

TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG



BÀI TẬP LỚN
NHẬP MÔN TRÍ TUỆ NHÂN TẠO

Đề tài: Phân loại văn bản bằng Machine Learning

GV hướng dẫn : ThS. Ngô Văn Linh

Danh sách thành viên nhóm 19:

STT	Họ và tên	MSSV
1	Nguyễn Quang Huy	20183554
2	Trần Quang Minh	20183594
3	Nguyễn Nam Hán	20183522
4	Vũ Tấn Khang	20183561

BÀI TẬP LỚN

NHẬP MÔN TRÍ TUỆ NHÂN TẠO

Đề tài: Phân loại văn bản bằng Machine Learning

GV hướng dẫn : ThS. Ngô Văn Linh

Nhóm sinh viên thực hiện : 19

STT	Họ và tên	MSSV
1	Nguyễn Quang Huy	20183554
2	Trần Quang Minh	20183594
3	Nguyễn Nam Hán	20183522
4	Vũ Tấn Khang	20183561

MỤC LỤC

MỤC LỤC.....	3
CHƯƠNG 1: GIỚI THIỆU BÀI TOÁN	4
1.1. Mô tả bài toán	4
1.2. Cách tiếp cận giải quyết bài toán.....	4
CHƯƠNG 2: CÁC PHƯƠNG PHÁP GIẢI QUYẾT BÀI TOÁN.....	6
2.1. Tiền xử lý văn bản.....	6
2.1.1. Chuẩn hóa bảng mã unicode.....	6
2.1.2. Chuẩn hóa kiểu gõ dấu tiếng Việt.....	6
2.1.3. Thực hiện tách từ tiếng Việt	6
2.1.4. Chuẩn hóa chữ viết thường.....	7
2.1.5. Xóa các ký tự, từ không cần thiết.....	7
2.2. Thuật toán phân loại văn bản với Naive Bayes	7
2.2.1. Cơ sở lý thuyết	7
2.2.2. Phương pháp thực hiện	8
2.3. Thuật toán phân loại văn bản với Neural Network.....	9
2.3.1. Cơ sở lý thuyết	9
2.3.2. Phương pháp thực hiện	9
2.3.3. Khó khăn và cách khắc phục	11
CHƯƠNG 3: KẾT QUẢ THỰC NGHIỆM, ĐÁNH GIÁ VÀ PHƯƠNG PHÁP CẢI TIẾN .	12
3.1. Kết quả thực nghiệm.....	12
3.2. Đánh giá, giải thích	14
3.3. Hệ thống đơn giản bằng Flask.....	14
3.4. Các phương pháp cải tiến trong tương lai	14
CHƯƠNG 4: CÁC THƯ VIỆN ĐÃ SỬ DỤNG.....	16
4.1. Numpy	16
4.2. Regex	16
4.3. Tensorflow	16
4.4. Pyvi.....	16
4.5. Flask.....	17
TÀI LIỆU THAM KHẢO.....	18

