BỘ GIÁO DỤC VÀ ĐÀO TẠO
ĐẠI HỌC KINH TẾ THÀNH PHỐ HỒ CHÍ MINH (UEH)
TRƯỜNG CÔNG NGHỆ VÀ THIẾT KẾ

ĐỒ ÁN MÔN HỌC

ĐỀ TÀI:

ỨNG DỤNG PHƯƠNG PHÁP HỌC MÁY TRONG DỰ ĐOÁN NHÃN CHO VĂN BẢN TIẾNG VIỆT

Học phần: Xử Lý Ngôn Ngữ Tự Nhiên

Nhóm Sinh Viên:

1. Trương Thị Hồng Mai
2. Võ Minh Nguyên
3. Hoàng Đức Dân
4. Nguyễn Thanh Vy

Chuyên Ngành: Khoa Học Dữ Liệu

Khóa: K47

Giảng Viên Hướng Dẫn: TS. Đặng Ngọc Hoàng Thành

Thành phố Hồ Chí Minh , tháng 12 năm 2023

LỜI MỞ ĐẦU

Trong thế giới số hóa ngày nay, lượng dữ liệu văn bản tiếng Việt tăng lên một cách đáng kể qua từng ngày. Điều này tạo ra một nhu cầu khẩn cấp về việc phân loại và dự đoán nhãn cho văn bản. Đề tài "**Ứng dụng phương pháp học máy trong dự đoán nhãn cho văn bản tiếng Việt**" được thiết kế để đáp ứng nhu cầu này.

Học máy, một nhánh quan trọng của trí tuệ nhân tạo, đã được chứng minh là công cụ hiệu quả trong việc phân loại và dự đoán nhãn cho văn bản. Với khả năng học từ dữ liệu và tự cải thiện qua thời gian, học máy có thể giúp chúng ta xử lý lượng lớn dữ liệu văn bản một cách hiệu quả.

Ứng dụng của việc dự đoán nhãn cho văn bản tiếng Việt rất đa dạng, từ việc phân loại tin tức, đánh giá sản phẩm, phân loại ý kiến, đến việc giám sát nội dung không phù hợp trên các nền tảng trực tuyến. Điều này không chỉ giúp cải thiện hiệu suất của các hệ thống tự động, mà còn góp phần vào việc tạo ra một môi trường trực tuyến lành mạnh và an toàn hơn.

Tính cấp thiết

Tuy nhiên, việc xử lý văn bản tiếng Việt có những thách thức riêng biệt do đặc điểm ngôn ngữ và văn hóa. Chính vì vậy, việc nghiên cứu và phát triển các phương pháp học máy phù hợp cho tiếng Việt là cần thiết.

Với đề tài này, nhóm nghiên cứu sẽ tập trung vào việc khám phá và áp dụng các phương pháp học máy đã được tiếp cận trong học phần "**Xử lý ngôn ngữ tự nhiên**" (được Tiến sĩ Đặng Ngọc Hoàng Thành hướng dẫn), để dự đoán nhãn cho văn bản tiếng Việt. Mục tiêu là tạo ra một hệ thống có độ chính xác cao, có thể hoạt động ổn định trong thực tế, và có thể mở rộng để xử lý lượng dữ liệu lớn.

Đề tài này không chỉ đóng góp vào lĩnh vực nghiên cứu học máy và xử lý ngôn ngữ tự nhiên, mà còn có tác động tích cực đến cộng đồng, bằng cách cung cấp công cụ để giải quyết các vấn đề thực tế liên quan đến dữ liệu văn bản tiếng Việt. Nhóm nghiên cứu hy vọng rằng, với sự phát triển của đề tài này, xã hội sẽ tiến gần hơn đến mục tiêu của một thế giới số hóa thông minh và hiệu quả hơn.

Từ khóa: Số hóa, Văn bản tiếng Việt, Dự đoán nhãn, Phương pháp học máy, Xử lý ngôn ngữ tự nhiên.

