ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
KHOA HỆ THỐNG THÔNG TIN

ĐỒ ÁN CUỐI KÌ

GÁN NHÃN TỪ LOẠI TIẾNG VIỆT

MÔN XỬ LÝ NGÔN NGỮ TỰ NHIÊN

Lớp: CS211.M11

GVHD: Nguyễn Trọng Chỉnh

Nhóm sinh viên thực hiện

STT	Tên	MSSV
1	Nguyễn Như Long	19521790
2	Hồ Anh Dũng	18520630

□□ TP. Hồ Chí Minh, 12/2021 □□

LỜI CẢM ƠN

Lời đầu tiên, nhóm xin gửi lời cảm ơn chân thành đến tập thể quý Thầy Cô Trường Đại học Công nghệ thông tin – Đại học Quốc gia TP.HCM và quý Thầy Cô khoa Khoa học máy tính đã giúp cho nhóm có những kiến thức cơ bản làm nền tảng để thực hiện đồ án này.

Đặc biệt, nhóm xin gửi lời cảm ơn đến Thầy Nguyễn Trọng Chỉnh (Giảng viên hướng dẫn môn Xử lý ngôn ngữ tự nhiên). Thầy đã cung cấp kiến thức, chỉ bảo trực tiếp, hướng dẫn tận tình, sửa chữa và đóng góp nhiều ý kiến quý báu giúp nhóm hoàn thành tốt đồ án của mình.

Xuất phát từ mục đích học tập môn Xử lý ngôn ngữ tự nhiên, cũng như tìm hiểu thêm về việc Gán nhãn từ loại, nhóm chúng em đã thực hiện đồ án tìm hiểu "**Gán nhãn từ loại Tiếng Việt**". Trong quá trình thực hiện đồ án, nhóm chúng em đã vận dụng những kiến thức nền tảng đã tích lũy dựa trên những kiến thức được thầy cung cấp, đồng thời kết hợp với việc học hỏi và nghiên cứu những kiến thức mới từ thầy cô, bạn bè cũng như nhiều nguồn tài liệu tham khảo, nhóm đã cố gắng thực hiện đồ án một cách tốt nhất. Tuy nhiên, vì kiến thức chuyên môn còn hạn chế và bản thân còn thiếu nhiều kinh nghiệm thực tiễn, nên nội dung của báo cáo không tránh khỏi những thiếu sót nhưng nó là kết quả của sự nỗ lực của các thành viên trong nhóm và sự giúp đỡ của Thầy.

Một lần nữa xin gửi đến Thầy lời cảm ơn chân thành và tốt đẹp nhất!

MỤC LỤC

Chương 1: GIỚI THIỆU BÀI TOÁN

1.1 Giới thiệu bài toán

Part of speech (POS) tagging là một trong những phương pháp quan trọng của xử lý ngôn ngữ tự nhiên, cũng như trong việc hiểu nội dung câu hoặc văn bản. POS là thuật ngữ truyền thống để chỉ các loại từ được phân biệt về mặt ngữ pháp trong một ngôn ngữ. Trong quá trình phát triển chúng ta quen với việc xác định từ loại trong văn bản. Đọc một câu chúng ta có thể xác định rõ tử loại như là danh từ, động từ hoặc tính từ… Để xác định từ rõ từ loại trong câu thường phức tạp hơn nhiều trong việc ánh xạ các từ qua từ điển. Đó là bởi vì một từ có thể được gán rất nhiều từ loại dựa vào ngữ cảnh của văn bản. Đây gọi là sự nhập nhằng. Thật khó để ta xác định một từ đó thuộc từ loại nào dựa vào một ngữ liệu nhất định vì tất cả ngữ cảnh mới và từ mới mỗi ngày liên tục xuất hiện đó cũng là vấn đề cho việc gán từ loại thủ công. Phân biệt các bộ phận của từ trong câu sẽ giúp ta hiểu rõ hơn về ý nghĩa của câu. Điều này cực kỳ quan trọng trong các truy vấn tìm kiếm. Việc xác định danh từ riêng, tổ chức, ký hiệu hoặc bất kỳ thứ gì tương tự sẽ cải thiện đáng kể mọi thứ, từ nhận dạng giọng nói đến tìm kiếm.

Nhãn (Tag): Là một ký hiệu được gán cho mỗi từ trong ngữ liệu, nhãn có thể thể hiện từ loại, chức năng, thì của từ trong câu đó.

Bộ nhãn (Tagset): Là một tập hợp các nhãn được quy ước cho một ngôn ngữ nhất định. Các bộ nhãn có thể rất khác nhau hoặc rất giống nhau tùy thuộc vào mức tương đồng giữa các ngôn ngữ

Trong đề tài này nhóm sẽ sử dụng các phương pháp tách từ là Maximum Matching và WFST. Gán nhãn từ loại với mô hình Hidden Markov kết hợp thuật toán Viterbi.

1.2 Ý tưởng thực hiện

- Bước 1 : Đầu vào là các câu văn thu thập được
- Bước 2 : Tiền xử lý các câu đầu vào
- Bước 3 : Thực hiện tách từ cho các câu đã tiền xử lý
- Bước 4 : Gán nhãn từ loại cho kết quả tách từ
- Bước 5 : Đầu ra là văn bản đã được gán nhãn từ loại

Chương 2: THU THẬP DỮ LIỆU

2.1 Nguồn thu thập

Gồm các câu bất kỳ thuộc nhiều chủ đề được thu thập từ các trang báo điện tử cũng như mạng xã hội như https://vnexpress.net/, https://Facebook.com.vn/ , và các câu mà nhóm tự nghĩ ra trong đó có các câu mang tính nhọc nhằn để kiểm tra cách xử lí của thuật toán.

2.2 Thông tin dữ liệu

Bộ dữ liệu thu thập được gồm :

- Số lượng câu : 100 câu
- Số từ nhiều nhất trong một câu : 30
- Số từ ít nhất trong một câu : 4
- Mỗi dòng là 1 câu
- Các từ được phân cách với nhau bằng dấu cách.

```
Việt Nam đang trên đà phát triển, hoà cùng với xu thế hội nhập toàn cầu trên thế giới đồng thời là đất nước giàu truyền thống nhân ái, đoàn kết
Chuyện học hành đối với học sinh là vô cùng quan trọng
Trong giờ sinh học, bạn Lan đã tìm cách trốn học
Rừng cây xanh biếc
Thành phố Hồ Chí Minh không những đẹp mà còn rất thơ mộng
Số tiền 100.000 khiến phú ông nổi lòng dạ đen tối
Riêng tư cách của anh đã không đủ để vào công ty này rồi!
Tôi đã sử dụng ứng dụng Yahoo! 2 năm này rồi
Vũ trụ bao gồm tất cả các vật chất, năng lượng và không gian hiện có, được coi là một khối bao quát
Chị Võ Thị Sáu sinh năm 1933 ở huyện Đất Đỏ, tỉnh Bà Rịa (nay là tỉnh Bà Rịa – Vũng Tàu)
Bàn tay rắn chắc của anh ấy chắn phía trước ngực của tôi
Con rắn chắc chắn sẽ tiến về đây, đừng hoảng sợ mà lùi bước
Sự thực hiện việc này khiến tôi trở nên kỉ luật, mạnh mẽ hơn
Mẹ tôi bảo rằng: "Học phải đi đôi với hành"
Vì vậy, cần thận trọng trong sử dụng ngôn ngữ cơ thể
GS.Phạm có đức tính nho nhã điềm đạm
Người dân địa phương cũng như khách du lịch đua nhau chụp những bức ảnh tuyệt đẹp tại Đảo Willis Island ở Australia
```

