãn dữ liệu: Gán nhãn cho các câu hỏi và câu trả lời để phù hợp với các tác vụ xử lý ngôn ngữ tự nhiên.

Dữ liệu được chia thành hai phần:

- Tập huấn luyện: 80% dữ liệu (948 dòng) dùng để huấn luyện mô hình.
- Tập kiểm tra: 20% dữ liệu (237 dòng) dùng để đánh giá hiệu suất của mô hình

STT	Bối cảnh	Câu hỏi	Câu trả lời
1	Bạn có thể hủy đơn hàng trong vòng 8 giờ kể từ khi đặt. Quy trình hủy đơn hàng: 1.Đăng nhập vào tài khoản của bạn trên trang web của chúng tôi. 2.Điều hướng đến phần "Lịch sử đặt hàng". 3. Xác định đơn hàng bạn muốn hủy. 4.Nhấp vào nút "Hủy đơn hàng". 5. Cung cấp lý do hủy (tùy chọn). 6.Xác nhận việc hủy bỏ. 7.Bạn sẽ nhận được email xác nhận sau khi quá trình hủy được xử lý.	Tôi phải hủy đơn hàng trong bao lâu?	Đơn đặt hàng có thể bị hủy trong vòng 8 giờ. Thực hiện theo các bước trên trang web của chúng tôi trong phần "Lịch sử đặt hàng" để hủy.
2	Đồ đi biển: - Phối màu: San hô, xanh nước biển, xanh ngọc, be cát, vàng chanh - Quần áo: Bikini, quần đùi, áo khoác ngoài, xà rông, mũ rơm	Bạn có bộ bikini cỡ lớn dễ thương nào không?	Tuyệt đối! Chúng tôi có những bộ bikini tuyệt vời cho mọi cơ thể. Kiểm tra trực tuyến bộ sưu tập ngoại cỡ của chúng tôi!
3	Bữa trưa ngoài trời: - Phối màu: Hồng phấn, xanh ô liu, denim nhạt, vàng pastel, oải hương - Quần áo: Áo cánh, váy, áo liền quần, áo liền quần, mũ mùa hè	Tôi có thể mang giày nổi bật với trang phục đơn giản không?	Chắc chắn! Những đôi giày nổi bật có thể tạo thêm yếu tố thú vị và bất ngờ cho bộ trang phục ăn sáng muộn của bạn.
4	Quần áo đi biển ngày dã ngoại: - Phối màu: Xanh da trời, be cát, san hô, xanh bọt biển, vàng	Bạn có lời khuyên nào về giày dép thân thiện với bãi biển không?	Dép xỏ ngón là một món đồ không thể thiếu trên bãi biển! Chúng dễ dàng mang vào và tháo ra và hoàn hảo

	chanh - Quần áo: Bikini, quần đùi, váy che thân, dép xỏ ngón, mũ rơm		cho những chuyến phiêu lưu trên cát.
5	Thời gian giao hàng sẽ được ước tính dựa trên địa chỉ của bạn ở phần "Thông tin" của sản phẩm bạn chọn.	Điều gì xảy ra nếu đơn hàng của tôi bị trì hoãn?	Sự chậm trễ hiếm khi có thể xảy ra. Nếu quan tâm, hãy liên hệ với nhóm hỗ trợ của chúng tôi để được hỗ trợ và cập nhật về trạng thái đơn hàng của bạn.
6	Chúng tôi chỉ chấp nhận thẻ tín dụng và thẻ ghi nợ cũng như các phương thức thanh toán trực tuyến an toàn như MOMO, VNPAY.	Sử dụng MOMO để thanh toán có an toàn không?	Tuyệt đối! MOMO là phương thức thanh toán trực tuyến an toàn mà chúng tôi tin tưởng để thực hiện các giao dịch an toàn.

Bảng 2.3 Dữ liệu

2.3 Mô hình huấn luyện

2.3.1 Kiến trúc Transformer

- Trong các bài toán seq2seq, các cấu trúc RNN và LSTM thường gặp những hạn chế nhất định, chẳng hạn như khó khăn trong việc xử lý các chuỗi dữ liệu dài do vấn đề vanishing gradient và hạn chế về khả năng tính toán song song. Mô hình transformer đã khắc phục được những nhược điểm này bằng cách sử dụng cơ chế self-attention, giúp cho quá trình huấn luyện nhanh hơn và kết quả đạt được cũng tốt hơn. Hiện nay, transformer đã được phát triển thành nhiều biến thể khác nhau không chỉ phục vụ cho các bài toán seq2seq mà còn cho các mô hình ngôn ngữ (language modeling).

2.3.1.1 Cơ chế Self Attention

- Self -attention chính là thành phần quan trọng nhất của transformer. Sự khác biệt là, trong khi cơ chế attention sẽ tính toán dựa trên trạng thái của decoder ở time-step hiện tại và tất cả các trạng thái ẩn của encoder. Còn self-attention có thể hiểu là attention trong một câu, khi từng thành phần trong câu sẽ tương tác với nhau. Từng token sẽ "quan sát" các tokens còn lại trong, thu thập ngữ cảnh của câu và cập nhập vector biểu diễn.
- Một chuỗi đầu vào $X = [x1, x2, \ldots, xn]$ với n n là số lượng từ trong chuỗi.
- Đầu tiên, mỗi từ trong chuỗi đầu vào được ánh xạ sang ba vector: Query (Q), Key (K), và Value (V) bằng cách nhân với các ma trận trọng số W_*Q , WK , và W_V* :

$$Q = X\,W\_Q\,,\; K = X\,W\_K\,,\; V = X\,W\_V$$

- Tính tích vô hướng (dot product) giữa vector Query của từ đang xét với tất cả các vector Key của các từ trong chuỗi để xác định mức độ tương đồng:

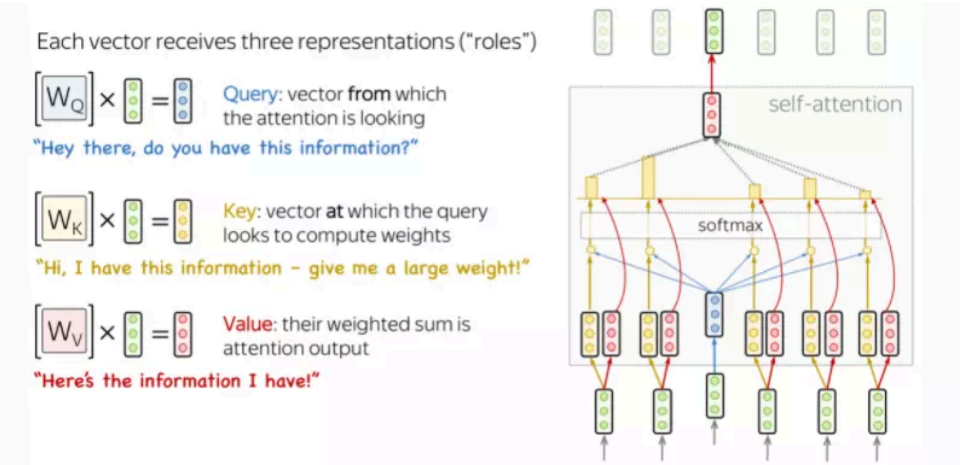
$$\textbf{Attention Score}(x\_i, x\_j) = Q_i \cdot K_j$$

- Chia các điểm tương tự này cho căn bậc hai của kích thước chiều của vector Key (d_k) để ổn định gradient
- Áp dụng hàm softmax lên các điểm số này để biến chúng thành các xác suất (trọng số attention):

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V$$

Hình 2.3.1.1

- Query: hỏi thông tin Key: trả lời rằng nó có một số thông tin
- Value: trả về thông tin đó Query được sử dụng khi một token "quan sát" những tokens còn lại, nó sẽ tìm kiếm thông tin xung quanh để hiểu được ngữ cảnh và mối quan hệ của nó với các tokens còn lại.
- Key sẽ phản hồi yêu cầu của Query và được sử dụng để tính trọng số attention.
- Cuối cùng, Value được sử dụng trọng số attention vừa rồi để tính ra vector đại diện (attention vector). Trong ảnh 3 ma trận *W Q*, W K và W V chính là các hệ số mà mô hình cần huấn luyện.



Hình 2.3.1.1

2.3.1.2 Cơ chế Multi-Head Attention

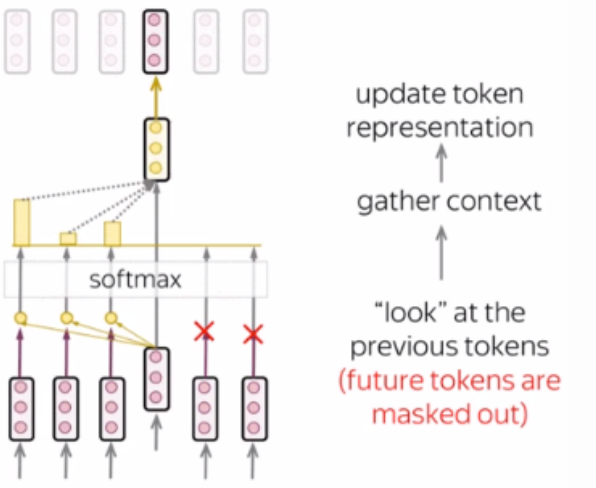
- Để tăng cường khả năng học các ngữ cảnh khác nhau, Self-Attention thường được thực hiện với nhiều "heads":
- Multiple Heads: Thực hiện nhiều Masked Self-Attention song song, mỗi head sử dụng các ma trận trọng số
- Concatenation: Kết hợp đầu ra của tất cả các heads và chiếu chúng qua một ma trận trọng số cuối cùng để tạo ra đầu ra cuối cùng của Multi-Head Attention:

Multi-Head(Q,K,V)=Concat(head1,head2,...,headh)W_O

- Cơ chế Self-Attention giúp mô hình Transformer có khả năng tập trung vào các từ có liên quan trong chuỗi đầu vào, giúp hiểu được ngữ cảnh và quan hệ giữa các từ một cách hiệu quả hơn. Điều này đặc biệt hữu ích trong các tác vụ như dịch máy, tóm tắt văn bản và nhiều ứng dụng khác trong NLP.

2.3.1.3 Cơ chế Masked Self Attention

- Đây là cơ chế được sử dụng cho decoder trong transformer, cụ thể nó thực hiện nhiệm vụ chỉ cho phép target token tại time-step hiện tại chỉ được phép dùng các tokens ở time-step trước đó. Về hoạt động nó cũng giống như đã giới thiệu ở trên, ngoại trừ việc nó không tính đến attention của những tokens trong tương lai.

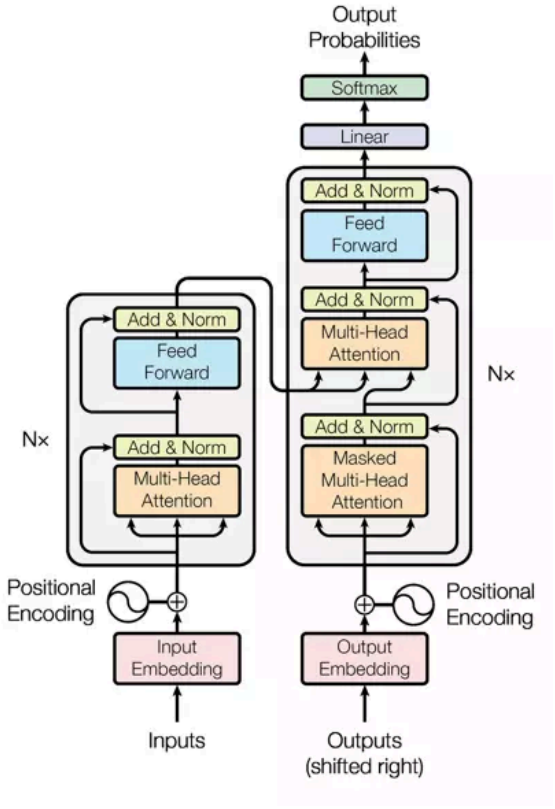


Hình 2.3.1.3

2.3.1.4 Transformer

- Cấu trúc của transformer được giới thiệu trong bài báo "Attention is all you need". Mô hình thực hiện chính xác những gì đã được giới thiệu ở trên. Ở bên trái là encoder, thông thường có Nx = 6 layers chồng lên nhau. Mỗi layer sẽ có multi-head

attention như đã tìm hiểu và khối feed-forward. Ngoài ra còn các các kết nối residual giống như trong mạng Resnet. Ở bên phải là decoder, tương tự cũng có Nx = 6 layers chồng lên nhau. Kiến trúc thì khá giống encoder những chỉ có thêm khối masked multi-head attention ở vị trí đầu tiên. Ta sẽ tìm hiểu sâu hơn các thành phần trong transformer.