CHƯƠNG 1: GIỚI THIỆU BÀI TOÁN

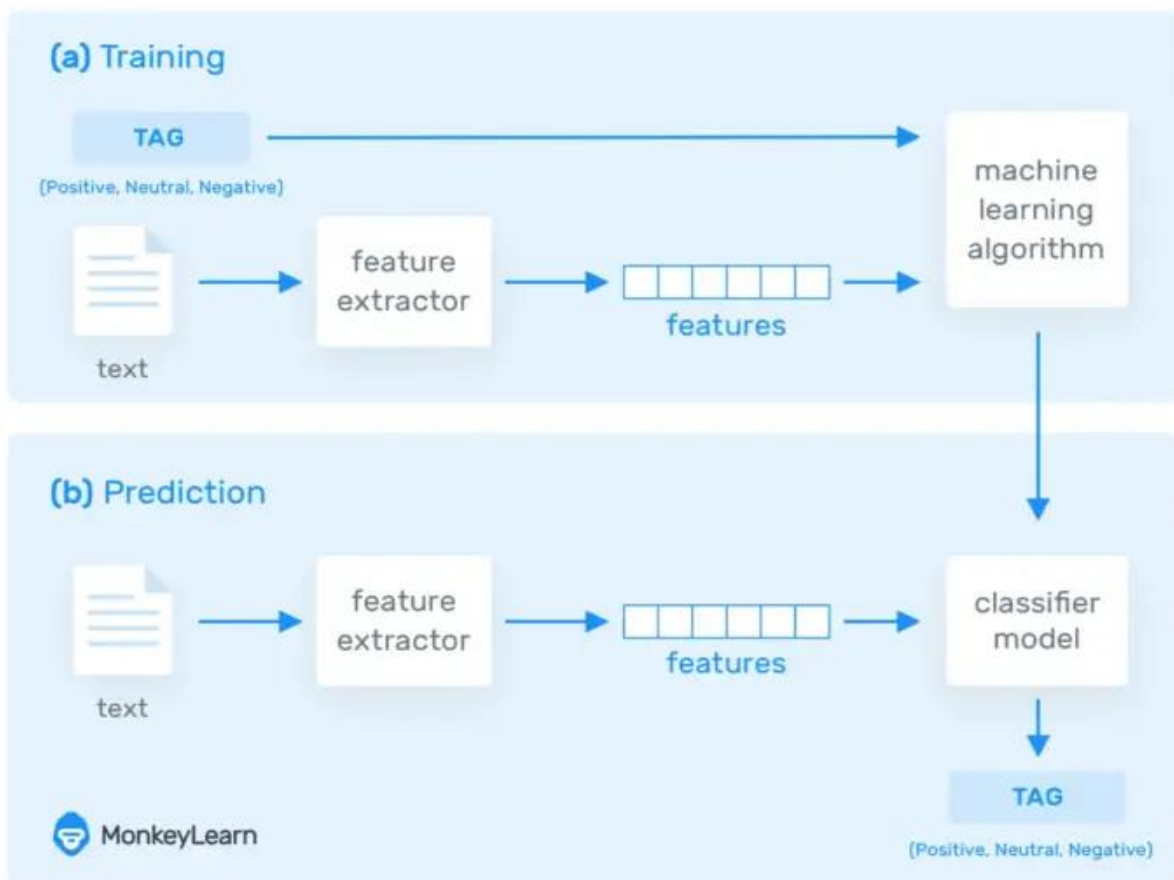
1.1. Mô tả bài toán

Phân loại văn bản là bài toán phổ biến trong xử lý ngôn ngữ tự nhiên. Đối với phân loại văn bản tiếng Việt, sẽ khó khăn hơn đôi chút so với phân loại văn bản tiếng Anh. Trong đề tài này nhóm sẽ đi giải quyết bài toán phân loại văn bản bằng tiếng Việt.

Ứng dụng của bài toán phân loại văn bản cũng khá phổ biến, chúng ta có thể sử dụng mô hình này để phân loại văn bản tin tức tiếng Việt cho một trang báo điện tử. Mỗi khi một bài báo được đăng, chương trình cần phải tự động xác định được bài báo đó nằm trong danh mục nào. Các danh mục phân loại bao gồm 10 danh mục : chính trị xã hội, công nghệ, đời sống, khoa học, kinh doanh, pháp luật, sức khỏe, thể giới, thể thao, văn hóa.

1.2. Cách tiếp cận giải quyết bài toán

Bài toán phân loại văn bản (text classification) là bài toán thuộc nhóm học có giám sát trong học máy. Vì thế, để giải quyết bài toán này nhóm quyết định sử dụng hai thuật toán học có giám sát như là Naive Bayes, Neural network.



Hình 1: Mô hình phân loại văn bản tiếng Việt tự động với Machine Learning

Để giải quyết bài toán học có giám sát nói chung và cụ thể trong đề tài này là bài toán phân loại văn bản bước đầu là phải thu thập dữ liệu có nhãn (có thể là kéo dữ liệu từ các website tin tức hoặc là lấy từ các dữ liệu đã được thu thập bởi cá nhân/tổ chức nào đó). Sau đó sẽ xây dựng một mô hình để có thể học từ những dữ liệu đã có để dự đoán cho các dữ liệu mới mà mô hình chưa gặp. Trong các thuật toán khác nhau việc xây dựng mô hình để học từ dữ liệu cũng khác nhau.

CHƯƠNG 2: CÁC PHƯƠNG PHÁP GIẢI QUYẾT BÀI TOÁN

2.1. Tiền xử lý văn bản

Văn bản trước khi đưa vào huấn luyện hay kiểm tra thì đều phải trải qua bước tiền xử lý cơ bản, đối với văn bản tiếng Việt các bước tiền xử lý bao gồm: chuẩn hóa bảng mã unicode, chuẩn hóa kiểu gõ dấu tiếng Việt, thực hiện tách từ tiếng Việt, chuẩn hóa chữ viết thường, xóa các ký tự, từ không cần thiết (stop word).

2.1.1. Chuẩn hóa bảng mã unicode

Có 2 loại bảng mã unicode được sử dụng phổ biến hiện nay là unicode tổ hợp và unicode dựng sẵn. Điều đó dẫn tới vấn đề, với cùng 1 chuỗi kí tự giống nhau như sau “tiếng việt bảo táp”, nếu là unicode dựng sẵn khi thể hiện dưới dạng chuỗi byte, dàn theo bảng mã UTF-8 thì chuỗi byte này là :

```
b'Ti\xel\xba\xbfng Vi\xel\xbb\x87t b\xcc3\xa3o t\xcc3\xa1p'
```

trong khi với unicode tổ hợp thì chuỗi byte này là :

```
b'Tie\xcc\x82\xcc\x81ng Vie\xcc\xa3\xcc\x82t ba\xcc\x83o ta\xcc\x81p'
```

do vậy khi mã hóa máy tính sẽ hiểu 2 chuỗi này là khác nhau. Vậy muốn so sánh hay muốn để máy tính hiểu được 2 chuỗi này là giống nhau thì ta phải đưa chúng về cùng 1 bảng mã. Trong đề tài này, các văn bản sẽ được đưa về cùng 1 loại bảng mã là unicode tổ hợp.