Chương 3: TẠO NGỮ LIỆU

3.1 Tạo ngữ liệu cho Tách từ

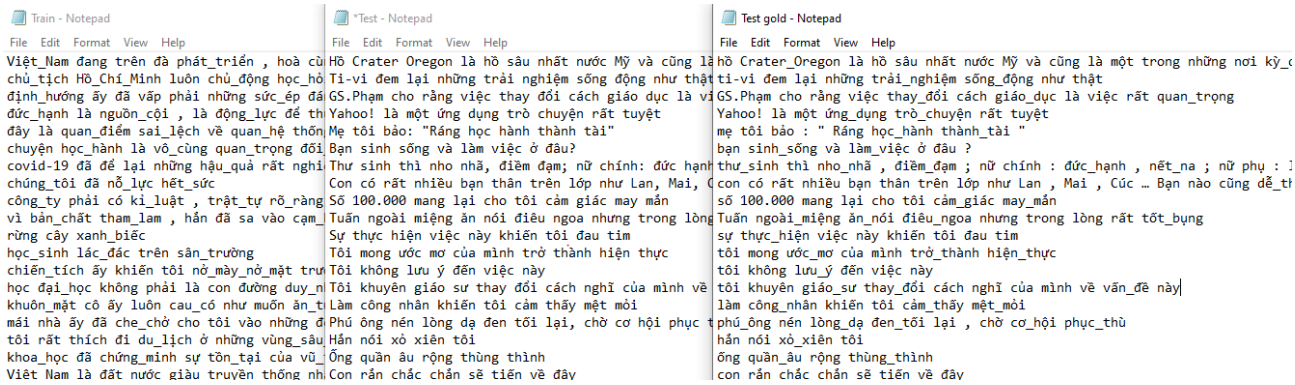
Phương pháp tách từ: Nhóm thực hiện tách từ thủ công bằng cách sử dụng từ điển tiếng Việt của Hồ Ngọc Đức để kiểm tra các từ tiếng việt trong câu có hay là không

Link từ điển : https://www.informatik.uni-leipzig.de/~duc/Dict/

Data: Từ bộ ngữ liệu thu thập ban đầu 100 câu, nhóm sử dụng 90 câu để tách từ

Nhóm chia data thành 3 tập: Train, Test, Test_gold:

- Tập Train : gồm 60 câu đã được tách từ thủ công dùng cho phương pháp WFST
- Tập Test : gồm 30 câu chưa được tách từ
- Tập Test_gold : gồm 30 câu ở tập Test đã được tách từ chuẩn



Hình mô tả bộ ngữ liệu cho phần tách từ

3.2 Tạo ngữ liệu cho Gán nhãn

Phương pháp gán nhãn thủ công: nhóm tiến hành gán nhãn thủ công sử dụng từ điển tiếng việt Hồ Ngọc Đức và bộ nhãn từ loại của Vncore NLP

Bảng danh sách nhãn từ loại

(Nguồn: https://github.com/vncorenlp/VnCoreNLP/blob/master/VLSP2013_POS_tagset.pdf)

STT	Nhãn	Tên	Ví dụ
1	N	Danh từ	tiếng, nước, thủ đô, nhân dân, đồ đạc
2	Np	Danh từ riêng	Hồ Chí Minh, Bình Dương, Võ Thị Sáu, Bà Rịa
3	Nc	Danh từ chỉ loại	con, cái, đứa, bức
4	Nu	Danh từ đơn vị	mét, cân, giờ, nắm, nhúm, hào, xu
5	Ni	Danh từ ký hiệu	A1, A4, 60A, 60B, 20a, 20b, ABC
6	V	Động từ	ngủ, ngồi, cười; đọc, viết, đá, đặt, thích
7	A	Tính từ	tốt, xấu, đẹp; cao, thấp, rộng
8	P	Đại từ	tôi, chúng tôi, hắn, nó, y, đại nhân, đại
9	L	Định từ	mỗi, từng, mọi, cái, các, những, mấy
10	M	Số từ	một, mười, mười ba, dăm, vài, mươi
11	R	Phó từ	đã, sẽ, đang, vừa, mới, từng, xong, rồi
12	E	Giới từ	trên, dưới, trong, ngoài; của, trừ, ngoài
13	C	Liên từ	vì vậy, tuy nhiên, ngược lại
14	Cc	Liên từ đẳng lập	và, hoặc, với, cùng
15	I	Thán từ	ôi, chao, a ha
16	T	Trợ từ	à, a, á, ạ, ấy, chắc, chăng, cho, chứ
17	B	Từ vay mượn	Internet, email, video, chat
18	Y	Từ viết tắt	OPEC, WTO, HIV
19	X	Các từ không thể phân loại	
20	Z	Yếu tố cấu tạo từ	bất, vô, phi

21	CH	Nhãn dành cho các loại dấu	. ! ? , ; :

- Data: Từ bộ ngữ liệu thu thập ban đầu 100 câu, nhóm sử dụng 70 câu để gán nhãn

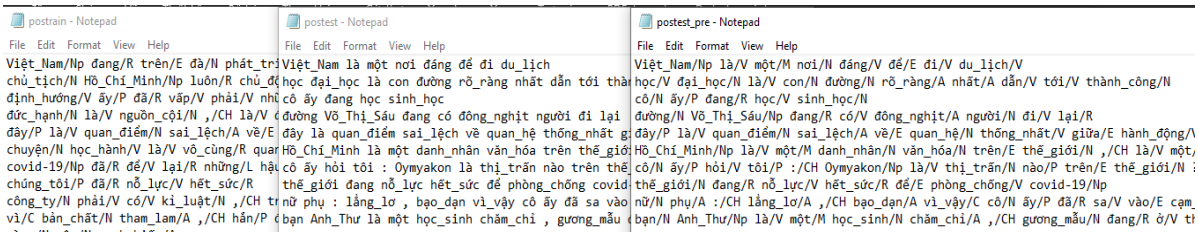
- Nhóm tiến hành tách tay thủ công cho 70 câu (trong đó có một số câu đã tách tay ở bước tách từ trước rồi, nhóm sử dụng lại), sau đó tiến hành gán nhãn cho từng câu

- Nhóm chia dữ liệu đã gán nhãn thành 3 tập: PosTrain, PosTest, PosTest_gold:

• Tập PosTrain : gồm 50 câu đã được tách từ và gán nhãn chuẩn

• Tập PosTest : gồm 20 câu đã tách từ chuẩn nhưng chưa gán nhãn

• Tập PosTest_gold: gồm 20 câu ở tập Test đã được gán nhãn chuẩn



Chương 4: TÁCH TỪ

4.1 Thuật toán Maximum Matching

4.1.1 Cơ sở lý thuyết

Longest Matching (So khớp cực đại) là thuật toán tách từ dựa trên ý tưởng xét các tiếng từ trái sang phải, hoặc từ phải sang trái trong một câu, các tiếng đầu tiên dài nhất có thể mà xuất hiện trong từ điển sẽ được tách ra làm một từ.