Hình 2.3.1.4

- Positional encoding: Bởi vì transformer không có các mạng hồi tiếp hay mạng tích chập nên nó sẽ không biết được thứ tự của các token đầu vào. Vì vậy, cần phải có cách nào đó để cho mô hình biết được thông tin này. Đó chính là nhiệm vụ của positional encoding. Như vậy, sau bước nhúng từ (embedding layers) để thu được các tokens thì ta sẽ cộng nó với các vector thể hiện vị trí của từ trong câu.

- Lớp Normalization: Trong hình ảnh cấu trúc, có lớp "Add & Norm" thì từ Norm thể hiện cho lớp Normalization. Lớp này đơn giản là sẽ chuẩn hóa lại đầu ra của multi-head attention, mang lại hiệu quả cho việc nâng cao khả năng hội tụ.
- Kết nối Residual: Kết nối residual bản chất rất đơn giản: thêm đầu vào của một khối vào đầu ra của nó. Với kết nối này giúp mạng có thể chồng được nhiều layers. Như trên hình, kết nối residual sẽ được sử dụng sau các khối FFN và khối attention. Như trên hình từ "Add" trong "Add & Norm" sẽ thể hiện cho kết nối residual.
- Khối Feed-Forward: Đây là khối cơ bản, sau khi thực hiện tính toán ở khối attention ở mỗi lớp thì khối tiếp theo là FFN. Có thể hiểu là cơ chế attention giúp thu thập thông tin từ những tokens đầu vào thì FFN là khối xử lí những thông tin đó.

2.3.2 T5: The Unified Text-To-Text Transformer

2.3.2.1 Text-to-Text framework

- T5 chuyển đổi mọi bài toán xử lý văn bản sang dạng “text-to-text” (tức là lấy văn bản làm đầu vào và xuất văn bản làm đầu ra). Cấu trúc chung này, cũng được LLM khai thác cho phép chúng tôi lập mô hình và giải quyết nhiều nhiệm vụ khác nhau bằng cách tiếp cận chung. Chúng ta có thể áp dụng cùng một mô hình, mục tiêu, quy trình đào tạo và quy trình giải mã cho mọi nhiệm vụ mà chúng ta xem xét ! Chúng chỉ áp dụng phương pháp prompt và yêu cầu mô hình ngôn ngữ của tạo ra câu trả lời ở định dạng văn bản.

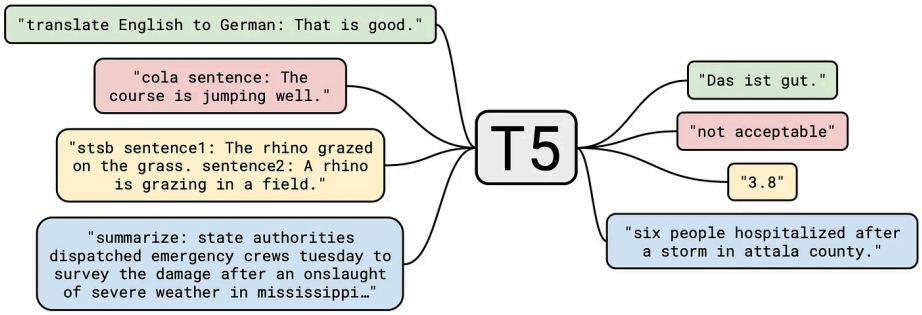


Figure 1: A diagram of our text-to-text framework. Every task we consider—including translation, question answering, and classification—is cast as feeding our model text as input and training it to generate some target text. This allows us to use the same model, loss function, hyperparameters, etc. across our diverse set of tasks. It also provides a standard testbed for the methods included in our empirical survey. "T5" refers to our model, which we dub the "**T**ext-**t**o-**T**ext **T**ransfer **T**ransformer".

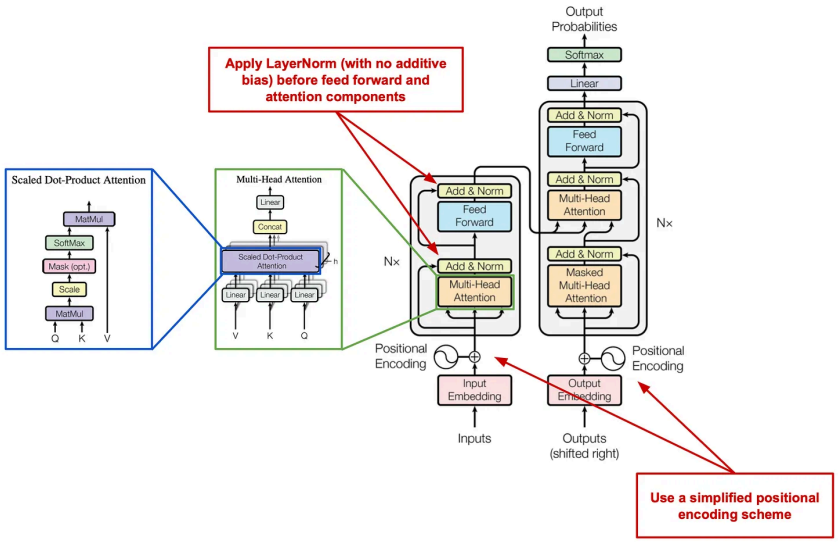
Hình 2.3.2.1

- Tất cả các tác vụ đang được T5 giải quyết có thể được chuyển đổi sang định dạng văn bản thành văn bản như sau:

1. Thêm tiền tố dành riêng cho nhiệm vụ vào chuỗi đầu vào ban đầu
2. Đưa trình tự này vào máy biến áp
3. Xây dựng mục tiêu của mô hình dưới dạng một chuỗi văn bản

- Sử dụng định dạng này, chúng ta có thể dễ dàng thực hiện các tác vụ như tóm tắt hoặc dịch thuật (tức là mục tiêu đương nhiên là một chuỗi). Ngoài ra, chúng ta có thể thực hiện phân loại bằng cách huấn luyện mô hình để tạo văn bản được liên kết với đúng lớp. Quá trình này hơi phức tạp đối với các vấn đề như hồi quy (tức là chúng ta phải làm tròn kết quả đầu ra có giá trị thực đến số thập phân gần nhất và coi nó như một vấn đề phân loại), nhưng nó có xu hướng hoạt động tốt đối với phần lớn các nhiệm vụ ngôn ngữ.

