**ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

🙤🙧🟍🙥🙦



**GÁN NHÃN TỪ LOẠI TIẾNG VIỆT**

**VỚI MÔ HÌNH HIDDEN MARKORV**

|  |  |  |
| --- | --- | --- |
| Giảng viên hướng dẫn: | | |
| Th.S Nguyễn Trọng Chỉnh | | |
| Danh sách thành viên: | | |
| STT | Họ tên | MSSV |
| 1 | Đặng Hoàng Quân | 18520339 |
| 2 | Phạm Phú Phước | 18520131 |
| 3 | Dương Quốc Lộc | 18521006 |

**TP. HỒ CHÍ MINH – 01/2021**

# MỤC LỤC

[MỤC LỤC 2](#_Toc62765635)

[TÓM TẮT 4](#_Toc62765636)

[CHƯƠNG 1: GIỚI THIỆU 5](#_Toc62765637)

[CHƯƠNG 2: THU THẬP DỮ LIỆU 6](#_Toc62765638)

[2.1. Nguồn thu thập 6](#_Toc62765639)

[2.2. Thông tin cơ bản 6](#_Toc62765640)

[CHƯƠNG 3: TÁCH TỪ 7](#_Toc62765641)

[3.1. Thuật toán Longest Matching 7](#_Toc62765642)

[3.1.1. Giới thiệu 7](#_Toc62765643)

[3.1.2. Mã giả 7](#_Toc62765644)

[3.1.3. Ưu và nhược điểm 8](#_Toc62765645)

[3.2. Cách triển khai 8](#_Toc62765646)

[3.3. Đánh giá kết quả 9](#_Toc62765647)

[CHƯƠNG 4: TẠO NGỮ LIỆU VÀ GÁN NHÃN 10](#_Toc62765648)

[4.1 Tạo ngữ liệu 10](#_Toc62765649)

[4.2. Xây dựng model gán nhãn từ loại 13](#_Toc62765650)

[4.2.2. Mô hình Hidden Markov 13](#_Toc62765651)

[4.2.3. Thuật toán Viterbi 16](#_Toc62765652)

[4.4. Thực nghiệm 17](#_Toc62765653)

[CHƯƠNG 5: KẾT QUẢ 19](#_Toc62765654)

[5.1. Các độ đo được sử dụng 19](#_Toc62765655)

[5.1.1. Độ đo Accuracy 19](#_Toc62765656)

[5.1.2. Độ đo Precision 19](#_Toc62765657)

[5.1.3. Độ đo Recall 19](#_Toc62765658)

[5.1.4. Độ đo F1-score 19](#_Toc62765659)

[5.2. Kết quả 20](#_Toc62765660)

[5.3. Nhận xét 20](#_Toc62765661)

[CHƯƠNG 6: KẾT LUẬN 21](#_Toc62765662)

# TÓM TẮT

# CHƯƠNG 1: GIỚI THIỆU

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. 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.

Trong đề tài này nhóm sẽ sử dung mô hình Hidden Markov kết hợp thuật toán Viterbi để gán nhãn từ loại tiếng Việt và so sánh độ chính xác với VnCoreNLP

# CHƯƠNG 2: THU THẬP DỮ LIỆU

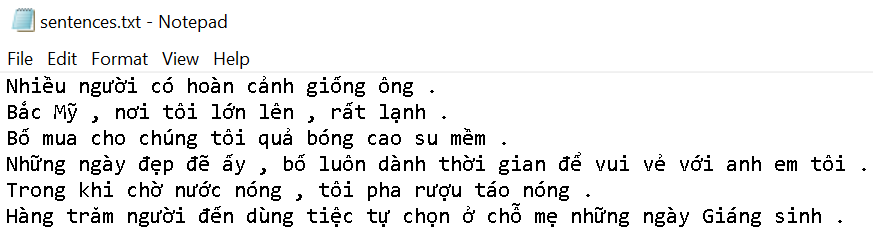
## Nguồn thu thập

Gồm các câu bất kỳ thuộc nhiều chủ đề được thu thập từ 2 trang <https://vnexpress.net/> và <https://dantri.com.vn/>. Do đây là 2 nguồn báo uy tín ở Việt Nam nên nhóm sẽ lựa chọn để sử dụng cho việc thu thập dữ liệu.