MỤC LỤC

DANH MỤC HÌNH ẢNH

DANH MỤC BẢNG BIỂU

CHƯƠNG I. GIỚI THIỆU ĐỀ TÀI

1. Lý do lựa chọn đề tài

Học máy và xử lý ngôn ngữ tự nhiên đã trở thành hai trong số những lĩnh vực nghiên cứu nổi bật nhất trong thập kỷ qua, với sự tiến bộ vượt bậc trong cả hai lĩnh vực. Các phương pháp học máy đã được áp dụng rộng rãi trong nhiều lĩnh vực khác nhau, từ y tế đến tài chính …, và đã cho thấy kết quả đáng kinh ngạc.

Trong bối cảnh này, việc áp dụng học máy vào dự đoán nhãn cho văn bản tiếng Việt đang mở ra một lĩnh vực nghiên cứu mới và hứa hẹn. Tiếng Việt, với cấu trúc ngữ pháp phức tạp và từ vựng phong phú, đặt ra nhiều thách thức cho việc phát triển các mô hình học máy. Tuy nhiên, những thách thức này cũng tạo ra cơ hội để khám phá và phát triển các phương pháp học máy mới và hiệu quả.

Đề tài "**Ứng dụng phương pháp học máy trong dự đoán nhãn cho văn bản tiếng Việt**" được chọn nhằm khám phá những tiềm năng đó. Bằng cách tập trung vào việc áp dụng học máy để dự đoán nhãn cho văn bản tiếng Việt, nhóm nghiên cứu không chỉ tìm hiểu cách thức hoạt động của các hệ thống xử lý ngôn ngữ tự nhiên hiện tại, mà còn mở rộng hiểu biết của nhóm về cách học máy có thể được áp dụng để giải quyết các vấn đề phức tạp trong xử lý ngôn ngữ tự nhiên.

2. Mục tiêu nghiên cứu

Nghiên cứu các phương phương pháp học máy trong việc xây dựng hệ thống dự đoán nhãn cho văn bản tiếng Việt với độ chính xác cao và khả năng ổn định khi ứng dụng thực tế. Đặc biệt, nhóm hướng tới việc đề xuất và tích hợp hệ thống này vào UEH Smart Digital Library của Đại học UEH. Việc này không chỉ mở rộng ứng dụng của học máy trong việc tối ưu hóa quản lý và truy cập thông tin trong thư viện mà còn giúp cải thiện trải nghiệm người dùng.

3. Phương pháp nghiên cứu

Phương pháp nghiên cứu chính của đề tài bao gồm những nội dung chính sau:

- ***Tiền xử lý cho những tập tin tiếng Việt***

  Tiền xử lý là bước quan trọng nhằm chuẩn hóa dữ liệu và loại bỏ những thông tin không cần thiết. Các công việc tiền xử lý có thể bao gồm việc loại bỏ các ký tự không phải chữ, chuyển đổi tất cả văn bản thành chữ thường, tách từ, và loại bỏ các từ không mang thông tin (stop words).

- ***Ứng dụng các mô hình máy học***

Các mô hình học máy như SVM (Support Vector Machines), Naive Bayes, và Mạng nơ-ron sẽ được áp dụng để huấn luyện trên tập dữ liệu đã được tiền xử lý. Ngoài ra, mô hình Maxent (Maximum Entropy) và mô hình học sâu LSTM (Long Short-Term Memory) cũng sẽ được sử dụng trong nghiên cứu này.

- ***So sánh, tìm kiếm mô hình nào hiệu quả nhất cho dự đoán nhãn***

  Sau khi huấn luyện, hiệu suất của các mô hình sẽ được đánh giá và so sánh dựa trên các chỉ số như độ chính xác (Accuracy), Recall, Precision, F1-score, và ma trận nhầm lẫn (Confusion Matrix). Mô hình nào cho kết quả tốt nhất sẽ được chọn để dự đoán nhãn cho văn bản tiếng Việt.

- ***Điều chỉnh các siêu tham số để cải thiện khả năng dự đoán***

  Để cải thiện hiệu suất của mô hình, việc điều chỉnh các tham số của mô hình (như Learning Rate, số lượng Epoch, kích thước Batch,…) sẽ được thực hiện. Quá trình này còn được gọi là tinh chỉnh mô hình (Model Fine-Tuning).