- Mô tả Thuật toán:
  - Input: một câu bất kì có số tiếng là k, từ điển D với độ dài từ dài nhất là m

    Bước 1: Tách câu thành các tiếng riêng biệt, Khởi tạo i=1, n=m

    Bước 2: Nếu i > k kết thúc. Nếu i = k => tách tiếng cuối cùng thành 1 từ, kết thúc. Nếu i+n - 1 > length, n= k – i + 1

    Bước 3: Xét các tiếng tính từ vị trí thứ i đến i+ n - 1 . Nếu các tiếng này không nằm trong từ điển thì nhảy tới bước 5

    Bước 4: Tách câu từ vị trí i đến i + n -1 thành 1 từ, i=i + n , n=m, nhảy tới bước 2

    Bước 5: Nếu n ≠ 1: n=n-1 . Nếu n=1, tách tiếng i thành 1 từ, i = i+1, n=m.Nhảy tới bước 2.

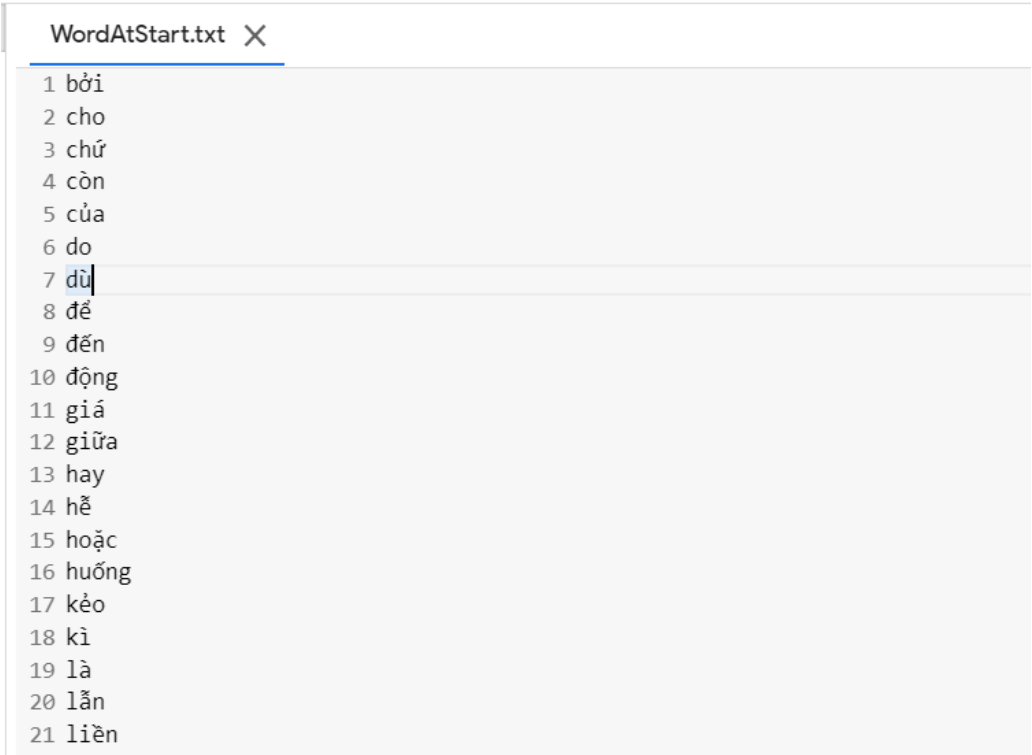
  - Output: Câu đã được tách thành các từ riêng biệt

4.1.2 Triển khai thực hiện

- **Data**: Từ ngữ liệu đã được chuẩn bị ở trên, nhóm sử dụng tập Test gồm 30 câu chưa được tách từ và tập Test_gold gồm 30 câu trong tập Test đã được tách chuẩn dùng để kiểm tra
- **Từ điển**: nhóm sử dụng từ điển tiếng việt của Underthesea với số tiếng dài nhất trong 1 từ trong từ điển là 4.

  Nguồn : 'https://github.com/undertheseanlp/dictionary/'

- Nhóm tiến hành thu thập 89 từ thường xuyên xuất hiện ở đầu câu để xử lí các trường hợp viết in hoa nằm đầu câu.



- Nhóm tiến hành liệt kê ra các dấu câu trong câu để dễ dàng tách câu sau này:

```
{'!', '"', '(', ')', ',', '...', ':', ';', '?', '-'}
```

- Sau đó nhóm viết thuật toán tách câu thành các tiếng riêng biệt, các dấu câu cũng được xem là 1 tiếng riêng biệt.
- Tiếp đến nhóm thiết kế phương pháp tách từ bằng Maximum Matching dựa trên mã giả ở trên

Ví dụ:

  - Câu ban đầu: Việt Nam đang trên đà phát triển, hoà cùng với xu thế hội nhập toàn cầu trên thế giới đồng thời là đất nước giàu truyền thống nhân ái, đoàn kết
  - Câu sau khi tách từ: ['Việt_Nam', 'đang', 'trên', 'đà', 'phát_triển', ',' , 'hoà', 'cùng', 'với', 'xu_thế', 'hội_nhập', 'toàn_cầu', 'trên', 'thế_giới', 'đồng_thời', 'là', 'đất_nước', 'giàu', 'truyền_thống', 'nhân_ái', ',' , 'đoàn_kết']

- Ở đây có thể thấy với MXM ta tách được các từ ghép Việt_Nam, phát_triển, xu_thế, hội_nhập, toàn_cầu, thế_giới , đồng_thời, đất_nước, truyền_thống, nhân_ái, đoàn_kết

- Đối với trường hợp viết hoa:
  - Trường hợp từ có 2 tiếng trở lên đều viết hoa và không nằm đầu dòng, xét nó là một từ và tách ra
  - Trường hợp từ có 2 tiếng trở lên đều viết hoa và nằm đầu dòng, kiểm tra xem từ đầu tiên có nằm trong bộ từ thường xuất hiện đầu câu hay không. Nếu có thì không xét là một từ. Nếu không thì xét là một từ và tách ra.

Ví dụ: Chị Võ Thị Sáu sinh năm 1933 ở huyện Đất Đỏ

+ Sau khi tách từ: ['Chị', 'Võ_Thị_Sáu', 'sinh', 'năm', '1933', 'ở', 'huyện', 'Đất_Đỏ']

+ Vì từ chị xuất hiện trong bộ từ đầu câu nên ở đây không xét từ chị Võ Thị Sáu là một từ,

+ Với từ Võ Thị Sáu gồm tất cả các tiếng đều viết hoa và không nằm đầu câu nên xét là một từ

- Cuối cùng nhóm lấy Data từ bộ Test để chạy thử và lưu két quả để sau này so sánh

4.2 Thuật toán WFST (Weighted Finite State Transducer)

4.2.1 Cơ sở lý thuyết

Mô hình chuyển dịch trạng thái hữu hạn có trọng số WFST là mô hình sử dụng trọng số xác suất xuất hiện của mỗi từ trong ngữ liệu. Dùng WFST duyệt qua câu cần xét. Cách tách từ nào có tổng xác suất các từ ghép lại lớn nhất là cách chọn tối ưu.

Mô tả thuật toán:

- **Bước 1**: Xây dựng từ điển trọng số, với nhãn i tương ứng với một từ trong từ điển, số *count* tương ứng với số lần xuất hiện của từ.