## Thông tin cơ bản

Bộ dữ liệu gốc được lưu với tên sentences.txt:

* Mỗi dòng là 1 câu.
* Các từ được phân cách với nhau bằng dấu cách ‘ ’.
* Cuối câu là 1 dấu chấm ‘.’.
* Số lượng câu: 60 câu.
* Câu có số từ nhiều nhất: 46 từ.
* Câu có số từ ít nhất: 8 từ.



Hình 2: Một số câu có trong bộ dữ liệu gốc.

# CHƯƠNG 3: TÁCH TỪ

Bài toán tách từ là bài toán quan trọng đối với tiếng Việt. Khác với tiếng Anh, một từ tiếng Việt có thể được tạo bởi nhiều hơn một âm. Ví dụ từ (word) “cá\_nhân” được tạo lên bởi 2 âm (syllable) là “cá” và “nhân”. Trong khi hai từ đơn “cá” và từ đơn “nhân” lại có thể mang ý nghĩa khác.

Do vậy, tách từ tiếng Việt là bước quan trọng chúng ta cần thực hiện trước khi đưa dữ liệu vào các bước tiếp theo, ví dụ như word embedding.

## Thuật toán Longest Matching

### Giới thiệu

Longest Matching là thuật toán dựa trên tử tưởng tham lam. Nó xét các tiếng từ trái qua phải, các tiếng đầu tiên dài nhất có thể mà xuất hiện trong từ điển sẽ được tách ra làm một từ. Với vị trí âm tiết hiện tại sẽ kiểm tra xem từ đó và 2 âm tiếp theo có thể ghép thành 1 từ có nghĩa hay không bằng cách kiểm tra trong từ điển tri\_gram. Nếu không thể tạo ra được từ có nghĩa từ 3 âm tiết thì ta tiếp tục kiểm tra xem âm tiết hiện tại và âm tiếp theo có thể ghép được thành một từ có nghĩa hay không bằng cách kiểm tra trong từ điển bi\_gram. Thuật toán sẽ dừng khi xét hết các tiếng trong câu.

### Mã giả

current\_word \_id = 0 (Đặt từ hiện tại ở đầu câu):

While (curret\_word\_id+1) != len(text) {

xét từ hiện tại với 2 từ sau nó trong từ điển tri\_gram

check if {“word, word+1, word+2”} is in tri\_gram

if true: take {word, word1, word2; current\_word\_id += 3}🡪 3 từ đó tạo thành từ ghép

xét từ hiện tại với tiếng sau trong từ điển trong từ điển bi\_gram

else check if: take {“word, word+1”} is in bi\_gram

if true: take {word, word1; current\_word\_id += 2}🡪 2 tiếng đó tạo thành từ ghép

else => {take word; current\_word\_id += 1} 🡪 tiếng đó là từ đơn

}

### Ưu và nhược điểm

Ưu điểm thuật toán này chính là cài đặt đơn giản, độ phức tạp tính toán hợp lý, không yêu cầu dữ liệu huấn luyện. Tuy nhiên, phương pháp này sẽ khó có thể xử lý được các tình huống nhập nhằng nhất định như việc lặp từ trong câu cũng như xử lý tình huống xuất hiện từ mới không tồn tại trong từ điển