Việc chuyển đổi từ unicode dựng sẵn sang unicode tổ hợp cũng khá đơn giản. Trong thư viện chuẩn của python đã có module unicodedata cung cấp sẵn chức năng này.

2.1.2. Chuẩn hóa kiểu gõ dấu tiếng Việt

Tiếng Việt hiện nay có 2 kiểu gõ dấu khác nhau. Chúng có tên là kiểu mới (ví dụ : òa, úy) và kiểu cũ (ví dụ : òa, úy). Các bộ gõ tiếng Việt cũng cho phép tùy chỉnh một trong hai kiểu gõ trên dẫn tới các từ giống nhau nhưng lại có cách viết khác nhau. Do đó việc ta cần làm là đưa chúng về một bộ gõ tiêu chuẩn. Kiểu gõ chữ tiếng Việt cũ là phổ biến hơn, vì thế hầu hết các văn bản sẽ có kiểu gõ này, vậy nên chúng ta sẽ đưa hết các chữ về kiểu gõ cũ. Việc chuyển đổi này nhóm tham khảo tại cách làm tại <https://nguyenvanhieu.vn/phan-loai-van-ban-tieng-viet/>. Phiên bản chuẩn hóa kiểu gõ dấu tiếng Việt viết bằng ngôn ngữ python đã được tích hợp trong mã nguồn.

2.1.3. Thực hiện tách từ tiếng Việt

Đơn vị từ trong tiếng Việt bao gồm từ đơn và từ ghép. Vì thế ví dụ ta có 2 chữ cái “học sinh” nhưng trong tiếng Việt phải hiểu đây là một từ. Thế nên chúng ta phải có cơ

chế sao cho máy tính hiểu được đâu là từ đơn, đâu là từ ghép. Nếu không máy tính sẽ hiểu tất cả là từ đơn điều đó dẫn tới việc mã hóa sẽ bị nhầm lẫn và kết quả sẽ kém đi.

Bài toán tách từ cũng là một bài toán cơ sở trong xử lý ngôn ngữ tự nhiên. Do đề tài này đề cập tới việc phân loại văn bản, không đi sâu vào việc tách từ nên việc tách từ sẽ sử dụng thư viện mã nguồn mở pyvi (đối với ngôn ngữ python). Ngoài ra chúng ta cũng có thể sử dụng thư viện underthesea để phục vụ cho việc tách từ.

2.1.4. Chuẩn hóa chữ viết thường

Máy tính sẽ hiểu chữ viết hoa và chữ viết thường là khác nhau. Vì vậy, việc đưa về chữ viết thường giúp giảm số lượng đặc trưng và tăng độ chính xác hơn cho mô hình bởi vì đặc trưng chữ hoa/thường hoàn toàn không có tác dụng gì cho bài toán phân loại văn bản.

2.1.5. Xóa các ký tự, từ không cần thiết

Trong văn bản có rất nhiều từ lặp đi lặp lại rất nhiều lần mà không giúp ích gì cho việc phân loại văn bản như các từ : và, hoặc, là, thì, dấu ngắt câu,... chúng không mang bất cứ đặc trưng nào, có thể xuất hiện ở bất cứ văn bản loại nào, vì vậy việc xóa bỏ các từ này giúp cho mô hình giảm số lượng chiều đặc trưng, tăng tốc độ học và xử lý và tránh làm ảnh hưởng xấu tới kết quả của mô hình.

Để loại bỏ các ký tự không cần thiết ta sẽ liệt kê ra các từ mà ta cho rằng là không giúp ích gì cho việc phân loại văn bản, sau đó sẽ kiểm tra từng văn bản và xóa trong văn bản đó đi các từ này.

Tập các stop word, nhóm tham khảo và thu thập tại đường link https://xltiengviet.fandom.com/wiki/Danh_s%C3%A1ch_stop_word.

2.2. Thuật toán phân loại văn bản với Naive Bayes

2.2.1. Cơ sở lý thuyết

- Công thức xác suất có điều kiện :

Xác suất điều kiện của biến cố A với điều kiện biến cố B đã xảy ra là một số không âm, ký hiệu là $P(A|B)$ nó biểu thị khả năng xảy ra biến cố A trong tình huống biến cố B đã xảy ra.

$$P(A|B) = \frac{P(AB)}{P(B)}$$

suy ra :

$$P(A|B).P(B) = P(B|A).P(A) = P(AB)$$

- Công thức xác suất đầy đủ :

Giả sử B_1, B_2, \dots, B_n là 1 nhóm đầy đủ. Xét biến cố A , sao cho A chỉ xảy ra khi một trong các biến cố B_1, B_2, \dots, B_n xảy ra. Khi đó :

$$P(A) = \sum_{i=1}^n P(B_i).P(A|B_i)$$

- Công thức xác suất Bayes :

$$P(A|B) = \frac{P(B|A).P(A)}{P(B)}$$

- Phương pháp phân loại văn bản bằng Bayes :

Phân loại Bayes là phương pháp phân loại sử dụng tri thức các xác suất đã qua huấn luyện. Phương pháp này thích hợp với những lớp bài toán đòi hỏi phải dự đoán chính xác lớp của mẫu cần kiểm tra dựa trên những thông tin từ tập huấn luyện ban đầu

2.2.2. Phương pháp thực hiện

Ta chia chương trình ra thành nhiều module để thực hiện các công việc khác nhau. Sau mỗi bước kết quả sẽ được lưu ra 1 file text để làm đầu vào cho các bước sau.