- **Bước 2**: Xây dựng từ điển chứa xác suất tương ứng của các từ: xác suất của một từ bằng số lần xuất hiện của từ chia cho tổng số từ trong từ điển
- **Bước 3**: Liệt kê tất cả trường hợp có thể tách từ của một câu, trong mỗi cách tách từ, các từ đều có nghĩa.
- **Bước 4**: Tính toán, lựa chọn cách tách từ có xác suất tốt nhất, kết thúc

4.2.2 Triển khai thực hiện

- Data: Từ ngữ liệu đã được chuẩn bị ở trên, nhóm sử dụng tập Train gồm 60 câu đã được tách từ chuẩn, tập Test gồm 30 câu chưa được tách từ và tập Test_gold gồm 30 câu trong tập Test đã được tách chuẩn dùng để kiểm tra
- Từ điển: nhóm sử dụng từ điển tiếng việt của Underthesea
- Nhóm gán trọng số của các từ trong từ điển là 1, với nhãn i là một từ trong từ điển, trọng số count= 1 là số lần xuất hiện của từ tương ứng trong từ điển
- Nhóm sử dụng bộ train gồm 60 câu, duyệt qua từng câu, kiểm tra xem mỗi từ có xuất hiện trong từ điển hay không. Nếu không xuất hiện trong từ điển thì thêm vào từ điển gán trọng số count= 1. Nếu có xuất hiện trong từ điển thì tăng trọng số count = count+1

  Ví dụ sau khi cập nhật trọng số:

```
., 'chúng_tôi': 3, 'chủng_viện': 1, 'chuốc': 1, 'chuộc': 1, 'chuồi': 1,
```

- Tiếp đến nhóm gán lại giá trị xác suất của các từ trong từ điển bằng cách lấy số lần xuất hiện của chúng chia cho tổng số từ trong từ điển đồng thời lấy -log2 để xác suất sau này trở thành phép cộng để xử lí:

  Prob_word= $-\log_2(count(word)/\text{len(dictionary)})$

- Tiếp đến nhóm tìm tất cả các trường hợp có thể tách của một từ bằng phương pháp Backtracking (quay lui), trong đó với mỗi các tách từ, các từ đều có nghĩa:

  Ví dụ: test_case="Riêng tư cách của anh đã không đủ rồi !"

  - Các Trường hợp có thể tách:

- Riêng tư cách của anh đã không đủ rồi !
- Riêng_tư cách của anh đã không đủ rồi !
- Riêng tư_cách của anh đã không đủ rồi !

- Vì xác suất của các từ là -log2 nên ta sẽ tìm xác suất của câu có giá trị nhỏ nhất là kết quả:

  Ví dụ: Trong từ điển xác suất của các từ lần lượt là:

  riêng= 13.9 ; tư= 14.9; cách= 13.3; riêng_tư= 14.9; tư_cách= 13.9

  + Trong ví dụ này ta có thể bỏ qua "của anh đã không đủ rồi" vì cách tách như nhau nên xác suất như nhau

  + Sau khi tính toán tổng xác suất của 3 cách tách trên , ta có xác suất của các cách tách lần lượt là:

  - Riêng tư cách : 42.1
  - Riêng_tư cách : 28.2
  - Riêng tư_cách : 27.8

  ⇨ Riêng tư_cách của anh đã không đủ rồi ! : là phương án chính xác

- Đối với các trường hợp in hoa, nhóm xét tương tự như ở phương pháp Maximum Matching
- Tiếp theo nhóm lấy 30 câu ở bộ test đến tiến hành chạy thử và lưu lại kết quả để sau này so sánh

4.3 Thư viện VnCoreNLP

VnCoreNLP là một package NLP trong Tiếng Việt, hỗ trợ Tokenize và các tác vụ NLP khác. Trong đề tài này nhóm sử dụng VnCoreNLP để thực hiện tách từ, sau đấy so sánh kết quả với 2 phương pháp tách từ nhóm đã làm ở trên

4.4 Kết quả và đánh giá

4.4.1 Phương pháp so sánh

a) Độ đo sử dụng

Độ chính xác Accuracy (P)

$$P = n/N$$

Trong đó :

- n là số từ ghép tách đúng của phương pháp so với dữ liệu chuẩn
- N là tổng số từ ghép có thể tách của bộ Test_gold

b) Các bước so sánh

Bước 1: Lấy kết quả tách từ 30 câu bộ Test của 3 phương pháp: MXM, WFST, VncoreNLP. Và bộ 30 câu tách tay chuẩn Test_gold

Bước 2: Đối với mỗi kết quả, phân tách câu thành các từ riêng biệt:

-***Ví dụ***:

['Tôi', ' khuyên', 'giáo', '_sư', ' thay', '_đổi', ' cách', ' nghĩ', ' của', ' mình', ' về', ' vấn', '_đề', ' này'] – Test_gold

['Tôi', ' khuyên', '_giáo', 'sư', ' thay', '_đổi', ' cách', ' nghĩ', ' của', ' mình', ' về', ' vấn', '_đề', ' này'] – Maximum Matching

Bước 3: Chạy theo index trên bộ gold, nếu phát hiện dấu '_' trước từ thì từ trước đó và từ tại vị trí index là 1 từ ghép, kiểm tra các từ sau nếu có _ thì cùng là 1 từ ghép.

-Trong trường hợp trên, trong bộ Test_gold, vì phát hiện dấu '_' trước từ 'sư' nên giáo sư là một từ ghép, index = 2 và length = 2

Bước 4: Chạy theo index và length trên các phương pháp tách từ, nếu giống với bộ Test_gold thì là tách đúng, tăng số từ tách đúng lên 1

-Trong trường hợp trên tại index=2 và length =2 từ '_giáo', 'sư' trong bộ MXM khác so với từ 'giáo', '_sư' trong bộ Test_gold nến cách tách này là sai.

Bước 5: Chạy trên tất cả các câu và thực hiện tính độ đo accuracy =n/N

-Ví dụ:

Tôi khuyên giáo_sư thay_đổi cách nghĩ của mình về vấn_đề này (Bộ test_gold)

Tôi khuyên_giáo sư thay_đổi cách nghĩ của mình về vấn_đề này (Kết quả tách từ MXM)

-Số từ ghép tách đúng của MXM: 2, Tổng số từ ghép có thể tách của bộ Test_gold: 3 => accuracy= 2/3

4.4.2 Kết quả

	Maximum Matching	WFST	VnCoreNLP
Accuracy	91,07 %	98,21%	92,85%

4.4.3 Đánh giá

- Nhìn vào bảng kết quả , có thể thấy cả 2 phương pháp nhóm thực hiện đều cho kết quả tốt đều trên 90% , trong đó phương pháp sử dụng thuật toán WFST cho kết quả rất cao với 98,21.