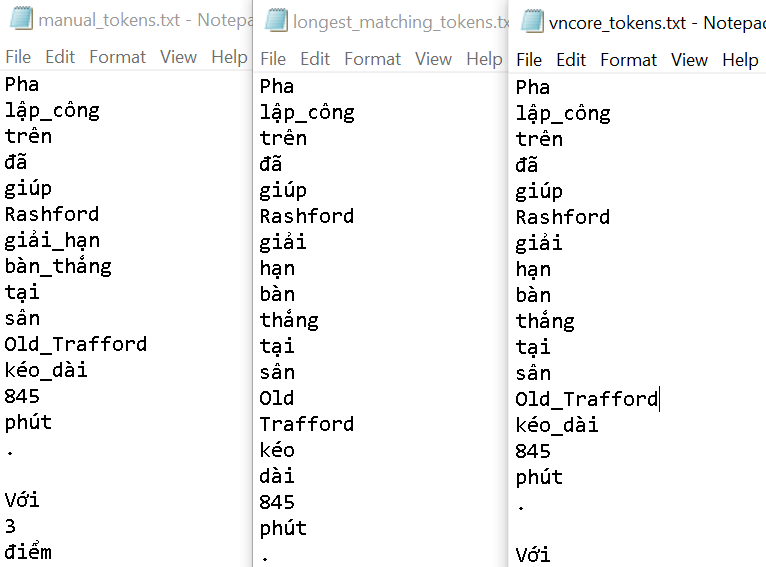
## Cách triển khai

Nhóm sẽ thực hiện tách từ bán thủ công: sử dụng thư viện VnCoreNLP để tách từ trước sau đó sẽ phân công các thành viên kiểm tra thủ công lại kết quả tách từ và sửa lại các từ bị sai để thu được 1 bộ dữ liệu mới chặt chẽ hơn.

Bộ dữ liệu mới được lưu với tên manual\_tokens.txt sẽ chứa kết quả tách từ đúng nhất cho 60 câu nhóm mà đã thu thập:

* Các âm tiết của từ ghép sẽ được nối bằng dấu gạch dưới ‘\_’.
* Mỗi dòng là 1 từ.
* Mỗi câu sẽ được ngăn cách bằng 1 dòng trống.
* Số lượng câu: 60 câu.
* Số lượng từ: 970 từ.
* Số lượng từ ghép: 302 từ

Sau đó, nhóm sẽ sử dụng bộ dữ liệu này làm chuẩn và đánh giá kết quả tách từ của 2 phương pháp: sử dụng thuật toán Longest Matching và sử dụng thư viện VnCoreNLP, với kết quả tách từ của mỗi phương pháp lần lượt được lưu trong các file có tên là longest\_matching\_tokens.txt chứa 1049 từ với 257 từ ghép và vncore\_tokens.txt chứa 1006 từ với 282 từ ghép.



Hình 3: Kết quả tách từ theo từng phương pháp khác nhau của câu đầu tiên

## Đánh giá kết quả

Bảng 1: Tổng hợp kết quả tách từ.

|  |  |  |
| --- | --- | --- |
|  | Longest Matching | VnCoreNLP |
| Accuracy | 0.855876 | 0.932373 |
| Precision | 0.888446 | 0.949275 |
| Recall | 0.738411 | 0.867550 |
| True Positive | 223 | 262 |
| False Positive | 28 | 14 |
| Total True | 772 | 841 |
| Total Errors | 208 | 97 |

Qua bảng 1, có thể thấy rằng các số liệu đánh giá của việc tách từ khi sử dụng thư viện VnCoreNLP đều rất tốt và tốt hơn nhiều so với khi dùng thuật toán Longest Matching đồng thời nhóm nhận thấy Longest Matching không phân biệt tốt được các tên riêng như Old\_Trafford, … có thể thấy ở hình 3. Đây có thể do bộ dữ liệu bi-grams và tri-grams được dùng cho thuật toán này chưa đủ tốt.

Vì quá trình tách từ bằng thuật toán và thư viện đều có thể tồn tại những sai sót nên nhóm quyết định sẽ sử dụng bộ dữ liệu có được sau khi thực hiện tách từ bán thủ công (ở đây là file manual\_tokens.txt) để bắt đầu tiến hành gán nhãn từ loại.

# CHƯƠNG 4: TẠO NGỮ LIỆU VÀ GÁN NHÃN

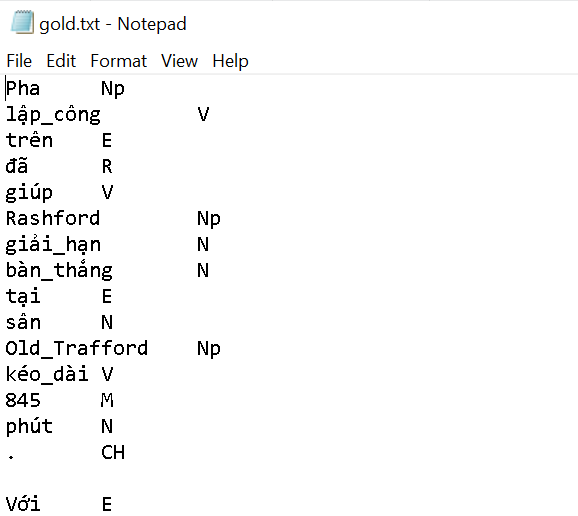
## 4.1 Tạo ngữ liệu

Sau khi có kết quả tách từ, nhóm lại tiếp tục tiến hành gán nhãn bán thủ công trên bộ dữ liệu tách từ đã chọn.