Tập học D_{train} , trong đó mỗi ví dụ học là một biểu diễn văn bản gắn với một nhãn lớp: $D = \{(d_k, c_i)\}$. Một tập các nhãn lớp xác định: $C = \{c_i\}$.

Giai đoạn học:

- Từ tập các văn bản trong D_{train} , trích ra tập các từ khóa (keywords/terms): $T = \{t_j\}$
- Gọi $D_{c_i} (\subseteq D_{train})$ tập các văn bản trong D_{train} có nhãn lớp c_i
- Đối với mỗi phân lớp c_i :
 - o Tính giá trị xác suất trước của phân lớp c_i :

$$P(c_i) = \frac{|D_{c_i}|}{|D|}$$

- o Đối với mỗi từ khóa t_j , tính xác suất từ khóa t_j xuất hiện đối với lớp c_i :

$$P(t_j|c_i) = \frac{(\sum_{d_k \in D_{c_i}} n(d_k, t_j)) + 1}{(\sum_{d_k \in D_{c_i}} \sum_{t_m \in T} n(d_k, t_m)) + |T|}$$

Giai đoạn phân lớp:

- Từ văn bản d , trích ra tập T_d gồm các từ khóa (keywords) t_j đã được định nghĩa trong tập T ($T_d \subseteq T$)
- Giả sử: Xác suất từ khóa t_j xuất hiện đối với lớp c_i là độc lập đối với vị trí của từ đó trong văn bản

$$P(t_j \text{ ở vị trí } k|c_i) = P(t_j \text{ ở vị trí } m|c_i), \forall k, m$$

- Đối với mỗi phân lớp c_i , tính giá trị likelihood của văn bản d đối với c_i :

$$\log(P(c_i)) + \sum_{t_j \in T_d} \log(P(t_j|c_i))$$

(dùng log để tránh cho các giá trị quá nhỏ, làm giảm độ chính xác cho việc tính toán)

- Phân lớp văn bản d thuộc vào lớp c^* :

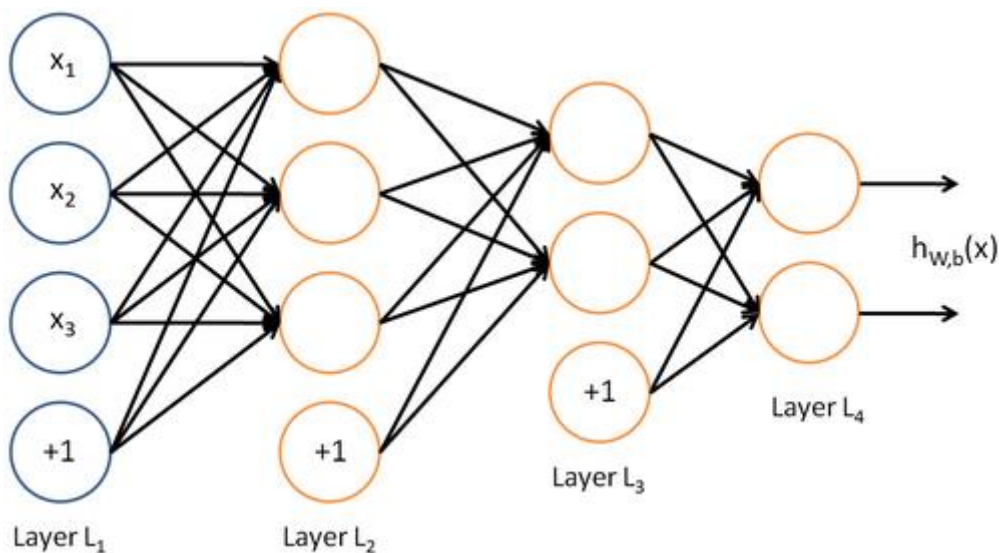
$$c^* = \operatorname{argmax} \left(\log(P(c_i)) + \sum_{t_j \in T_d} \log(P(t_j|c_i)) \right)$$

2.3. Thuật toán phân loại văn bản với Neural Network

2.3.1. Cơ sở lý thuyết

Mạng thần kinh nhân tạo (Neural network) là một mô hình phổ biến trong lĩnh vực Machine Learning, được áp dụng trong rất nhiều bài toán. Về cơ bản, mô hình này lấy ý tưởng dựa trên mạng neural sinh học trong bộ não con người. Nó gồm các neural nhân tạo (nút) nối với nhau, xử lý thông tin bằng cách truyền theo các kết nối và tính giá trị mới tại các nút. Trong mô hình mạng neural cơ bản, nó sẽ bao gồm ba phần:

- Tầng input (Input layer): là đầu vào của mô hình
- Tầng ẩn (Hidden layer): các tầng ẩn, thực hiện tính toán, truyền và lưu trữ thông tin từ tầng input tới tầng output. Có thể có một hoặc nhiều tầng ẩn
- Tầng output (Output layer): là đầu ra của mô hình.