- Ưu nhược điểm

	Ưu điểm	Nhược điểm
Maximum Matching	- Cách tách từ đơn giản, nhanh, chỉ cần dựa vào từ điển. - Tách được các từ trong từ điển	- Độ chính xác của phương pháp phụ thuộc hoàn toàn vào tính đủ và tính chính xác của từ điển. - Nhập nhằng trong ngôn ngữ tự nhiên

WFST	- Độ chính xác cao. - Kết quả tách từ với độ tin cậy(xác suất) kèm theo - Giảm tính nhập nhằng	- Phụ thuộc vào độ lớn của bộ dữ liệu. - Phụ thuộc vào sự tương đồng giữa tập train và tập test

- Ví dụ về cách tách từ của 3 phương pháp:

- Câu ban đầu : Tôi không lưu ý đến việc có người đứng sau châm chọc tôi

- Câu sau khi tách từ :

  - MXM: Tôi không_lưu ý đến việc có người đứng sau châm_chọc tôi
  - WFST: Tôi không lưu_ý đến việc có người đứng sau châm_chọc tôi
  - VNCoreNLP: Tôi không_lưu ý đến việc có người đứng sau châm_chọc tôi

- Nhận xét :

  - Đối với MXM vì từ điển duyệt từ trái qua phải, vì 'không lưu' là từ đầu tiên xuất hiện trong từ điển nên phương pháp tách từ 'không lưu' là một dẫn tới sai về mặt ngữ nghĩa.
  - Đối với phương pháp WFST, vì trong bộ train có sự xuất hiện của từ 'lưu ý', cụ thể là câu 'tôi không lưu_ý đến việc này' nên xác suất của từ này cao hơn so với không_lưu => thuật toán tách từ theo từ 'lưu ý' và cách tách này là chính xác
  - Đối với VNCoreNLP, vì dữ liệu train của thư viện lớn hơn, nên từ 'không lưu' được tách thay vì 'lưu ý' dẫn tới cách tách là sai.

- Tổng quan: Tuy cách tách của phương pháp WFST là đúng , tuy nhiên phụ thuộc vào độ tương đồng giữa bộ dữ liệu train và bộ test

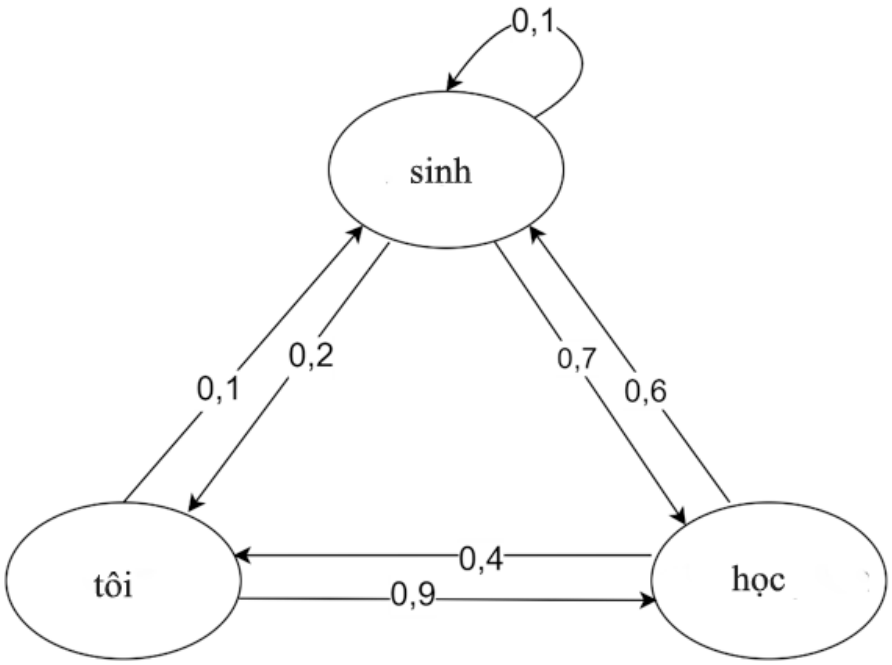
Chương 5: GÁN NHÃN TỪ LOẠI

5.1 Cơ sở lý thuyết

5.1.1 Mô hình Hidden Markov (HMM)

a) Markov Chain

Markov Chain (Xích Markov), hay Visible Markov Model, là một dạng FSA được dùng để mô hình hóa xác suất của các biến ngẫu nhiên có quan hệ với nhau theo dạng chuỗi.



Các nút tôi, học, sinh là các trạng thái, sự chuyển trại thái là các cạnh với xác suất kèm theo. Tổng xác suất các cạnh đi ra phải bằng 1

Một Markov Chain M là một bộ <Q, δ, q_0 >, trong đó:

- $Q = q_1 q_2 \ldots q_N$: một tập N **trạng thái**.
- δ là hàm chuyển đổi trạng thái có trọng số. Trọng số trong δ là xác suất chuyển trạng thái tương ứng.
- q_0 là trạng thái bắt đầu.

b) Giới thiệu HMM (Hidden Markov Models)

Mô hình Markov ẩn (Hidden Markov Models- HMM) là một trong những thuật toán được sử dụng phổ biến nhất trong Xử lý ngôn ngữ tự nhiên và là nền tảng cho nhiều kỹ thuật học sâu. Ngoài gán nhãn từ loại, HMM còn được dùng để nhận dạng giọng nói, tổng hợp giọng nói, …

Một xích Markov thường được sử dụng khi chúng ta cần tính một xác suất cho một chuỗi các sự kiện quan sát được. Nhưng khi các sự kiện đó bị ẩn và chúng ta không thể thấy nó trực tiếp thì xích Markov không thể giải quyết được. Nên giải pháp thay thế ở đây là mô hình Markov ẩn

Ví dụ, trong bài toán gán nhãn, chúng ta chỉ thấy các từ là các trạng thái quan sát được, chúng ta không thấy nhãn (từ loại) của nó là các trạng thái ẩn, và chúng ta muốn gán nhãn cho chuỗi các từ đó, thì chúng ta có thể sử dụng mô hình Markov ẩn. Nó cho phép chúng ta quan tâm đến cả các sự kiện *quan sát* và các sự kiện *ẩn*. Một mô hình Markov ẩn được chỉ rõ bởi các thành phần sau:

- S = $\{s_1 , s_2 , .., s_n \}$ là tập các trạng thái ẩn

- Trạng thái đặc biệt s_0 là trạng thái bắt đầu

- K = $\{k_1 , k_2 , .., k_m\}$ là tập các giá trị quan sát được

- A = $\{a_{ij}\}$, (i,j = 1..n) là ma trận chuyển trạng thái, trong đó a_{ij} là xác suất chuyển từ trạng thái s_i sang trạng thái s_j .

- B = $\{b_{ij}\}$ (i=1..n, j=1..m) là ma trận emisson (thể hiện), trong đó b_{ij} là xác suất trạng thái ẩn s_i thể hiện bằng giá trị quan sát k_j.