Nhóm sẽ sử dụng thư viện VnCoreNLP để gán nhãn trước, sau đó sẽ phân công các thành viên kiểm tra thủ công lại kết quả gán nhãn và sửa lại các nhãn bị sai để thu được tập dữ liệu Gold cho 60 câu đã thu thập:

Bộ dữ liệu Gold được lưu với tên gold.txt:

* Các âm tiết của từ ghép sẽ được nối bằng dấu gạch dưới ‘\_’.
* Từ được phân tách với nhãn bằng dấu tab.
* Mỗi dòng là 1 từ kèm nhãn.
* Mỗi câu sẽ được ngăn cách bằng 1 dòng trống.
* Số lượng câu: 60
* Số lượng từ: 970.



Hình 4: Một số từ kèm nhãn trong bộ dữ liệu GOLD

Danh sách nhãn từ loại mà nhóm sử dụng:

*Bảng 2: Danh sách nhãn từ loại*

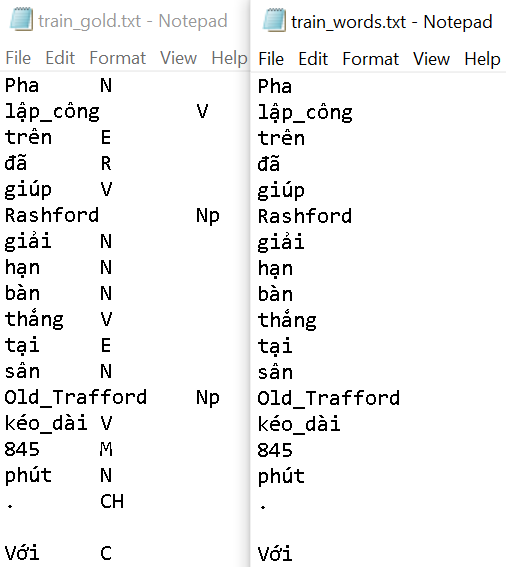
*(*[*https://github.com/vncorenlp/VnCoreNLP/blob/master/VLSP2013\_POS\_tagset.pdf*](https://github.com/vncorenlp/VnCoreNLP/blob/master/VLSP2013_POS_tagset.pdf)

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Nhãn** | **Tên** | **Ví dụ** |
| 1 | N | Danh từ | tiếng, nước, thủ đô, nhân dân, đồ đạc, cây cối, chim muông |
| 2 | Np | Danh từ riêng | Nguyễn Du, Việt Nam, Hải Phòng, Trường Đại học Bách khoa Hà Nội, Mộc tinh, Hoả tinh, Phật, Đạo Phật |
| 3 | Nc | Danh từ chỉ loại | con, cái, đứa, bức |
| 4 | Nu | Danh từ đơn vị | mét, cân, giờ, nắm, nhúm, hào, xu, đồng |
| 5 | Ni | Danh từ ký hiệu | A1, A4, 60A, 60B, 20a, 20b, ABC, ABCD |
| 6 | V | Động từ | ngủ, ngồi, cười; đọc, viết, đá, đặt;  thích, yêu, ghét, giống, muốn |
| 7 | A | Tính từ | tốt, xấu, đẹp; cao, thấp, rộng |
| 8 | P | Đại từ | tôi, chúng tôi, hắn, nó, y, đại nhân, đại  ca, huynh, đệ |
| 9 | L | Định từ | mỗi, từng, mọi, cái; các, những, mấy |
| 10 | M | Số từ | một, mười, mười ba; dăm, vài, mươi;  nửa, rưỡi |
| 11 | R | Phó từ | đã, sẽ, đang, vừa, mới, từng, xong, rồi;  rất, hơi, khí, quá |
| 12 | E | Giới từ | trên, dưới, trong, ngoài; của, trừ,  ngoài, khỏi, ở |
| 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 | . ! ? , ; : |

Tiếp theo nhóm sẽ chia 60 câu đã được tách từ và gán nhãn trong bộ dữ liệu Gold này thành 2 tập dữ liệu: 50 câu cho tập Train và 10 câu cho tập Test.