2.3.2.2 Mô hình



Hình 2.3.2.2

- Kiến trúc Transformer chứa cả bộ mã hóa và mô-đun giải mã. Công việc gần đây về mô hình hóa ngôn ngữ đã khám phá các biến thể kiến trúc chỉ dành cho bộ mã hóa hoặc bộ giải mã; ví dụ: BERT chỉ sử dụng bộ mã hóa, trong khi hầu hết các mô hình ngôn ngữ (lớn) chỉ sử dụng bộ giải mã. T5 sử dụng kiến trúc bộ mã hóa-giải mã gần giống với máy biến áp ban đầu. Sự khác biệt là:

1. LayerNorm được áp dụng ngay trước mỗi attention và feed forward
2. Không có additive bias nào được sử dụng cho LayerNorm
3. Một chiến lược nhúng vị trí đơn giản được sử dụng để thêm đại lượng vô hướng vào logit tương ứng được sử dụng để tính trọng số attention
4. Dropout được áp dụng trên toàn mạng

2.3.2.3 ViT5

- ViT5 tuân theo kiến trúc bộ mã hóa-giải mã đề xuất bởi Vaswani et al. (2017) và framework T5 được đề xuất bởi (Raffel và cộng sự, 2019).

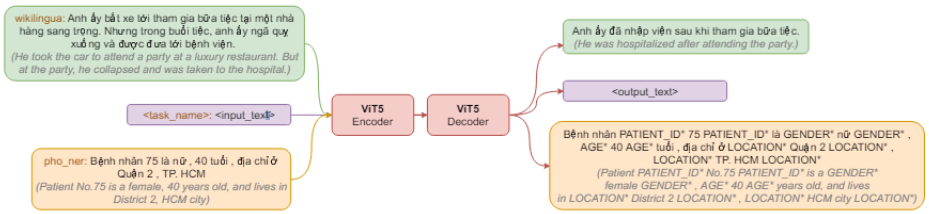


Figure 2: An overview of ViT5 encoder-decoder architecture, with input-output examples of two downstream tasks. For Named Entity Recognition, the decoder reconstructs the sentence with inserted Entity tags.

Hình 2.3.2.3

- Khác với một số mô hình ngôn ngữ dựa trên Transformer tiếng Việt hiện nay, tác giả nhận thấy rằng vốn từ vựng hiệu quả có thể góp phần cải thiện đáng kể hiệu suất mô hình. Do đó, tác giả đã xử lý trước tập hợp con 5GB của kho dữ liệu huấn luyện trước một cách cẩn thận như chuẩn hóa dấu câu và viết hoa, tách số. Sửa kích thước từ vựng thành 36K từ phụ và đào tạo mô hình SentencePiece (Kudo và Richardson, 2018) trên tập dữ liệu đó.

- Tác giả sử dụng Bộ dữ liệu CC100 (Bộ dữ liệu đơn ngữ từ dữ liệu thu thập thông tin trên web) (Wenzek và cộng sự, 2020; Conneau và cộng sự, 2020). Kho ngữ liệu chứa dữ liệu đơn ngữ của hơn 100 ngôn ngữ. Kho dữ liệu được xây dựng bằng cách sử dụng quy trình do (Wenzek và cộng sự, 2020) cung cấp thông qua quá trình xử lý ảnh chụp nhanh Commoncrawl tháng 1 tháng 12 năm 2018. Tổng kích thước của Corpus tiếng Việt là 138GB văn bản thô. Xử lý và lọc ra 69GB đoạn văn ngắn cho mô hình có độ dài 256 và 71GB đoạn văn dài cho mô hình có độ dài 1024

Table 1: Input and Output Length of Finetuned Datasets

	Wikilingua	Vietnews
Train	13707	99134
Test	3916	22498
#avg body length	521	519
#avg abstract length	44	38

Hình

- Báo cáo kết quả của mô hình ViT5 trên hai bộ dữ liệu: Wikilingua và Vietnews. Thực hiện thử nghiệm với hai phiên bản tiền huấn luyện ViT5: độ dài 256 và độ dài 1024 để hiểu rõ hơn tầm quan trọng của độ dài đoạn văn của dữ liệu tiền huấn luyện đối với việc tóm tắt bằng tiếng Việt. So sánh kết quả của mô hình ViT5base và ViT5large. Chúng tôi sử dụng ROUGE (Recall-Oriented Understudy for Gisting Evaluation) làm số liệu chuẩn cho cả hai bộ dữ liệu tóm tắt tài liệu đơn lẻ. Số liệu đo lường sự chồng chéo của n-gram và chuỗi từ giữa hai chuỗi ứng cử viên và chuỗi tham chiếu. ROUGE-1, ROUGE-2 và ROUGE-L lần lượt có nghĩa là sự chồng chéo giữa chuỗi unigram, bigram và chuỗi khớp dài nhất.

Table 2: Test result on Wikilingua and Vietnews Summarization

Models	WikiLingua			Vietnews		
	ROUGE-1	ROUGE-2	ROUGE-L	ROUGE-1	ROUGE-2	ROUGE-L
Transformer (RND2RND)	46.25	16.57	29.82	57.56	24.25	35.53
PhoBERT2PhoBERT	50.4	19.88	32.49	60.37	29.12	39.44
mBERT2mBERT	52.82	20.57	31.55	59.67	27.36	36.73
mBART	55.21	25.69	37.33	59.81	28.28	38.71
mT5	55.27	27.63	38.30	58.05	26.76	37.38
BARTpho	57.16	31.18	40.89	61.14	30.31	40.15
$ViT5_{base\ 256\text{-}length}$	57.86	29.98	40.23	61.85	31.70	41.70
$ViT5_{base\ 1024\text{-}length}$	58.61	31.46	41.45	62.77	33.16	42.75
$ViT5_{large\ 1024\text{-}length}$	**60.22**	**33.12**	**43.08**	**63.37**	**34.24**	**43.55**

Notes: The best scores are in bold and second best scores are underlined. The scores in gray color are our experiments. Code and models for reproducing our experiments: https://github.com/vietai/ViT5

Hình

2.3.3 GPT và GPT-2

- Các mô hình GPT được đào tạo trước trên một kho dữ liệu/tập dữ liệu văn bản không được gắn nhãn bằng cách sử dụng mục tiêu mô hình hóa ngôn ngữ. Nói một cách đơn giản, điều này có nghĩa là chúng tôi huấn luyện mô hình bằng cách *(i)* lấy mẫu một số văn bản từ tập dữ liệu và *(ii)* huấn luyện mô hình để dự đoán từ tiếp theo; xem hình minh họa ở trên. Quy trình đào tạo trước này là một hình thức học tập tự giám sát , vì từ "tiếp theo" chính xác có thể được xác định bằng cách chỉ cần nhìn vào từ tiếp theo trong tập dữ liệu.biểu thị bộ mã thông báo có kích thước N bao gồm tập dữ liệu đào tạo trước như sau.

$$\mathcal{U} = \{u_1, u_2, \ldots, u_N\}$$

- Với một mô hình học sâu có các tham số θ, mục tiêu mô hình hóa ngôn ngữ sẽ cố gắng tối đa hóa likelihood được hiển thị bên dưới.