Một số giả thiết trong HMM:

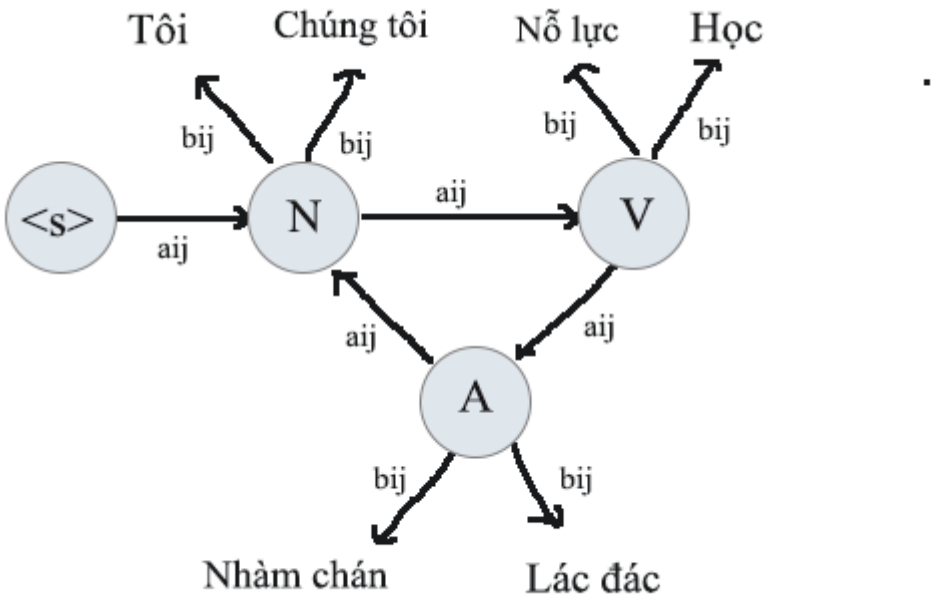
- Sự độc lập của các kết quả quan sát được: Kết quả quan sát chỉ phụ thuộc vào trạng thái ẩn hiện tại, không phụ thuộc vào các trạng thái ẩn trước đó.

$$p(O_t | X_1 .. X_T , O_1 .. O_T) = p(O_t | X_T).$$

- Giới hạn kinh nghiệm: trạng kế tiếp phụ thuộc vào trạng thái hiện tại và một số hữu hạn k các trạng thái trước đó:

$$p(X_T | X_1 .. X_{T-1}) = p(X_T | X_{T-k} X_{T-1})$$

Ví dụ:



-Trạng thái ẩn Hidden State : N, V, A

-Trạng thái bắt đầu: S

-Trạng thái quan sát được Observation: Tôi, Chúng tôi, Nỗ lực, học, lác đác, nhàm chán

-a_{ij} là các xác suất chuyển trạng thái, b_{ij} là các xác suất trong ma trận thể hiện (emission)

c) Laplace Smoothing

Giả sử ta có ma trận xác suất chuyển trạng thái A và ma trận thể hiện B, các chỉ số hiện tại trong ma trận có thể có giá trị 0, tuy nhiên đây không phải là giá trị tuyệt đối, vì ta chỉ làm việc với bộ train nhỏ, tức không có nghĩa là sẽ không có trường hợp nào. Nên với phương pháp Laplace, ta cộng tất cả các chỉ số này với 1. Việc này giúp ta

bao quát hơn tất cả các trường hợp, đồng thời giúp việc chia xác suất trở nên thuận tiện hơn

5.1.2 Thuật toán Viterbi

-Viterbi là thuật toán dùng để xác định được chuỗi trạng thái mà có xác suất đưa ra được chuỗi quan sát cho trước cao nhất , thuật toán này chọn các giá trị xác suất chuyển đổi lớn nhất tại mỗi bước để tối ưu các bước giải, thay vì tính tổng của chúng.

-Giả sử ta có mô hình HMM M=(A,B) trong đó A là ma trận xác suất chuyển trạng thái , B là ma trận thể hiện ,và một chuỗi quan sát được O, thuật toán viterbi giúp ta xác định chuỗi trạng thái ẩn X tương ứng với O sao cho p(X,O|M) là lớn nhất

-Các bước thực hiện:

B1: Khởi tạo 2 ma trận Prob và Pointer, trong đó Prob dùng để lưu trữ xác suất tốt nhất ở các bước, Pointer là ma trận dùng để lưu trữ các trạng thái ẩn trước đó , HMM M=(A,B),

B2:

- Số lượng trạng thái ẩn là n (không tính S)

- Gọi xác suất từ S chuyển sang các trạng thái ẩn khác trong ma trận A là A_{Si} , xác suất này được lấy trong ma trận A

- Gọi xác suất từ các trạng thái ẩn tương ứng với trạng thái quan sát được đầu tiên là B_{i0} xác suất này được lấy trong ma trận B

- Tính xác suất chuyển từ S (vị trí bắt đầu) sang trạng thái quan sát được đầu tiên:

Prob[i][0] = $A_{Si} * B_{i0}$, i chạy từ 0 đến n

-Đồng thời lưu giá trị trạng thái ẩn là S vào Ma trận pointer N:

Pointer[i][0]= S , i chạy từ 0 đến n

B3: Tính xác suất Max của các trạng thái ẩn tương ứng với trạng thái quan sát được ở trước chuyển sang trạng thái ẩn tương ứng với trạng thái quan sát được tiếp theo:

-Gọi k là bước xét trạng thái quan sát được hiện tại.

- i = 0, t = 0

-Xác suất trạng thái ẩn trước đó tương ứng với trạng thái quan sát trước đó: $Prob_{(k-1\ ,\ i\ )}$, i chạy từ 0 đến n

-Xác suất trạng thái ẩn hiện tại tương ứng với trạng thái quan sát được hiện tại là $B_{(\ t\ )}$

-Gọi xác suất chuyển từ trạng thái ẩn trước đó sang trạng thái ẩn hiện tại là $A_{(i\ ,\ t)}$

⇨ Công thức:

Prob[k][t] = max ($Prob_{(k-1\ ,\ i\ )} * B_{(t)} * A_{(i\ ,\ t)}$) với i chạy từ 1 đến n.

Sau mỗi lần tính Prob[k][t] ta lần lượt tăng t đến n để tính các xác suất của trạng thái ẩn còn lại tại bước k

-Đồng thời lưu giá trị trạng thái ẩn tương ứng với giá trị max trước đó vào Ma trận Pointer.

-Lặp lại bước 3 cho đến khi xét hết trạng thái quan sát được

B4 :Trong ma trận xác suất M, ở cột cuối cùng ta tìm được giá trị max cũng là xác suất cao nhất của đường đi tốt nhất. Dựa vào ma trận Pointer ta truy ngược kết quả và kết thúc

5.2 Triển khai thực hiện

a) Hidden Markov Model:

-Data nhóm sử dụng: Bộ PosTrain gồm 50 câu đã được tách tay chuẩn và gán nhãn dùng để train model

-Xử lí dữ liệu đầu vào:

+Từ bộ PosTrain, nhóm tạo một tập các trạng thái ẩn, một tập các trạng thái quan sát được, các trạng thái này không lặp lại, đồng thời trong tập trạng thái quan sát được, nhóm thêm vào các trạng thái 'unk' dành cho các trường hợp trạng thái chưa gặp bao giờ

Ví dụ: Giả sử bộ PosTrain gồm 3 câu sau:

- học_sinh/N lác_đác/A trên/E sân_trường/N
- chúng_tôi/P đã/R nỗ_lực/V hết_sức/R
- công_việc/N văn_phòng/N khiến/V tôi/P cảm_thấy/V nhàm_chán/A

⇨ Trạng thái ẩn: ['S', 'A', 'E', 'N', 'P', 'R', 'V']

⇨ Trạng thái quan sát được: ['chúng_tôi', 'công_việc', 'cảm_thấy', 'hết_sức', 'học_sinh', 'khiến', 'lác_đác', 'nhàm_chán', 'nỗ_lực', 'sân_trường', 'trên', 'tôi', 'văn_phòng', 'đã', 'unk1', 'unk2']