Tập Train gồm các file train\_gold.txt và train\_words.txt:

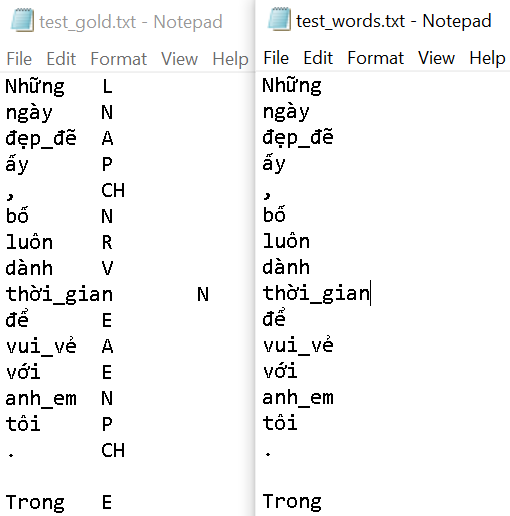
* train\_gold.txt: chứa các từ kèm nhãn để thực hiện cho việc huấn luyện
* train\_words.txt: chỉ chứa từ của các câu để thực hiện cho việc kiểm thử trên tập train (kiểm tra overfitting).
* Số lượng câu của tập Train: 50 câu
* Số lượng từ của tập Train: …từ



Hình 5: Một số từ trong tập Train

Tập Test gồm các file test\_gold.txt và test\_words.txt:

* test\_gold.txt: chứa các từ kèm nhãn để thực hiện cho việc đánh giá kết quả gán nhãn
* test\_words.txt: chỉ chứa từ của các câu để thực hiện cho việc dự đoán
* Số lượng câu của tập Test: 10 câu
* Số lượng từ của tập Test: … từ



Hình 6: Một số từ trong tập Test

## 4.2. Xây dựng model gán nhãn từ loại

### 4.2.2. Mô hình Hidden Markov

**Markov chain**

Mô hình Hidden Markov (HMM) dựa trên Markov chain (xích Markov). Một Markov chain là một mô hình cho chúng ta biết về xác suất của một chuỗi các biến ngẫu nhiên hay còn được gọi là các trạng thái, mỗi trạng thái có thể nhận những giá trị từ một tập nào đó. Các tập này có thể là tập các từ, các nhãn hoặc những biểu tưởng trình bày một điều gì đó. Một Markov chain thì đưa ra một giả định rằng nếu dự đoán tương lai của chuỗi thì những trạng thái hiện tại là điều quan trọng nhất để dự đoán. Tương lai thì không bị ảnh hưởng bởi tất cả trạng thái trước trạng thái hiện tại mà nó thông qua trạng thái hiện tại. Ví dụ nếu bạn muốn dự báo thời tiết của ngày mai, bạn có thể dựa vào thời tiết của hôm nay nhưng không được phép xem thời tiết của hôm qua.

Một cách tổng quát hóa, xét chuỗi các biến trạng thái . Mô hình Markov thể hiện giả định Markov về các xác suất của chuỗi này đó là khi dự đoán tương lai không quan trọng quá khứ mà chỉ cần hiện tại.

Giả định Markov:

Một Markov chain được xác định bởi các thành phần sau:

: là một tập của N trạng thái.

: là một ma trận chuyển trạng thái A, mỗi phần tử thể hiện xác suất chuyển từ trạng thái đến trạng thái với ràng buộc là .

: là một phân phối xác suất ban đầu trên mỗi trạng thái. là xác suất mà Markov chain sẽ bắt đầu ở trạng thái , một vài trạng thái có thể có , có nghĩa là chúng không được khởi tạo phân phối xác suất ban đầu, nó cũng có một ràng buộc là .

**Mô hình Hidden Markov (HMM)**

Một chuỗi Markov thì hữu ích khi chúng ta tính cho chuỗi các sự kiện có thế quan sát được. Tuy nhiên, trong nhiều trường hợp có những sự kiện mà chúng ta quan tâm thì nó ẩn, có nghĩa là chúng ta không thể quan sát nó một cách trực tiếp. Ví dụ chúng ta thông thường không thấy nhãn từ loại của từ trong văn bản. Thay vào đó, chúng ta thấy các từ và phải suy ra nhãn từ loại từ chuỗi các từ. Vì thế, chúng ta gọi là nhãn từ loại ẩn bởi vì chúng không được quan sát.