4. Tài nguyên sử dụng

- Ngôn ngữ lập trình: Python
- Xây dựng giao diện: Qt Designer
- Dữ liệu nghiên cứu: News Dataset Vietnameses

5. Cấu trúc bài nghiên cứu

- **Chương I:** Giới thiệu sơ lược về đề tài.
- **Chương II:** Xác định bài toán cụ thể trong lĩnh vực Xử lý ngôn ngữ tự nhiên mà nhóm lựa chọn nghiên cứu và thực hiện.
- **Chương III:** Các lý thuyết về các mô hình liên quan, bao gồm mô hình học máy như Support Vector Machine, Naive Bayes, Decision Tree; mô hình ensemble như Random Forest; mô hình MaxEnt; mô hình học sâu như Long short-term memory (LSTM).
- **Chương IV:** Quá trình thực hiện trên bộ dữ liệu sử dụng, bao gồm tiền xử lý dữ liệu, huấn luyện mô hình ban đầu, tinh chỉnh các siêu tham số và so sánh kết quả của các mô hình sử dụng.
- **Chương V:** Trình bày về giao diện xây dựng để ứng dụng các mô hình với siêu tham số tốt nhất được chọn lựa để phục vụ cho bài toán phân lớp văn bản.

- **Chương VI:** Kết luận về các kết quả mà báo cáo đã thu hoạch được, đồng thời đưa ra đề xuất để cải thiện và điều chỉnh phù hợp hơn cho các trường hợp ứng dụng khác.

CHƯƠNG II. XÁC ĐỊNH BÀI TOÁN

Bài toán phân loại văn bản (Text Classification) là một trong những nhiệm vụ cơ bản và phổ biến trong lĩnh vực xử lý ngôn ngữ tự nhiên (NLP). Mục tiêu của bài toán này là gán một hoặc nhiều nhãn (label) cho một đoạn văn bản dựa trên nội dung. Mục tiêu của một hệ thống phân loại văn bản là nó có thể tự động phân loại một văn bản cho trước, để xác định xem văn bản đó thuộc thể loại nào. Một số ứng dụng của hệ thống phân loại như:

- Hiểu được ý nghĩa, đánh giá, bình luận của người dùng từ mạng xã hội.
- Phân loại email là spam hay không spam.
- Tự động gắn thẻ cho những truy vấn, tìm kiếm của người dùng.
- Phân loại các bài báo điện tử.

Bài toán phân loại văn bản là một bài toán học có giám sát (Supervised Learning), khi chúng ta cần thực hiện huấn luyện mô hình trên tập dữ liệu đã được gán nhãn và tiến hành gán nhãn tự động đối với tập dữ liệu mới. Trong bài báo cáo này, nhóm quyết định sẽ thực hiện phân loại nhãn của các thể loại báo điện tử được lấy từ bộ dữ liệu tiếng việt của trang báo điện tử laodong.vn.

Có 4 bước để giải quyết bài toán Text Classification:

- Chuẩn bị dữ liệu (Dataset Preparation)
- Xử lý thuộc tính của dữ liệu (Feature Engineering)
- Xây dựng mô hình (Build Model)
- Tinh chỉnh mô hình và cải thiện hiệu năng (Improve Performance)

1. Chuẩn bị dữ liệu

- *Bước 1*: Xác định nội dung để thực hiện gán nhãn (content, title, summary,...) và các nhãn có sẵn cho việc phân lớp (các category có thể bao gồm: thể thao, sức khỏe, pháp luật,...).
- *Bước 2:* Sử dụng các thư viện để tách từ (Underthesea, Pyvi để sử dụng cho tiếng Việt). Cần lưu ý đến việc loại bỏ các dấu câu và các ký tự đặc biệt cũng như các chữ cái phải là chữ thường.
- *Bước 3:* Xử lý nhãn với Label encoder hoặc One-hot encoder.

2. Xử lý thuộc tính của dữ liệu

Cần chuyển đổi dữ liệu văn bản về dạng vector số học để thuận tiện cho việc huấn luyện mô hình. Có rất nhiều phương pháp để chuyển đổi dữ liệu từ dạng text sang dạng số như:

- Count Vectors as features
- TF-IDF Vectors as features
- Word level
- N-Gram level
- Character level
- Word Embeddings as features
- Text / NLP based features

3. Xây dựng mô hình

Chia tập train, validation, test để áp dụng cho mô hình phân loại nhãn. Sử dụng các mô hình máy học như (SVM, Naive Bayes,...) hoặc các mô hình học sâu (CNN, LSTM, BiLSTM, Pho-Bert,...) để tiến hành huấn luyện.