- Tiếp đến nhóm khởi tạo ma trận chuyển trạng thái A và ma trận thể hiện B

- Sau đó từ bộ PosTrain, nhóm đếm số lượng trạng thái ẩn i chuyển sang trạng thái ẩn j, trong đó j không bao gồm trạng thái ẩn S

- Ứng với mỗi số đếm được, ta cộng thêm 1 vì nhóm sử dụng phương pháp Smooth Laplace

-Tính tổng tất cả các số trong mỗi hàng vào cột Sum và cập nhật vào ma trận A:

Ví dụ: Tương ứng ví dụ trên, ta có ma trận A:

	A	E	N	P	R	V	Sum
S	('1')	('1')	('3')	('2')	('1')	('1')	('9')
A	('1')	('2')	('1')	('1')	('1')	('1')	('7')
E	('1')	('1')	('2')	('1')	('1')	('1')	('7')
N	('2')	('1')	('2')	('1')	('1')	('2')	('9')
P	('1')	('1')	('1')	('1')	('2')	('2')	('8')
R	('1')	('1')	('1')	('1')	('1')	('2')	('7')
V	('2')	('1')	('1')	('2')	('2')	('1')	('9')

-Tiếp đến tính xác suất chuyển từ trạng thái ẩn i sang trạng thái ẩn j bằng các lấy A[i][j] / Sum[i]:

	A	E	N	P	R	V
S	('0.111')	('0.111')	('0.333')	('0.222')	('0.111')	('0.111')
A	('0.143')	('0.286')	('0.143')	('0.143')	('0.143')	('0.143')
E	('0.143')	('0.143')	('0.286')	('0.143')	('0.143')	('0.143')
N	('0.222')	('0.111')	('0.222')	('0.111')	('0.111')	('0.222')
P	('0.125')	('0.125')	('0.125')	('0.125')	('0.250')	('0.250')
R	('0.143')	('0.143')	('0.143')	('0.143')	('0.143')	('0.286')
V	('0.222')	('0.111')	('0.111')	('0.222')	('0.222')	('0.111')

-Tương tự với ma trận A, ta đếm số lượng trạng thái ẩn i tương ứng với trạng thái quan sát được j, sử dụng smooth Laplace và cập nhật vào ma trận thể hiện B:

	chúng_tôi	công_việc	cảm_thấy	hết_sức	học_sinh	khiến	lác_đác	nhàm_chán	nỗ_lực	sân_trường	trên	tôi	văn_phòng	đã	unk1	unk2	Sum
A	('1')	('1')	('1')	('1')	('1')	('1')	('2')	('2')	('1')	('1')	('1')	('1')	('1')	('1')	('1')	('1')	('18')
E	('1')	('1')	('1')	('1')	('1')	('1')	('1')	('1')	('1')	('1')	('2')	('1')	('1')	('1')	('1')	('1')	('17')
N	('1')	('2')	('1')	('1')	('2')	('1')	('1')	('1')	('1')	('2')	('1')	('1')	('2')	('1')	('1')	('1')	('20')
P	('2')	('1')	('1')	('1')	('1')	('1')	('1')	('1')	('1')	('1')	('1')	('2')	('1')	('1')	('1')	('1')	('18')
R	('1')	('1')	('1')	('2')	('1')	('1')	('1')	('1')	('1')	('1')	('1')	('1')	('1')	('2')	('1')	('1')	('18')
V	('1')	('1')	('2')	('1')	('1')	('2')	('1')	('1')	('2')	('1')	('1')	('1')	('1')	('1')	('1')	('1')	('19')

-Tính xác suất tương ứng với tổng:

	chúng_tôi	công_việc	cảm_thấy	hết_sức	học_sinh	khiến	lác_đác	nhàm_chán	nỗ_lực	sân_trường	trên	tôi	văn_phòng	đã	unk1	unk2
A	('0.056')	('0.056')	('0.056')	('0.056')	('0.056')	('0.111')	('0.111')	('0.056')	('0.056')	('0.056')	('0.056')	('0.056')	('0.056')	('0.056')	('0.056')	('0.056')
E	('0.059')	('0.059')	('0.059')	('0.059')	('0.059')	('0.059')	('0.059')	('0.059')	('0.059')	('0.059')	('0.118')	('0.059')	('0.059')	('0.059')	('0.059')	('0.059')
N	('0.050')	('0.100')	('0.050')	('0.050')	('0.100')	('0.050')	('0.050')	('0.050')	('0.050')	('0.100')	('0.050')	('0.050')	('0.100')	('0.050')	('0.050')	('0.050')
P	('0.111')	('0.056')	('0.056')	('0.056')	('0.056')	('0.056')	('0.056')	('0.056')	('0.056')	('0.056')	('0.056')	('0.111')	('0.056')	('0.056')	('0.056')	('0.056')
R	('0.056')	('0.056')	('0.056')	('0.111')	('0.056')	('0.056')	('0.056')	('0.056')	('0.056')	('0.056')	('0.056')	('0.056')	('0.056')	('0.111')	('0.056')	('0.056')
V	('0.053')	('0.053')	('0.105')	('0.053')	('0.053')	('0.105')	('0.053')	('0.053')	('0.105')	('0.053')	('0.053')	('0.053')	('0.053')	('0.053')	('0.053')	('0.053')

b) Viterbi

B1) Tính ma trận xác suất Prob và ma trận Pointer:

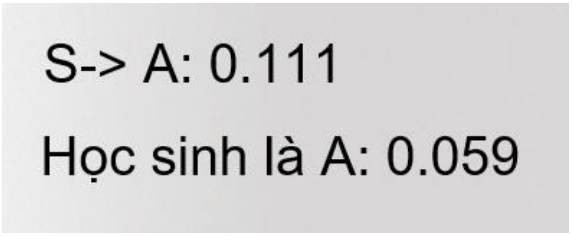
Dựa vào mã giả của ma trận Viterbi đã trình bày ở trên, nhóm tiến hành tạo thuật toán đế tính ma trận xác suất Prob, các bước cụ thể:

+ Khởi tạo ma trận xác suất Prob và ma trận pointer

+ Tính xác suất chuyển từ S (vị trí bắt đầu) sang trạng thái quan sát được đầu tiên

Ví dụ: Tương ứng ví dụ ở trên , ta sẽ tính xác suất đường đi tốt nhất cho câu:

‘Học sinh đang cảm thấy nhàm chán’:



Trong đó xác xuất chuyển từ S sang Học sinh là A = Xác suất chuyển từ S sang A * Xác suất Học sinh là A = 0.111 * 0.059 ~ 0.0065. Thực hiện tương tự với các cột còn lại

+Tiếp đến ta tính xác suất Max của các trạng thái ẩn tương ứng với trạng thái quan sát được ở trước chuyển sang trạng thái ẩn tương ứng với trạng thái quan sát được tiếp theo.