HMM cho phép chúng ta đề cập về cả những trạng thái được quan sát (observed states) và những trạng thái ẩn (hidden states). HMM được xác định bởi các thành phần sau:

: là một tập của N trạng thái.

: là một ma trận chuyển trạng thái A, mỗi phần tử thể hiện xác suất chuyển từ trạng thái đến trạng thái với ràng buộc là .

: là một chuỗi có quan sát, mỗi quan sát được rút ra từ một từ vựng .

: là một chuỗi quan sát likelihoods, còn được gọi là emission probabilities, mỗi thể hiện xác suất của quan sát được tạo ra từ một trạng thái .

: là một phân phối xác suất ban đầu trên mỗi trạng thái. là xác suất mà Markov chain sẽ bắt đầu ở trạng thái , một vài trạng thái có thể có , có nghĩa là chúng không được khởi tạo phân phối xác suất ban đầu, nó cũng có một ràng buộc là .

HMM bậc nhất đưa ra hai giả định. Thứ nhất, giống với chuỗi Markov, xác suất của một trạng thái cụ thể chỉ phụ thuộc vào trạng thái trước đó:

Thứ hai, xác suất của một đầu ra quan sát chỉ phụ thuộc vào trạng thái mà đã tạo ra quan sát chứ không phải phụ thuộc vào bất kì trạng thái nào khác hoặc quan sát nào khác:

Các thành phần của HMM cho việc gán nhãn từ loại:

Một HMM gồm hai thành phần là ma trận chuyển trạng thái A và ma trận emission xác suất B. Ma trận A chứa các xác suất chuyển đổi nhãn là xác suất xảy ra của nhãn khi biết nhãn trước đó. Chúng ta tính maximum likelihood estimate của xác suất chuyển trạng thái này bằng cách đếm số lần nhãn từ loại thứ nhất trong corpus đã được gán nhãn từ loại mà được theo bởi nhãn thứ hai:

Đối với ma trận xác suất emission B, là xác suất mà một nhãn từ loại đã cho đi với một từ đã cho. MLE của xác suất emission là:

**Thực thi gán nhãn từ loại bằng HMM**

Cho đầu vào là HMM và một chuỗi quan sát , tìm chuỗi trạng thái có thể xảy ra nhất .

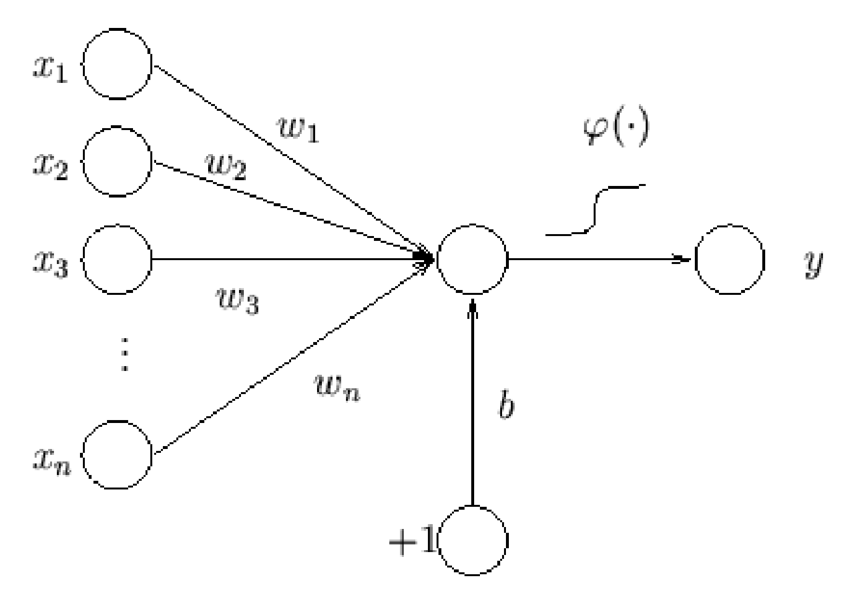
Đối với gán nhãn từ loại, mục tiêu của HMM là tìm chuỗi nhãn từ loại có khả năng cao nhất là nhãn của chuỗi quan sát gồm từ :

*1:n*

*1:n*, với

### 4.2.3. Thuật toán Viterbi

Thuật toán Perceptron được phát minh vào năm 1958 tại Phòng thí nghiệm Hàng không Cornell bởi Frank Rosenblatt. Đây là một trong những thuật toán mạng neuron cơ bản và đơn giản nhất khi chỉ có 3 lớp là một lớp đầu vào, một lớp ẩn và một lớp đầu ra.