4. Tinh chỉnh mô hình và cải thiện hiệu năng

Ở bước này, chúng ta cần tinh chỉnh các tham số của mô hình bằng các phương pháp GridSearch, HyperParameters, Fine-tuning để đánh giá kết quả huấn luyện trên tập validation. Sau đó xem xét và phân tích các chỉ số đánh giá (accuracy, train error, test error, f1-score,...) để có thể lựa chọn ra mô hình tốt nhất và phù hợp với yêu cầu bài toán.

CHƯƠNG III. LÝ THUYẾT

1. Mô hình học máy

a. Giới thiệu SVM

Mô hình học máy SVM (Support Vector Machine) được phát triển tại AT&T Bell Laboratories bởi Vladimir Vapnik cùng với các đồng nghiệp. SVM là một trong những mô hình được nghiên cứu nhiều nhất, dựa trên các khung học thống kê hoặc lý thuyết VC được đề xuất bởi Vapnik (1982, 1995) và Chervonenkis (1974)1. Đây là một mô hình phân loại phân biệt được định nghĩa chính thức bởi một siêu phẳng phân cách. Nói cách khác, với dữ liệu huấn luyện đã được gán nhãn (học có giám sát), thuật toán xuất ra một siêu phẳng tối ưu phân loại các ví dụ mới.

Cơ chế hoạt động:

SVM hoạt động dựa trên việc tìm một siêu phẳng (hyperplane) trong không gian đặc trưng, có khả năng tách hai lớp dữ liệu khác nhau sao cho khoảng cách giữa các điểm dữ liệu gần nhất của hai lớp là lớn nhất. Từ đó, SVM có thể được sử dụng để phân loại các điểm dữ liệu mới vào các lớp tương ứng.

b. Khái niệm liên quan

Để hiểu cách SVM hoạt động, chúng ta cần biết về các khái niệm và công thức liên quan:

- **Siêu phẳng (Hyperplane)**

  Trong không gian đặc trưng n chiều, siêu phẳng là một đường thẳng hoặc mặt phẳng có số chiều bằng n-1. Siêu phẳng được định nghĩa bởi phương trình toán học.

  $$w^T \times x + b = 0$$

  Trong đó:

  - **w** là vector hệ số của siêu phẳng.
  - **x** là vector đặc trưng của một điểm dữ liệu.
  - **b** là điều chỉnh độ lệch của siêu phẳng.

- **Đường biên (Margin)**

Đường biên là khoảng cách từ siêu phẳng tới các điểm dữ liệu gần nhất của hai lớp. Khoảng cách này được đo bằng đường vuông góc từ một điểm trên siêu phẳng tới điểm dữ liệu gần nhất. Đường biên tối ưu là khoảng cách lớn nhất giữa các điểm dữ liệu gần nhất của hai lớp.

- **Vector hỗ trợ (Support Vector)**

Vector hỗ trợ là các điểm dữ liệu nằm trên đường biên hoặc gần đường biên. Các điểm này quan trọng trong việc xác định siêu phẳng tối ưu và không bị ảnh hưởng bởi các điểm dữ liệu khác.

- **Hàm mất mát (Loss Function)**

Hàm mất mát trong SVM được gọi là hàm mất mát Hinge. Hàm mất mát này đo lường vi phạm của các điểm dữ liệu đến siêu phẳng. Công thức của hàm mất mát Hinge là:

$$L(w) = C \times \Sigma[max(0, 1 - y_i \times (w^T \times x_i + b))] + \frac{1}{2} \times ||w||^2$$

Trong đó:

- **C** là tham số điều chỉnh đánh đổi giữa việc tối đa hóa đường biên và vi phạm điểm dữ liệu.
- y_i là nhãn của điểm dữ liệu x_i (+1 hoặc -1).
- **w** là vector hệ số của siêu phẳng.
- **b** là điều chỉnh độ lệch.
- ||**w**|| là độ dài của vector **w**.

- **Tối ưu hóa**

Mục tiêu là tìm vector hệ số w và b sao cho hàm mất mát L(w) là nhỏ nhất. Để làm điều này, chúng ta có thể sử dụng các phương pháp tối ưu hóa như phương pháp Gradient Descent hoặc phương pháp Lagrange.