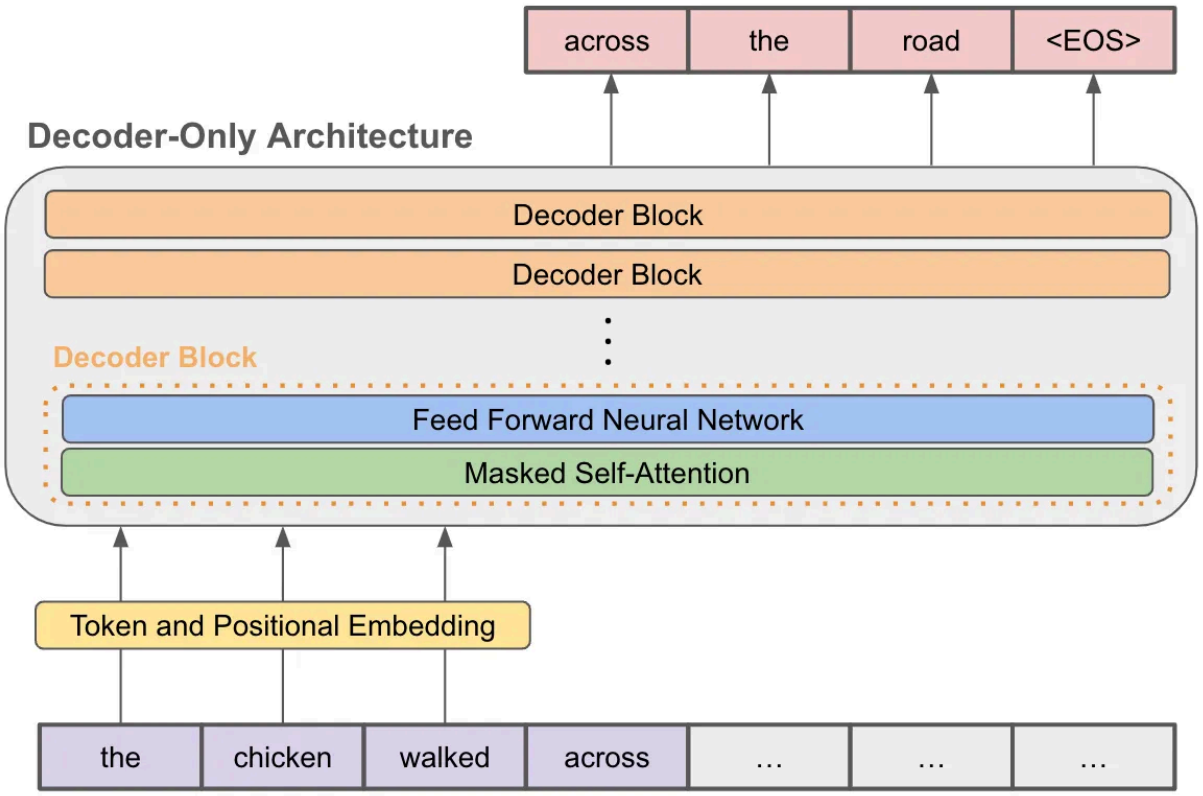
$$\mathcal{L}(\mathcal{U}) = \sum_{i=1}^{N} \log\left(\mathbb{P}(u_i | u_{i-k}, \ldots, u_{i-1}, \Theta)\right)$$

Language model loss over the full text corpus

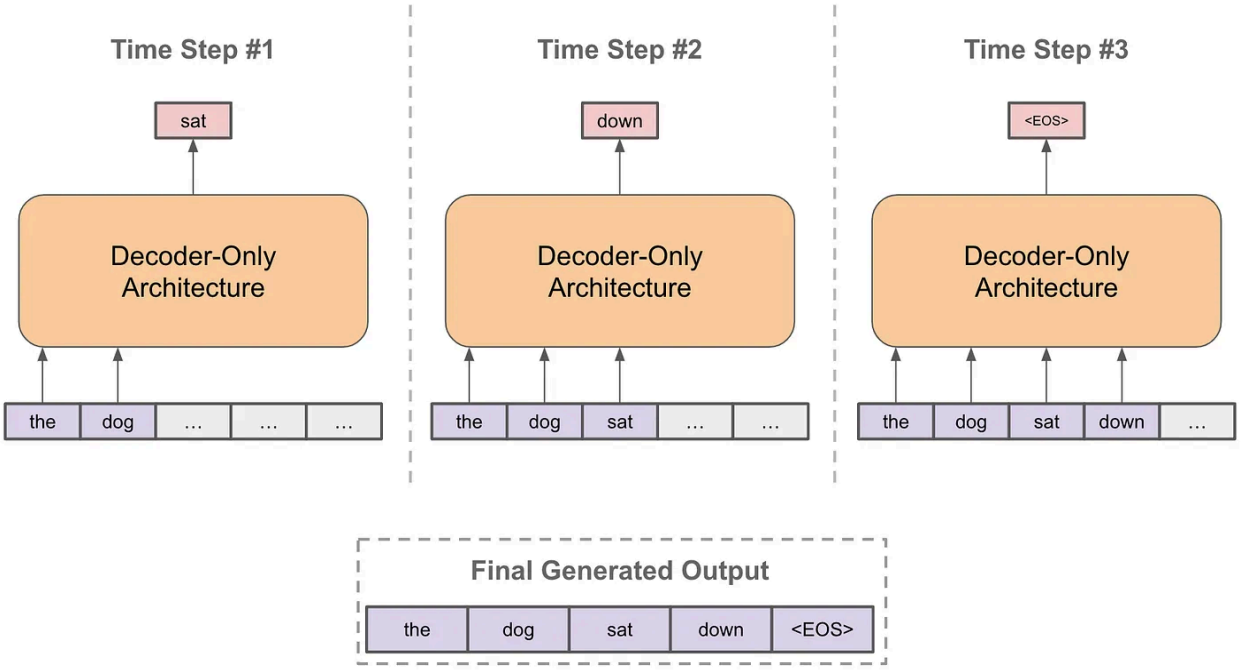
Conditional probability of i-th token given k preceding tokens and model parameters θ

- Các mô hình được đào tạo trước bằng cách sử dụng mục tiêu mô hình hóa ngôn ngữ tự giám sát như vậy thường được gọi là mô hình ngôn ngữ (LM). LM trở nên hiệu quả hơn khi chúng được mở rộng quy mô (nghĩa là có nhiều lớp, tham số, v.v.). Vì vậy, chúng ta sẽ thường thấy các phiên bản lớn hơn của các mô hình này (ví dụ: GPT-3 [7]), được gọi là mô hình ngôn ngữ lớn (LLM).