Hình 2: Mô hình mạng Neural network cơ bản

Đối với bài toán phân loại văn bản sử dụng mạng neural, đầu vào của mô hình sẽ là một vector được mã hóa từ văn bản cần phân loại, đầu ra sẽ là một vector với số chiều là số nhãn của các văn bản trong bài toán, phần tử nào có giá trị lớn nhất trong vector sẽ là nhãn mà mô hình dự đoán cho văn bản đó. Mục tiêu của chúng ta là sẽ xây dựng mô hình với các tham số phù hợp để thực hiện dự đoán một cách chính xác nhất.

2.3.2. Phương pháp thực hiện

Từ cơ sở lý thuyết đã trình bày ở trên, chúng ta sẽ thực hiện bài toán theo các bước như sau:

- **Bước 1: Vector hóa văn bản :**

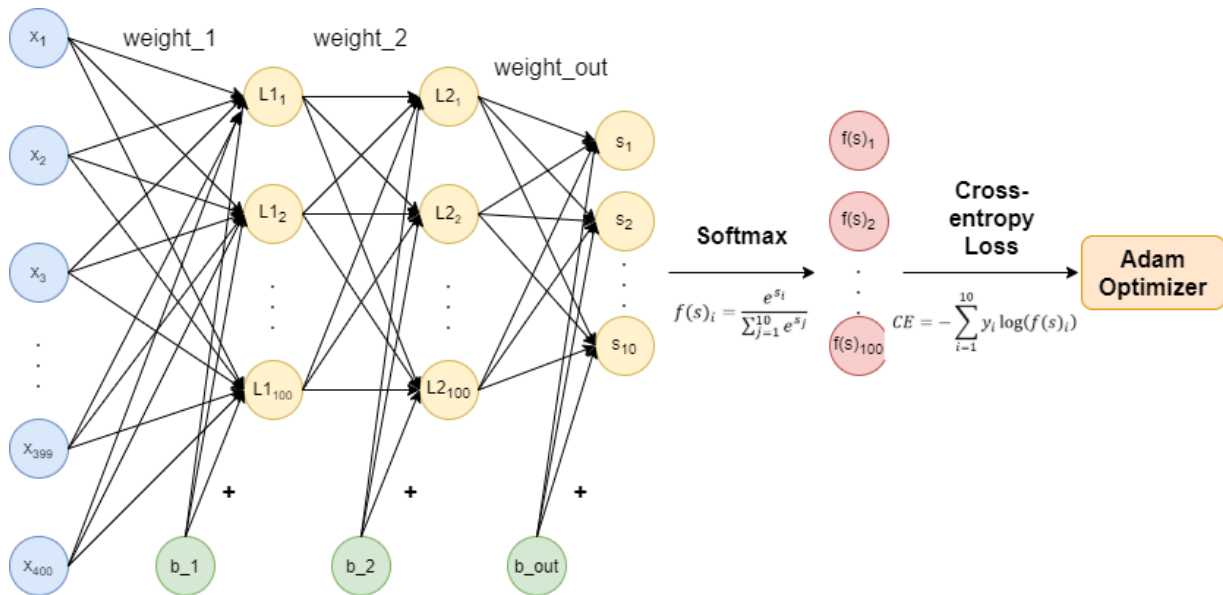
Việc mã hóa văn bản thành các vector từ tập từ điển là không khả thi do tập từ điển là quá lớn, không phù hợp với chi phí tính toán. Do đó để phù hợp với bài toán, chúng ta sẽ chọn ra 40 từ xuất hiện nhiều nhất ở mỗi nhãn (vì những từ này có khả năng mang tính đặc trưng cao nhất ở mỗi nhãn), tổng hợp lại thành một tập 400 từ phân biệt để mã hóa các văn bản thành vector có 400 chiều, giá trị của mỗi chiều tương ứng là số lần xuất hiện của các từ trong văn bản. Lưu kết quả mã hóa các văn bản ra một file text để chuẩn bị cho việc training.

- Bước 2: Xây dựng mạng neural network :

Có rất nhiều mô hình neural network, do đó chúng ta cần chọn mô hình phù hợp cho chi phí tính toán cũng như độ chính xác. Ở đây nhóm chọn mô hình mạng gồm tầng input có 400 nút, 2 tầng ẩn mỗi tầng có 100 nút, tầng output có 10 nút. Như vậy chúng ta sẽ cần phải tối ưu hàm chi phí để tìm các ma trận `weight_h1` (400 x 100), vector `bias_b1` (100 chiều) kết nối tầng input với tầng ẩn 1, ma trận `weight_h2` (100 x 100), vector `bias_b2` (100 chiều) kết nối tầng ẩn 1 với tầng ẩn 2, ma trận `weight_out` (100 x 10), vector `bias_out` (10 chiều) để kết quả dự đoán là chính xác nhất.