*Hình 11: Kiến trúc của thuật toán Perceptron.*

Thuật toán Perceptron gồm các bước sau: khởi tạo, tính toán giá trị và cập nhật tham số**.**

* Khởi tạo các giá trị trọng số. Thường khởi tạo bằng 0 hoặc các giá trị nhỏ.
* Tính toán giá trị đầu ra theo công thức
* Cập nhật tham số:

Trong đó w là trọng số, x là giá trị dữ liệu và là đạo hàm của W theo y.

Khi qua đến cuối lớp ẩn, ta tính được giá trị đầu ra, từ đó tính được hàm mất mát với công thức: . Trong đó Y là giá trị thực tế và y là giá trị dự đoán. Việc sử dụng vòng lặp để lặp đến khi lượng mất mát này giảm đến một mức chấp nhận được.

## 4.4. Thực nghiệm

Thông qua nhiều lần chạy mô hình để tìm ra các bộ tham số tối ưu nhất. Đối với thuật toán đầu tiên là Transformation Based Learning, các tập luật mà thông qua thực nghiệm mà nhóm em tìm được cho kết quả tốt nhất là:

*Bảng 3: Các luật được sử dụng trong TBL.*

|  |  |
| --- | --- |
| **STT** | **Dạng luật** |
| 1 | Template(Word([0])) |
| 2 | Template(Word([2])) |
| 3 | Template(Word([-1]), Pos([-1])) |
| 4 | Template(Word([-2]), Pos([-2])) |
| 5 | Template(Word([-1]), Pos([0])) |
| 6 | Template(Word([0]), Pos([1])) |
| 7 | Template(Word([-2]), Pos([0])) |
| 8 | Template(Word([1])) |

Đối với thuật toán Perceptron, các siêu tham số được sử dụng là learning\_rate = 0.001, hàm kích hoạt là hàm relu có công thức là max(0, x), số vòng lặp là 500 vòng.

# CHƯƠNG 5: KẾT QUẢ

## 5.1. Các độ đo được sử dụng

### 5.1.1. Độ đo Accuracy

Độ đo accuracy là tỉ lệ giữa số điểm dữ liệu được dự đoán đúng và tổng số điểm dữ liệu trong tập dữ liệu. Độ chính xác giúp ta đánh giá hiệu quả dự báo của mô hình trên một bộ dữ liệu. Độ chính xác càng cao thì mô hình của chúng ta càng chuẩn xác. Công thức chung: Accuracy = . Hệ số accuracy cho bảng kết quả trên là (55+900)/1000 = 90.5%.

*Bảng 4: Ví dụ về các độ đo.*

|  |  |  |  |
| --- | --- | --- | --- |
| Dự đoán/Thực tế | | Thực tế | |
| BAD (Positive) | GOOD (Negative) |
| Dự đoán | BAD(Positive) | 55 (True Positive - TP) | 50 (False Positive - FP) |
| GOOD (Negative) | 45 (False Negative - FN) | 850 (True Negative - TN) |
| Tổng cộng |  | 100 | 900 |

### 5.1.2. Độ đo Precision

Độ đo Precision cho biết có bao nhiêu phần trăm trong các trường hợp được dự báo là positive là đúng. Công thức cho độ đo này là Precision = . Như vậy, độ đo precision cho bảng kết quả trên là 55/(50+55) = 52.4%.

### 5.1.3. Độ đo Recall

Recall đo lường tỷ lệ dự báo chính xác các trường hợp positive trên toàn bộ các mẫu thuộc nhóm positive. Công thức cho độ đo này là Recall = .

Như vậy, độ đo recall cho bảng kết quả trên là 55/(55+45) = 55%.