- Cả GPT và GPT-2 đều sử dụng kiến trúc máy biến áp chỉ dành cho bộ giải mã.
- Kiến trúc chỉ dành cho bộ giải mã sẽ loại bỏ các thành phần sau khỏi máy biến áp:
  - Toàn bộ mô-đun mã hóa
  - Tất cả các mô-đun tự chú ý của bộ mã hóa-giải mã trong bộ giải mã
- Sau khi các thành phần này được loại bỏ, mỗi lớp của bộ giải mã chỉ bao gồm một lớp masked self-attention, theo sau là feed forward neural network. Việc xếp chồng một số lớp như vậy lên nhau tạo thành một kiến trúc máy biến áp sâu, chỉ dành cho bộ giải mã, chẳng hạn như các kiến trúc được sử dụng cho GPT hoặc GPT-2.

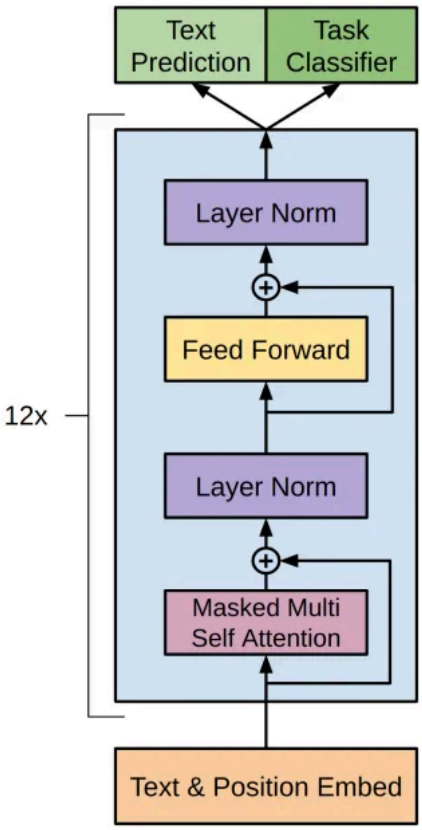


- Việc lựa chọn sử dụng kiến trúc bộ giải mã (ngược lại với bộ mã hóa) cho LM không phải là tùy ý. Các lớp masked self-attention trong bộ giải mã đảm bảo rằng mô hình không thể nhìn trước chuỗi từ khi tạo biểu diễn mã thông báo. Ngược lại, khả năng tự chú ý hai chiều (như được sử dụng trong bộ mã hóa) cho phép cách biểu diễn của mỗi mã thông báo được điều chỉnh dựa trên tất cả các mã thông báo khác trong một chuỗi.



Đầu ra tự hồi quy từ kiến trúc máy biến áp chỉ có bộ giải mã

- GPT là mô hình hiểu ngôn ngữ có mục đích chung được đào tạo theo hai giai đoạn: đào tạo trước và tinh chỉnh.



- GPT sử dụng kiến trúc máy biến áp 12 lớp, chỉ dành cho bộ giải mã ứng với bộ giải mã của Transfomer (ngoài việc sử dụng các phần nhúng vị trí có thể học được); xem hình trên. Trước tiên, GPT thực hiện đào tạo trước mô hình ngôn ngữ trên tập dữ liệu BooksCorpus, sau đó được tinh chỉnh riêng biệt (theo cách được giám sát) đối với nhiều nhiệm vụ hiểu ngôn ngữ.

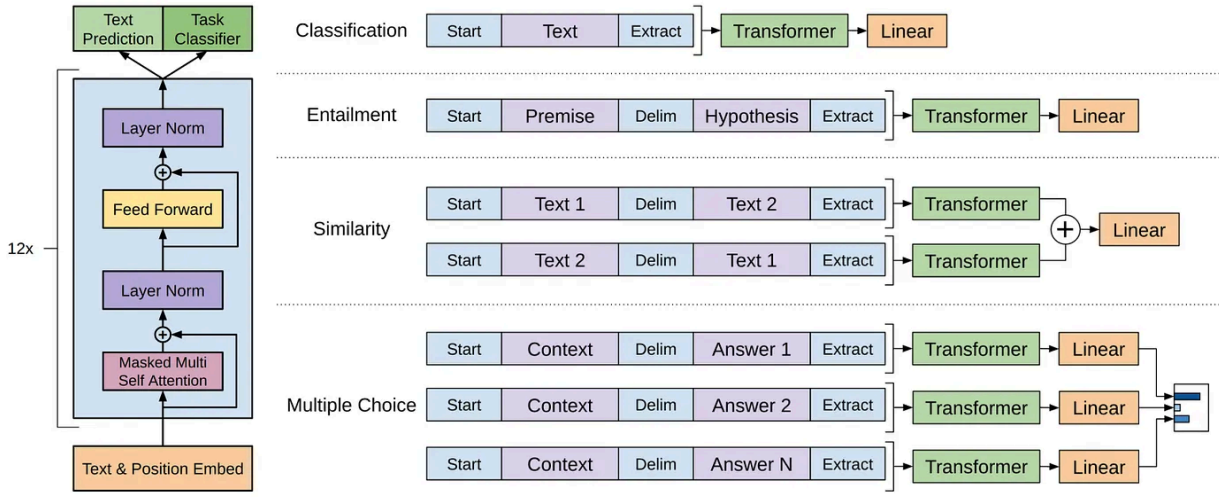


Figure 1: **(left)** Transformer architecture and training objectives used in this work. **(right)** Input transformations for fine-tuning on different tasks. We convert all structured inputs into token sequences to be processed by our pre-trained model, followed by a linear+softmax layer.