Trong ví dụ này ta tính xác suất Max chuyển từ các trạng thái ẩn của ‘Học sinh’ sang các trạng thái ẩn của ‘đang’:

Đầu tiên tính xác suất Max chuyển từ các trạng thái ẩn của ‘Học sinh’ sang trạng thái ẩn ‘A’ của ‘đang’:

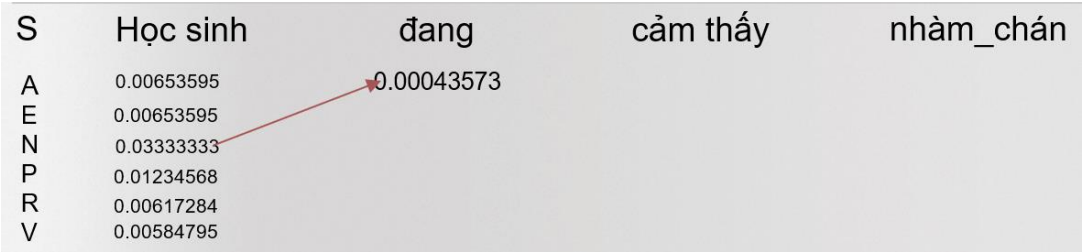
Ta có: Xác suất đang là A: 0.59,

Xác suất A-> A: 0.111,

Xác suất học sinh là A: 0.00653595

⇨ Xác suất học sinh là A chuyển sang đang là A là 0.59*0.111* 0.00653595 ~ 0.0000425685

Tương tự ta tính Xác suất 'học sinh' là E, N, P, R, V chuyển sang 'đang' là A, cuối cùng ta tìm được giá trị Max là 0.00043573 tương ứng với trạng thái ẩn là N chuyển sang.



+Đồng thời ta lưu trữ giá trị N này vào bảng Ma trận Pointer.

+Tiếp tục tính tương tự với các giá trị còn lại ta được kết quả:

S	Học sinh	đang	cảm thấy	nhàm_chán
A	0.00653595	0.00043573	0.00000481	2.13916898E-7
E	0.00653595	0.00021786	0.00000692	8.08929449E-8
N	0.03333333	0.00037037	0.00000412	9.88049540E-8
P	0.01234568	0.00020576	0.00000481	1.06958449E-7
R	0.00617284	0.00020576	0.00000481	1.06958449E-7
V	0.00584795	0.00038986	0.00000866	7.23778980E-8

Ma trận Prob

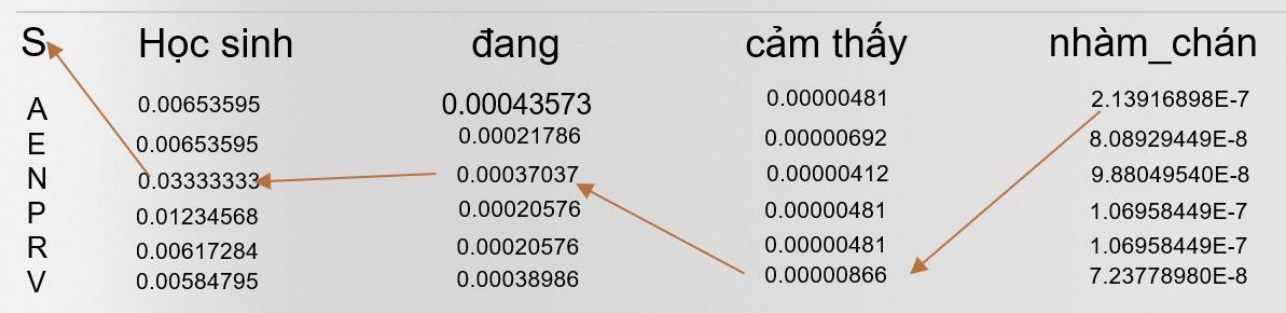
	Học sinh	đang	cảm thấy	nhàm chán
A	S	N	V	V
E	S	N	A	A
N	S	N	N	E
P	S	N	V	V
R	S	N	V	V
V	S	N	N	R

Ma trận pointer

B2) Truy ngược viterbi

Dựa vào ma trận xác suất Prob, nhóm tìm được giá trị max trên cột cuối cùng và truy ngược lại dựa vào giá trị lưu trong ma trận Pointer

Tương ứng với ví dụ trên ta có



⇨ Kết quả : Học sinh/N đang/N cảm thấy/V nhàm chán/A

Nhận xét: từ đang không xuất hiện trong bộ train nên phương pháp gán nhãn sai.

Cuối cùng, nhóm sử dụng bộ PosTest để chạy thử và so sánh kết quả với bộ PosTest_gold để kiểm tra

5.3 Kết quả và đánh giá

5.3.1 Phương pháp so sánh

Độ chính xác Accuracy (P):

$$P = n/N$$

Trong đó :

- n là số nhãn gán đúng
- N là tổng số nhãn gán

5.3.2 Kết quả

	HMM + Viterbi nhóm cài đặt	HMM của thư viện NLTK (smooth='laplace')	HMM của thư viện NLTK (smooth='WittenBell')
ACURACY	0,854	0,854	0,936

5.3.1 Đánh giá:

- Accuracy phương pháp nhóm cài đặt có kết quả khá cao , có thể cải thiện kết quả bằng smooth Written Bell
- Bộ ngữ liệu để train còn ít, kết quả xử lí thấp với những trường hợp từ chưa gặp

Chương 6: TỔNG KẾT

6.1 Kết luận

Trong đề tài này, nhóm đã áp dụng các kiến thức về xử lý ngôn ngữ tự nhiên để xây dựng bộ tách từ bằng thuật toán Maximum Matching và WFST(Weighted Finite State Transducer), cùng với đó là xây dựng được mô hình Hidden Markov kết hợp thuật toán Viterbi để gán nhãn cho các từ đã tách. Với việc tách từ thì thuật toán WFST cho kết quả cao hơn khá nhiều so với Maximum Matching, còn với việc gán nhãn thì mô hình cho kết quả khá tốt.

6.2 Hướng phát triển

Nhóm sẽ tìm hiểu thêm những cách triển khai khác sử dụng gán nhãn 2 chiều (Bidirectional POS tagging). Gán nhãn 2 chiều yêu cầu biết được từ trước đó và từ tiếp theo trong ngữ liệu khi dự đoán nhãn của từ hiện tại. Gán nhãn 2 chiều sẽ cho ta biết thêm về nhãn thay vì chỉ biết từ trước đó. Vì đã học được cách triển khai phương pháp tiếp cận đơn hướng qua đề tài này, nhóm đã có nền tảng để triển khai các trình gán nhãn khác được sử dụng trong thực tế.

TÀI LIỆU THAM KHẢO

[1] Lý thuyết hidden Markov model

Địa chỉ: https://web.stanford.edu/~jurafsky/slp3/

[2] Lý thuyết tách từ và gán nhãn của thầy Nguyễn Trọng Chỉnh.

[3] Bộ nhãn tham khảo Vncore NLP

Địa chỉ: *https://github.com/vncorenlp/VnCoreNLP/blob/master/VLSP2013_POS_tagset.pdf*

[4] Từ điển tham khảo : Từ điển tiếng việt Underthesea

Địa chỉ: https://github.com/undertheseanlp/dictionary/

PHỤ LỤC – BẢNG PHÂN CÔNG ĐÁNH GIÁ THÀNH VIÊN

Họ và tên	MSSV	Phân công	Đánh giá
Hồ Anh Dũng	18520630	- Thu thập dữ liệu - Tạo ngữ liệu - Thực hiện phương pháp tách từ Maximum Matching - Thực hiện phương pháp gán nhãn - Viết báo cáo , làm slide	Hoàn thành tốt
Nguyễn Như Long	19521790	- Thu thập dữ liệu - Tạo ngữ liệu - Thực hiện phương pháp tách từ WFST - Thực hiện phương pháp gán nhãn - Viết báo cáo, làm slide	Hoàn thành tốt