### 5.1.4. Độ đo F1-score

F1 Score là trung bình điều hòa giữa precision và recall. Do đó nó đại diện hơn trong việc đánh g**iá** độ chính xác trên đồng thời precision và recall.

Công thức của F1-score là:F1 = .

Như vậy, F1 của bảng kết quả trên là 2\*

## 5.2. Kết quả

Sau khi chạy thực nghiệm, nhóm em đã được kết quả sau trên bộ dữ liệu gold.

*Bảng 5: Kết quả của 3 thuật toán*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **Thuật toán** | **Accuracy** | **Precision** | **Recall** | **F1** |
| 1 | TBL | 0.8125 | 0.8095 | 0.7857 | 0.7809 |
| 2 | HMM | 0.9375 | 0.9523 | 0.9761 | 0.9584 |
| 3 | Perceptron | 0.9617 | 0.9242 | 0.9586 | 0.9245 |

## 5.3. Nhận xét

Thuật toán Transformation Based Learning cho kết quả thấp nhất giữa 3 thuật toán. Các độ đo chỉ ở mức 0.80 với Accuracy đạt 81.25%. Độ đo F1 đạt 78.09%. Đây là một kết quả đạt ở mức chấp nhận được nhưng kết quả này có thể cải thiện hơn nữa trong 2 thuật toán tiếp theo.

Ở thuật toán Hidden Markov Model, tỉ lệ chính xác của mô hình này đạt 93.75%, một con số khá cao. Nhưng 3 độ đo còn lại trên thuật toán này còn đạt cao hơn với Precision, Recall, F1 lần lượt là 95.23%, 97.61% và 95.84%. Như vậy, F1 của HMM là cao nhất giữa 3 thuật toán.

Mặc dù thuật toán Perceptron cho kết quả Accuracy cao nhất (96.17%) nhưng 3 độ đo còn lại trên thuật toán này lại không đạt tốt bằng thuật toán Hidden Markov Model. Độ đo recall đạt 95.86%, cao nhất giữa 3 độ đo nên F1 đạt 92.45%.

Một điều đặc biệt giữa các độ đo là Recall cho kết quả cao hơn Precision trên 2 thuật toán Hidden Markov Model và Perceptron. Điều đó cho thấy số lượng dự đoán đúng trên tổng số lượng thực tế là đúng là tốt hơn so với tỉ lệ trên độ đo Precision.

# CHƯƠNG 6: KẾT LUẬN

Qua đồ án môn học Xử lí Ngôn ngữ Tự nhiên này, nhóm em đã làm được nhiều phần trong bài toán gán nhãn từ loại cho Tiếng Việt. Từ bộ dữ liệu thu thập trên một bài báo ở trang báo điện tử Zingnews. Nhóm em đã thực hiện các bước xử lí như tách từ bằng WFST, gán nhãn thủ công qua công cụ VLSP.

Sau khi có được bộ dữ liệu hoàn chỉnh, nhóm em tiến hành thực nghiệm trên ba mô hình là Transformation Based Learning, Hidden Markov Model và Perceptron. Mô hình TBL cho kết quả thấp nhất giữa ba mô hình với Accuracy đạt 81.25%. Mô hình Hidden Markov Model dù cho độ đo Accuracy thấp hơn mô hình Perceptron nhưng cho F1, Recall và Perceptrong với F1 đạt 95.84%. Độ đo Accuracy cao nhất ở thuật toán Perceptron với kết quả 96.17%.

Hạn chế của đồ án này là bộ dữ liệu không thật sự lớn. Dữ liệu là một phần rất quan trọng, không thể thiếu trong các dự án về máy học. Chất lượng dữ liệu ảnh hưởng lớn đến chất lượng của mô hình nhưng vì trong phạm vi đồ án này nên nhóm em chỉ có thể xây dựng một bộ dữ liệu nhỏ để phục vụ cho bài toán. Khi phải giải quyết với dữ liệu chưa có trong bộ dữ liệu thì độ chính xác của mô hình sẽ không cao như bảng kết quả trên. Vì vậy, hướng phát triển của bài toán này có thể là mở rộng dữ liệu hơn nữa, xây dựng một bộ dữ liệu lớn hơn, bao quát đầy đủ trường hợp ngữ cảnh trong tiếng Việt.