- Đề xuất của GPT-2 tuân theo mô hình tương tự như người tiền nhiệm của nó. Mô hình này được đào tạo trước bằng cách sử dụng mục tiêu mô hình hóa ngôn ngữ, nhưng nó không thực hiện tinh chỉnh, thay vào đó chọn giải quyết các tác vụ tiếp theo theo cách không cần thực hiện. Nói một cách đơn giản, GPT-2 thực hiện việc học đa tác vụ bằng cách:

1. Đào tạo trước LM chung trên dữ liệu văn bản thô
2. Sử dụng "lời nhắc" bằng văn bản để thực hiện suy luận không cần thực hiện đối với nhiều nhiệm vụ khác nhau

- Kiến trúc mô hình giống hệt với GPT, ngoại trừ một số khác biệt nhỏ (ví dụ: khởi tạo trọng số khác nhau, từ vựng lớn hơn, chuỗi đầu vào dài hơn, v.v.). Bất chấp kích thước của các LM này, chúng được phát hiện là underfit với tập dữ liệu WebText

trong quá trình đào tạo trước, cho thấy rằng các LM lớn hơn sẽ hoạt động tốt hơn nữa.

- Trong bài báo cáo này, chúng em sẽ thử nghiệm NlpHUST/gpt2-vietnamese là một mô hình ngôn ngữ dựa trên GPT-2, được phát triển bởi Viện Công nghệ thông tin và Truyền thông (HUST) tại Đại học Bách khoa Hà Nội. Đây là một trong những nỗ lực đầu tiên nhằm xây dựng một mô hình ngôn ngữ mạnh mẽ dành riêng cho tiếng Việt, nhằm hỗ trợ các ứng dụng ngôn ngữ tự nhiên (NLP) trong tiếng Việt.

2.4 Kết quả

- ROUGE (Recall-Oriented Understudy for Gisting Evaluation) là một bộ các độ đo dùng để đánh giá chất lượng tóm tắt tự động và các hệ thống sinh văn bản. Được đề xuất bởi Chin-Yew Lin năm 2004, ROUGE chủ yếu đo lường mức độ tương đồng giữa văn bản được sinh ra và văn bản tham chiếu (văn bản chuẩn).

- ROUGE bao gồm nhiều biến thể khác nhau, trong đó phổ biến nhất là:

- ROUGE-N: Đo lường sự tương đồng dựa trên n-gram. ROUGE-L: Dựa trên độ dài chuỗi con chung dài nhất (Longest Common Subsequence - LCS).
- ROUGE-W: Một phiên bản có trọng số của
- ROUGE-L. ROUGE-S: Dựa trên chuỗi con được bỏ qua (Skip-bigram).

Kết quả của model trên tập dữ liệu test clothesshop.vi

Models	ROUGE-1	ROUGE-2	ROUGE-L
VietAI/vit5-base	0.717	0.503	0.584

- Kết quả dự đoán của mô hình VietAI/vit5-base trên tập test so với câu trả lời thực tế

Context	Question	Predicted Answer	Actual Answer
quần áo đi biển ngày dã ngoại: - phối màu: xanh da trời, be cát, san hô, xanh bọt biển, vàng chanh - quần áo: bikini, quần đùi, váy che thân, dép xỏ ngón, mũ rơm	bạn có chiếc mũ đi biển nào có vành rộng không?	vâng, chúng tôi có! mũ rộng vành giúp chống nắng và phong cách trên cát.	vâng, chúng tôi có! mũ rơm rộng vành giúp bạn chống nắng tốt hơn đồng thời vẫn giữ cho bạn phong cách trên cát.
chúng tôi chỉ chấp nhận thẻ tín dụng và thẻ ghi nợ cũng như các phương thức thanh toán trực tuyến an toàn như momo, vnpay.	sử dụng momo để thanh toán có an toàn không?	chúng tôi chấp nhận thẻ tín dụng, thẻ ghi nợ, momo và vnpay.	tuyệt đối! momo là phương thức thanh toán trực tuyến an toàn mà chúng tôi tin tưởng để thực hiện các giao dịch an toàn.
nếu bạn muốn xem sản phẩm quần áo của chúng tôi theo từng loại quần áo, bạn có thể chọn một số tùy chọn tại menu danh mục để lọc loại quần áo như "áo gililе", "áo khoác", "áo len", "áo hoodie", "nam và nữ". mục áo phông", "quần/baggy", "bộ đồ ngủ", "quần short", "quần jeans", "giày", "áo sơ mi" hoặc "váy".	tôi có thể xem tất cả các sản phẩm cùng một lúc không?	tuyệt đối! xem phần "quần áo" để biết nhiều loại quần áo, bao gồm "áo sơ mi gile", "áo khoác", "áo len" và nhiều sản phẩm khác trong phần "quần áo" của danh mục.	để có cái nhìn toàn diện, hãy truy cập trang chính hoặc sử dụng menu thư mục để khám phá các loại quần áo cụ thể. tận hưởng mua sắm!

trang phục, quần áo sự kiện thể thao: - phối màu: vàng neon, đen, xanh dương, xanh chanh, san hô, xám than - quần áo: quần short thể thao, áo thấm ẩm, giày thể thao, mũ bóng chày, kính râm	cách tốt nhất để làm sạch kính râm bằng lớp phủ chống phản chiếu là gì?	sử dụng dung dịch nước và xà phòng nhẹ với vải sợi nhỏ mềm để nhẹ nhàng lau sạch bụi bẩn và vết ố trên kính râm.	sử dụng dung dịch nước và xà phòng nhẹ cùng với vải sợi nhỏ mềm để nhẹ nhàng lau sạch tròng kính và khung kính râm, tránh dùng chất tẩy rửa có tính ăn mòn và các vật liệu thô ráp có thể làm hỏng lớp phủ.
trang phục, quần áo dã ngoại mùa thu - phối màu: các màu thu hoạch như cam bí ngô, đỏ nam việt quất, xanh ô liu và vàng ngô - quần áo: áo len dệt kim ấm cúng, váy kẻ sọc, bốt cao đến mắt cá chân, tất đan, khăn quàng cổ	giày nào hợp với váy kẻ sọc?	giày cao đến mắt cá chân hoặc giày đế bệt là sự lựa chọn hoàn hảo để thêm nét phong cách mùa thu cho trang phục của bạn.	giày cao đến mắt cá chân hoặc giày đế bệt là sự lựa chọn hoàn hảo để thêm nét phong cách mùa thu cho bộ váy kẻ sọc của bạn.
quần áo đi biển ngày dã ngoại: - phối màu: xanh da trời, be cát, san hô, xanh bọt biển, vàng chanh - quần áo: bikini, quần đùi, váy che thân, dép xỏ ngón, mũ rơm	màu nào hot cho bãi biển năm nay?	màu be cát và san hô đang là xu hướng trên bãi biển năm nay. đừng quên kính râm của bạn!	xanh da trời, san hô và vàng chanh đều có! hoàn hảo cho những ngày đi biển đầy nắng.
có, bạn có thể xem giỏ hàng của mình bằng cách nhấp vào biểu tượng giỏ hàng nằm ở góc trên cùng	tôi có thể xem lại giỏ hàng của mình trên ứng dụng di động không?	tuyệt đối! trên ứng dụng di động, bạn có thể xem lại giỏ hàng của mình trên ứng dụng di động. nhấp	chắc chắn! nhấn vào biểu tượng giỏ hàng trên ứng dụng để xem, sửa đổi hoặc tiến hành thanh toán