- Bước 3: Training :

Sau khi thực hiện xong hai bước 1 và 2, tập các vector đã mã hóa từ các văn bản trong tập training ta sẽ đưa vào mô hình để huấn luyện. Trước khi đưa vào hàm chi phí để tối ưu, vector output sẽ được tính toán một lần nữa bằng hàm softmax. Hàm chi phí và thuật toán tối ưu nhóm sử dụng ở đây là hàm cross-entropy và thuật toán Adam với tỉ số học là 0.1. Sau 1000 bước lặp, ta thu được các ma trận `weight_h1`, `weight_h2`, `weight_out` và các vector `bias_b1`, `bias_b2`, `bias_out`. Khi phân loại một văn bản mới, ta sẽ tiếp tục vector hóa văn bản này từ tập 400 từ, sử dụng kết quả từ bước training làm các tham số để tính ra được vector 10 chiều. Giá trị ở chiều nào lớn nhất sẽ là nhãn tương ứng với chiều đó mà mô hình dự đoán phân loại cho văn bản. Ở đây nhóm đã sử dụng thư viện tensorflow để hỗ trợ cho việc training.



Hình 3: Mô hình mạng Neural network được sử dụng

2.3.3. Khó khăn và cách khắc phục

Do có rất nhiều mô hình mạng neural, việc chọn mô hình phù hợp cho bài toán là khá khó khăn để khả thi cho việc tính toán cũng như để đảm bảo độ chính xác. Do đó, nhóm đã tiến hành thử nghiệm trên nhiều mô hình bằng cách thay đổi số tầng ẩn, thay đổi số nút trên tầng ẩn,... và kiểm tra độ chính xác trên tập test. Sau một số lần thử nghiệm, kết quả trên mô hình đã sử dụng (2 tầng ẩn, mỗi tầng 100 nút) đạt độ chính xác khá cao (xấp xỉ 81%) so với các mô hình trước đó (chỉ rơi vào khoảng 70-75%) và có thời gian training phù hợp nên nhóm đã quyết định sử dụng mô hình này vào bài toán.

CHƯƠNG 3: KẾT QUẢ THỰC NGHIỆM, ĐÁNH GIÁ VÀ PHƯƠNG PHÁP CẢI TIẾN

3.1. Kết quả thực nghiệm

Đây là thông kê dữ liệu được sử dụng trong mỗi nhãn (dữ liệu lấy tại đường link <https://github.com/duyvuleo/VNTC/>)

Mã lớp 1 - chinh_tri_xa_hoi - 12786

Mã lớp 2 - cong_nghe - 7041

Mã lớp 3 - doi_song - 5195

Mã lớp 4 - khoa_hoc - 3916

Mã lớp 5 - kinh_doanh - 7828

Mã lớp 6 - phap_luat - 7656

Mã lớp 7 - suc_khoe - 8801

Mã lớp 8 - the_gioi - 9614

Mã lớp 9 - the_thao - 11965

Mã lớp 10 - van_hoa - 9330

Dữ liệu được sử dụng có phần không đều, tuy nhiên lượng dữ liệu sử dụng là khá lớn nên có thể bù đắp được phần nào cho việc này.

Tập dữ liệu trên được chia làm 2 tập train và test với tỷ lệ 3 : 1. Tập train sẽ sử dụng để cho mô hình học, còn tập test sẽ giúp kiểm tra lại mô hình để đánh giá kết quả thu được từ mô hình đã xây dựng.

Sau khi loại bỏ stop word, các từ trong tập train có 155770 từ. Để đánh giá độ chính xác, sau khi cho mô hình học trong tập train, chúng ta tiếp tục cho thử nghiệm trên tập test để thu lại kết quả thử nghiệm như sau :

Bảng 1: Kết quả kiểm chứng bộ phân lớp bằng Naive Bayes

Mã lớp	1	2	3	4	5	6	7	8	9	10
1	2780	8	62	21	136	94	14	47	6	30
2	45	1608	12	11	58	3	2	11	1	10
3	22	11	902	45	6	7	52	10	2	243
4	70	29	56	729	23	0	32	25	2	14
5	99	43	3	3	1753	16	4	30	1	6
6	82	2	27	0	32	1745	3	4	14	6
7	98	5	24	43	20	6	1959	42	1	3
8	52	7	35	17	30	10	12	2198	2	42
9	39	3	10	6	4	71	6	24	2793	36
10	31	5	13	11	7	5	2	11	0	2249