Một số định nghĩa mở rộng khác:

- **Kernel Trick**

Trong trường hợp dữ liệu không thể phân tách tuyến tính, SVM sử dụng một kỹ thuật gọi là "kernel trick". Kỹ thuật này ánh xạ dữ liệu vào một không gian chiều cao hơn nơi dữ liệu có thể được phân tách tuyến

tính. Các hàm kernel phổ biến bao gồm hàm tuyến tính, đa thức, RBF (Radial Basis Function) và Sigmoid.

- **Soft Margin**

Trong thực tế, dữ liệu thường chứa nhiễu và không thể phân tách hoàn toàn. Do đó, SVM cung cấp một "soft margin" cho phép một số điểm dữ liệu vi phạm đường biên. Tham số C trong hàm mất mát Hinge kiểm soát mức độ linh hoạt của đường biên. Giá trị C lớn tạo ra đường biên cứng hơn (ít vi phạm hơn nhưng có thể gây ra hiện tượng Overfitting), trong khi giá trị C nhỏ tạo ra đường biên mềm hơn (nhiều vi phạm hơn nhưng có thể giảm hiện tượng Overfitting).

- **Multi-class Classification**

Mô hình SVM thông thường chỉ được áp dụng cho các bài toán phân loại nhị phân để phân loại các điểm dữ liệu thành 2 lớp riêng biệt. Đối với bài toán phân loại đa lớp (multi-class classification), SVM sẽ sử dụng 2 cách tiếp cận đó là One-versus-One và One-versus-Rest.

- **One-versus-One**: mỗi lớp sẽ được so sánh với mỗi lớp khác nhau để tạo ra một số lượng các bài toán phân loại nhị phân riêng biệt.
- **One-versus-Rest**: mỗi lớp sẽ được so sánh với tất cả các lớp còn lại để tạo ra N bài toán phân loại nhị phân.

One-Versus-One tạo ra nhiều mô hình hơn, nhưng mỗi mô hình chỉ đào tạo trên một tập hợp nhỏ dữ liệu, điều này có thể giúp tăng hiệu suất trong một số trường hợp. Ngược lại, One-Versus-Rest đào tạo ít mô hình hơn, nhưng mỗi mô hình đào tạo trên một lớp lớn hơn, điều này có thể hữu ích khi dữ liệu lớn và có nhiều lớp.

2. Kỹ thuật Ensemble Learning

a. Giới thiệu về Random Forest

Random Forest là một kỹ thuật Ensemble Learning phổ biến trong lĩnh vực học máy. Được giới thiệu bởi Leo Breiman vào năm 2001, Random Forest kết hợp các cây quyết định (decision trees) để tạo ra một mô hình mạnh mẽ và ổn định.

Cơ chế hoạt động:

Random Forest hoạt động bằng cách xây dựng nhiều cây quyết định tại thời gian huấn luyện. Đối với bài toán phân loại, kết quả của Random Forest

là lớp được chọn bởi hầu hết các cây. Đối với bài toán hồi quy, giá trị trung bình hoặc trung vị của các dự đoán từ các cây riêng lẻ sẽ được trả về. Random Forest giúp khắc phục vấn đề của cây quyết định thường xuyên Overfitting với tập dữ liệu huấn luyện.

Mục tiêu của Random Forest là tạo ra một mô hình dự đoán chính xác và ổn định bằng cách kết hợp dự đoán từ nhiều cây quyết định. Các cây quyết định được xây dựng độc lập và mỗi cây đóng góp vào quá trình dự đoán cuối cùng của Random Forest.

b. Khái niệm liên quan

- **Cây quyết định (Decision Tree):**

  Cây quyết định là một cấu trúc dữ liệu dạng cây có cấu trúc phân cấp, trong đó mỗi nút đại diện cho một quyết định hoặc một thuộc tính. Cây quyết định được sử dụng để tạo ra quyết định dựa trên các quy tắc và điều kiện trong dữ liệu.

- **Ensemble Learning:**

  Ensemble Learning là kỹ thuật kết hợp dự đoán từ nhiều mô hình học máy đơn lẻ để tạo ra một mô hình mạnh mẽ hơn. Random Forest là một phương pháp Ensemble Learning phổ biến, trong đó các cây quyết định được kết hợp lại.