bên phải của trang web. điều này cho phép bạn xem lại các mục đã chọn, thực hiện thay đổi và tiến hành thanh toán khi sẵn sàng.		vào biểu tượng giỏ hàng để tiếp tục mua sắm.	với các mặt hàng bạn đã chọn.
có, bạn có thể xem giỏ hàng của mình bằng cách nhấp vào biểu tượng giỏ hàng nằm ở góc trên cùng bên phải của trang web. điều này cho phép bạn xem lại các mục đã chọn, thực hiện thay đổi và tiến hành thanh toán khi sẵn sàng.	quá trình xem xét giỏ hàng của tôi là gì?	có, hãy nhấp vào biểu tượng giỏ hàng ở góc trên cùng bên phải của trang web để xem lại các mặt hàng đã chọn. sau đó bạn có thể xem lại các mặt hàng đã chọn.	nhấp vào biểu tượng giỏ hàng trên cùng bên phải để truy cập các mặt hàng bạn đã chọn. xem lại, thực hiện thay đổi và tiến hành thanh toán khi bạn đã sẵn sàng.
chúng tôi chỉ chấp nhận thẻ tín dụng và thẻ ghi nợ cũng như các phương thức thanh toán trực tuyến an toàn như momo, vnpay.	phương thức thanh tóan nào được chấp nhận?	chúng tôi chấp nhận thẻ tín dụng, thẻ ghi nợ và các phương thức trực tuyến an toàn như momo và vnpay.	chúng tôi chấp nhận thẻ tín dụng, thẻ ghi nợ và các phương thức trực tuyến an toàn như momo và vnpay.

Kết quả của model NlpHUST/gpt2-vietnamese:

Question	Answer
bạn có tất đan màu xanh ô liu không?	bạn có tất đan màu xanh ô liu không?ÁÁ

Bạn có thể hủy đơn hàng trong vòng 8 giờ kể từ khi đặt	Bạn có thể hủy đơn hàng trong vòng 8 giờ kể từ khi đặt hàng, hoặc sau thời gian trên, quý khách vui lòng liên hệ với chúng tôi theo thông tin dưới đây để biết thêm chi tiết: Trong trường hợp này, Quý khách sẽ không phải chịu bất cứ trách nhiệm nào đối với các giao dịch của mình. Chúng tôi chỉ chấp nhận thanh toán bằng thẻ tín dụng hoặc thẻ ghi nợ của các ngân hàng có uy tín và được cấp phép của Ngân hàng Nhà nước.
Tôi phải hủy đơn hàng trong bao lâu?	Tôi phải hủy đơn hàng trong bao lâu? Tôi có thể liên hệ trực tiếp với nhà cung cấp để được tư vấn cụ thể hơn không? Chào bạn, Cảm ơn bạn đã quan tâm đến sản phẩm và dịch vụ của công ty chúng tôi. Chúng tôi xin trả lời câu hỏi của bạn như sau: Hiện nay trên thị trường có rất nhiều đơn vị phân phối và bán các loại máy hút bụi công nghiệp. Tuy nhiên, để lựa chọn được một địa chỉ uy tín để mua

- Nhận xét: mô hình GPT-2 được fine-tuning với bộ Vietnamese Oscar dataset không tối ưu để dùng làm chatbot. Bộ dữ liệu Vietnamese Oscar chủ yếu là văn bản chung, không tập trung vào đối thoại hoặc trả lời câu hỏi. Để mô hình GPT-2 hoạt động hiệu quả như một chatbot, cần phải fine-tuning với bộ dữ liệu chuyên về đối thoại và các mẫu câu hỏi-đáp
- Vietnamese Oscar dataset chủ yếu chứa văn bản thông tin, không có mẫu đối thoại tự nhiên.
- Cách khắc phục:

● Thu thập dữ liệu đối thoại: Thu thập hoặc tạo ra bộ dữ liệu đối thoại tự nhiên bằng tiếng Việt, bao gồm các câu hỏi và câu trả lời cụ thể.

- Fine-tuning với dữ liệu đối thoại: Fine-tuning lại mô hình với dữ liệu đối thoại để nó học cách trả lời câu hỏi theo ngữ cảnh.
- Sử dụng mô hình hỗ trợ đối thoại: Sử dụng các mô hình GPT được thiết kế riêng cho đối thoại như DialoGPT hoặc ChatGPT.

TÀI LIỆU THAM KHẢO

Tiếng Việt

1. https://trituenhantao.io/kien-thuc/byte-pair-encoding-vu-khi-bi-mat-cua-nlp-hien-dai/
2. https://huggingface.co/learn/nlp-course/vi/chapter6/5#byte-pair-encoding-tokenization

PHỤ LỤC

Phần này bao gồm những nội dung cần thiết nhằm minh họa hoặc hỗ trợ cho nội dung luận văn như số liệu, biểu mẫu, tranh ảnh. . . . nếu sử dụng những câu trả lời cho một *bảng câu hỏi thì bảng câu hỏi mẫu này phải được đưa vào phần Phụ lục ở dạng nguyên bản* đã dùng để điều tra, thăm dò ý kiến; **không được tóm tắt hoặc sửa đổi**. Các tính toán mẫu trình bày tóm tắt trong các biểu mẫu cũng cần nêu trong Phụ lục của luận văn. Phụ lục không được dày hơn phần chính của luận văn.