Bảng 2: Kết quả kiểm chứng bộ phân lớp bằng Neural Network

Mã lớp	1	2	3	4	5	6	7	8	9	10
1	2470	70	71	37	171	171	46	63	29	70
2	38	1566	23	12	72	11	7	17	1	14
3	146	33	585	85	30	20	107	75	14	205
4	69	76	54	608	12	1	51	77	10	22
5	259	62	9	4	1529	30	5	35	5	2
6	169	6	17	0	33	1650	4	6	21	9
7	51	23	58	74	10	4	1937	35	5	4
8	50	22	9	18	35	11	18	2188	15	39
9	33	10	8	3	3	24	8	29	2857	17
10	63	15	65	7	8	13	6	38	19	2100

Công thức tính các chỉ số :

$$Precision = \frac{TruePositive}{TruePositive + FalsePositive}$$

$$Recall = \frac{TruePositive}{TruePositive + FalseNegative}$$

$$F1 = 2 * \frac{Precision * Recall}{Precision + Recall}$$

Bảng 3: So sánh hiệu quả phân loại văn bản giữa Naive Bayes và Neural Network

Mã lớp	Naive Bayes			Neural Network		
	Precision	Recall	F1	Precision	Recall	F1
1	0.84	0.87	0.85	0.74	0.77	0.75
2	0.93	0.91	0.92	0.83	0.89	0.86
3	0.79	0.69	0.74	0.65	0.45	0.53
4	0.82	0.74	0.78	0.72	0.62	0.67
5	0.85	0.9	0.87	0.8	0.79	0.8
6	0.89	0.91	0.9	0.85	0.86	0.86
7	0.94	0.89	0.91	0.88	0.88	0.88
8	0.92	0.91	0.91	0.85	0.91	0.88
9	0.99	0.93	0.96	0.96	0.95	0.96
10	0.85	0.96	0.9	0.85	0.9	0.87
Trung bình : 0.88				Trung bình : 0.81		

3.2. Đánh giá, giải thích

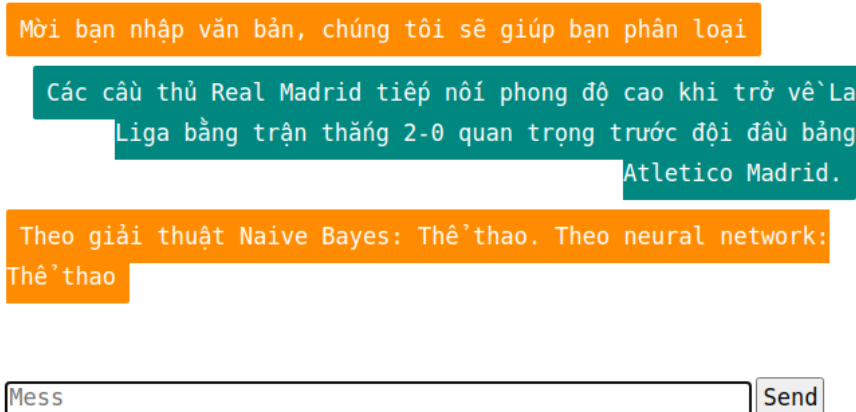
Từ bảng kết quả cũng như các chỉ số Precision, Recall và F1 có thể thấy phương pháp sử dụng thuật toán Naïve Bayes hiệu quả hơn so với phương pháp sử dụng mạng Neural network. Có thể lý giải điều này do phương pháp Naïve Bayes sử dụng tập hợp toàn bộ các từ trong các văn bản trong khi phương pháp sử dụng mạng Neural network chỉ sử dụng tập hợp gồm 400 từ. Do đó, khi vector hóa các văn bản trong mạng neural có thể không phản ánh được mức độ đặc trưng của văn bản đó đối với các thể loại.

Quan sát độ chính xác của từng nhãn có thể thấy các nhãn “Thể thao”, “Sức khỏe”, “Văn hóa” có độ chính xác rất cao. Điều này là do các văn bản này luôn có các từ có tính đặc trưng cao, phù hợp với thuật toán. Trong khi đó các nhãn “Đời sống”, “Khoa học” lại có độ chính xác chưa cao có thể do trong văn bản này có nhiều từ gây nhiễu, cũng có thể có nhiều từ trong các văn bản thuộc nhãn khác.

3.3. Hệ thống đơn giản bằng Flask

Trong đề tài này nhóm sẽ sử dụng công cụ Flask để thiết lập giao diện minh họa cho hệ thống phân loại văn bản.

Phân loại văn bản



Mời bạn nhập văn bản, chúng tôi sẽ giúp bạn phân loại

Các cầu thủ Real Madrid tiếp nối phong độ cao khi trở về La Liga bằng trận thắng 2-0 quan trọng trước đội đầu bảng Atletico Madrid.

Theo giải thuật Naive Bayes: Thể'thao. Theo neural network: Thể'thao

Mess Send

Hình 4: Hệ thống phân loại văn bản bằng Flask

Để sử dụng hệ thống phân loại văn bản, bạn chỉ cần copy văn bản của bạn vào ô mess sau đó ấn Enter, hệ thống sẽ tự động phân loại văn bản bằng 2 thuật toán.