- **Bootstrap Aggregating (Bagging):**

  Bagging là một phương pháp trong Ensemble Learning, trong đó các mô hình đơn lẻ được huấn luyện trên các tập con dữ liệu được chọn ngẫu nhiên từ tập dữ liệu huấn luyện gốc. Random Forest sử dụng Bagging để tạo ra các tập con dữ liệu cho từng cây quyết định.

- **Feature Subspacing:**

  Feature Subspacing là một phương pháp trong Random Forest, trong đó chỉ một số lượng ngẫu nhiên các đặc trưng được chọn để xây dựng mỗi cây quyết định. Điều này giúp đảm bảo tính đa dạng và độc lập giữa các cây quyết định.

- **Độ đo đánh giá độ tương tự giữa các điểm dữ liệu (Similarity Measure):**

  Độ đo đánh giá độ tương tự giữa các điểm dữ liệu được sử dụng để xác định sự tương đồng giữa các điểm trong quá trình xây dựng cây quyết

định và đưa ra các quyết định dự đoán. Phổ biến nhất là độ đo Gini impurity và entropy.

Gini impurity là một độ đo được sử dụng để đánh giá độ tương tự giữa các điểm dữ liệu trong quá trình xây dựng cây quyết định.

$$Gini(p) = 1 - \Sigma(p_i)^2$$

Trong đó:

- **p** là vectơ xác suất của các lớp tại nút đó.
- $\Sigma(p_i)^2$ là tổng bình phương xác suất của các lớp.

- **Voting:**

Trong Random Forest, quyết định cuối cùng được đưa ra bằng cách áp dụng phương pháp voting. Các cây quyết định đóng góp vào quá trình voting và kết quả dự đoán cuối cùng được xác định dựa trên đa số phiếu.

- **Out-of-Bag (OOB) Error:**

OOB Error là một phương pháp đánh giá hiệu suất của Random Forest. Trong quá trình Bagging, một số điểm dữ liệu không được chọn vào tập con dữ liệu huấn luyện cho một số cây. OOB Error được tính bằng cách đánh giá hiệu suất của mô hình trên các điểm dữ liệu không được chọn và không thuộc tập huấn luyện. OOB Error có thể được sử dụng để đánh giá hiệu suất và điều chỉnh các siêu tham số của Random Forest.

3. Maxent Classifier

a. Giới thiệu Maximum Entropy

Maximum Entropy là một bộ khung tối ưu hóa với mục tiêu tìm ra mô hình xác suất có thể độ hỗn loạn cao nhất trong các mô hình có thể quan sát được.

Theo Jaynes (1957), lý do chọn mô hình có độ hỗn loạn cao nhất là vì các mô hình khác đưa ra chứng cứ chưa được quan sát. Các chứng cứ này tượng trưng cho số lần xuất hiện chung của giá trị dự đoán và bối cảnh của dự đoán trong một corpus. Ví dụ, tần số đồng xảy ra giữa từ that và tag determiner là một chứng cứ được quan sát. Nếu giá trị ước tính của tần số đồng xảy ra với tần số đã quan sát trong corpus thì ta nói mô hình xác suất nhất quán với chứng cứ đã quan sát.

Khái niệm về độ hỗn loạn cao nhất đã được nhắc đến từ rất lâu trong lịch sử. Theo Laplace trong "Principle of Insufficient Reason", "khi ta không sở hữu thông tin để phân biệt xác suất của hai sự kiện, lựa chọn tốt nhất là hãy xem chúng có khả năng xảy ra như nhau." Jaynes (1990), một trong những nhà tiên phong trong lĩnh vực này, đã khẳng định: "sự thật rằng có một phân phối xác suất tối đa độ hỗn loạn với ràng buộc tượng trưng cho thông tin chưa hoàn thiện, chính là đặc điểm nền tảng giải thích cho việc ta dùng phân phối để suy luận; nó tương đồng với tất cả mọi thứ chúng ta đã biết; nhưng cũng tránh giả định những thứ ta không biết. Đây là sự thuật lại của toán học cho một nguyên tắc thông thái cổ đại …"

b. Mục tiêu

Tìm ra mô hình nhất quán với tần số đồng xảy ra nhưng cũng có sự không chắc chắn cao. Kết hợp nhiều chứng cứ vào một mô hình xác suất. Quá trình ước tính tham số được lặp lại để tìm ra mô hình có độ hỗn loạn cao nhất.