3.4. Các phương pháp cải tiến trong tương lai

Đối với phương pháp phân loại bằng thuật toán Naïve Bayes, có thể thấy tập các từ trong văn bản sau bước tiền xử lý còn khá lớn và còn khá nhiều từ gây nhiễu. Do đó,

để cải thiện độ chính xác của thuật toán, ta có thể quan sát dữ liệu một cách kỹ hơn để loại thêm một số lượng các stop word làm giảm độ gây nhiễu cho thuật toán. Đồng thời cũng nhờ đó, độ phức tạp khi tính toán cũng giảm đi một chi phí đáng kể.

Đối với phương pháp sử dụng mạng Neural network, việc tập hợp 40 từ xuất hiện nhiều nhất ở mỗi nhãn văn có một số điểm hạn chế có thể do một số từ trong số này không mang độ đặc trưng nhiều cho các nhãn của văn bản. Do đó, ta có thể cải tiến phương pháp bằng cách quan sát dữ liệu và chọn ra các từ mang tính đặc trưng hơn cho văn bản (ví dụ nhãn “Thể thao” có thể là “cầu thủ”, “trận đấu”,...). Nhờ đó khi vector hóa cho các văn bản, các đặc trưng sẽ tập trung nhiều hơn vào các chiều tương ứng của vector và độ chính xác sẽ cao hơn.

CHƯƠNG 4: CÁC THƯ VIỆN ĐÃ SỬ DỤNG

4.1. Numpy

Numpy là một thư viện toán học phổ biến và mạnh mẽ của Python, cho phép làm việc hiệu quả với ma trận và mảng, đặc biệt là dữ liệu ma trận và mảng lớn với tốc độ xử lý nhanh hơn nhiều lần khi chỉ dùng “core Python” đơn thuần.

Để thực hiện đề tài này, nhóm đã sử dụng Numpy phục vụ cho việc vector hóa các văn bản để đưa vào các công thức, mô hình tính toán một cách thuận tiện và dễ dàng. Cú pháp cài đặt thư viện này như sau (trong command prompt):

```
pip install numpy
```

4.2. Regex

Regular expression (dịch ra tiếng Việt là biểu thức chính quy) là một đoạn các ký tự đặc biệt dùng để so khớp các chuỗi hoặc một tập các chuỗi. Cũng có thể gọi Regular expression là một ngôn ngữ. Và hầu như ngôn ngữ lập trình nào cũng hỗ trợ Regular expression, nhưng được sử dụng nhiều nhất và phổ biến nhất là trong UNIX.

Regular expression trong Python được thể hiện qua module `re`. Trong đề tài, nhóm đã sử dụng thư viện này cho việc chuẩn hóa dấu câu của các từ. Để cài đặt thư viện này, ta dùng cú pháp sau:

```
pip install regex
```

4.3. Tensorflow

Tensorflow hiện tại chính là thư viện mã nguồn mở cho Machine Learning nổi tiếng nhất thế giới, được phát triển bởi các nhà nghiên cứu từ Google. Việc hỗ trợ mạnh mẽ các phép toán học để tính toán trong Machine Learning và Deep Learning đã giúp việc tiếp cận các bài toán trở nên đơn giản, nhanh chóng và tiện lợi hơn nhiều.

Nhóm đã sử dụng tensorflow để phục vụ cho việc xây dựng mạng neural network, dùng hàm chi phí và thuật toán tối ưu được trang bị sẵn trong thư viện này. Nhờ đó việc xây dựng nên model và training trở nên đơn giản và thuận tiện.

Cú pháp cài đặt tensorflow:

```
pip install tensorflow
```

4.4. Pyvi

Pyvi là một thư viện xử lý ngôn ngữ tự nhiên với tiếng Việt rất phổ biến bên cạnh `underthesea`. Đây chính là một sản phẩm của TS. Trần Việt Trung, hiện đang công tác

tại trường Đại học Bách Khoa Hà Nội. Thư viện có khá nhiều tiện ích, trong đó nhóm đã sử dụng chức năng tách từ của thư viện để phục vụ cho bài toán.

Để cài đặt pyvi, thực hiện câu lệnh sau:

```
pip install pyvi
```

4.5. Flask

Flask là một Web Framework rất nhẹ của Python, dễ dàng giúp người mới bắt đầu học Python có thể tạo ra website nhỏ. Flask cũng dễ mở rộng để xây dựng các ứng dụng web phức tạp.

Trong đề tài này, nhóm sẽ sử dụng Flask để thiết kế giao diện minh họa cho các mô hình phân loại văn bản đã xây dựng.

Cú pháp cài đặt flask:

```
pip install flask
```

TÀI LIỆU THAM KHẢO

1. Bài giảng Nhập môn Trí tuệ nhân tạo – ThS. Ngô Văn Linh
2. Mạng thần kinh nhân tạo – Wikipedia tiếng Việt
3. <https://codelearn.io/sharing/tim-hieu-thu-vien-numpy-trong-python>
4. <https://topdev.vn/blog/tensorflow-la-gi/>
5. <https://github.com/trungtv/pyvi>
6. <https://toidicode.com/regular-expression-trong-python-365.html>