c. Các định nghĩa

i. Thuộc tính trong NLP

Có hai khái niệm về feature bao gồm:

- input-feature: đặc điểm của token chưa được gán nhãn
- joint-feature: đặc điểm của token đã được gán nhãn

Trong lý thuyết mô tả mô hình MaxEnt, input-feature được gọi là "bối cảnh", còn joint-feature mới được gọi là "thuộc tính".

ii. Thuộc tính trong MaxEnt

Một thuộc tính là một hàm với giá trị ràng buộc. Mỗi thuộc tính sẽ có trọng số, trọng số dương cho thấy khả năng chính xác cao và trọng số âm cho thấy ngược lại.

Tổng quát về thuộc tính, theo Ratnaparkhi (1998):

$$f_j(a,b) = \begin{cases} 1 & \text{if } a = x \text{ and } q(b) = \text{true} \\ 0 & \text{otherwise} \end{cases}$$

Trong đó:

- f_j: thuộc tính
- a: giá trị dự đoán

- b: bối cảnh ngữ nghĩa

Xét ví dụ về thuộc tính như sau:

$$[f_1(x, d) = [c = \text{LOCATION} \wedge w_{-1} = \text{"in"} \wedge \text{isCapitalized}(w)]$$

Hàm thuộc tính trên sẽ trả về trọng số (khả năng) thể loại là LOCATION khi từ trước đó là "in" và từ hiện tại được viết hoa./

Trong đó:/.

- $f_1(x, d)$ là hàm thuộc tính
- x là dữ liệu đầu vào
- d là tài liệu
- c là thể loại đang xét
- w là từ đang được xét trong tài liệu
- w_{-1} là từ đứng trước từ w
- $\text{isCapitalized}(w)$ là hàm giả định có thể xét nếu một từ có viết hoa hay không

Chính xác hơn, thuộc tính được định nghĩa bởi phương trình sau:

iii. Nguyên tắc tối đa hỗn loạn

Chọn mô hình p* với điều kiện như sau:

$$p^* = \arg\max_{p \in P} H(p)$$

Trong đó:

- p^* là phân phối xác suất tối ưu có thể tối đa độ hỗn loạn với ràng buộc thuộc tính.
- $arg\max$ là giá trị đầu vào mà tại đó đạt được đầu ra tối đa
- $H(p)$ là độ hỗn loạn của phân phối (p).

Độ hỗn loạn được xác định bằng công thức sau:

$$H(p) = -\sum_{a,b} p(b \mid a) \log \left(\frac{p(a \mid b)}{p(b)} \right)$$

Trong đó:

- $H(p)$ là độ hỗn loạn của phân phối.
- Tổng được xác định bởi mọi cặp giá trị.
- $p(b \mid a)$ là xác suất của b với a đã được biết trước.
- Hàm log xác định tỷ lệ của xác suất có điều kiện $(a \mid b)$ và xác suất ngưỡng $p(b)$.

Mô hình xác suất được xác định như sau:

$$P = \prod_{j=1}^{k} \exp\left(E_p Q(f_j); j = 1, \ldots, k\right)$$

Trong đó:

- P là mô hình xác suất.
- f_j là các thuộc tính của bộ dữ liệu.
- E_p là kỳ vọng của mô hình
- Q là hàm có trọng số.

d. Mô hình Maxent và Logistic Regression

Mối liên hệ giữa mô hình MaxEnt và hồi quy Logistic đã được nhắc đến bởi nhiều nhà nghiên cứu về lĩnh vực Xử lý ngôn ngữ tự nhiên.

Theo Berger et al. (1996), giải pháp cho vấn đề tối ưu đã nêu trên chính là phân phối xác suất của mô hình **phân phối đa thức hồi quy Logistic** (multinomial Logistic Regression) với trọng số w giúp tối ưu khả năng xảy ra của dữ liệu huấn luyện.

Còn Manning (), mô hình Maxent trong xử lý ngôn ngữ tự nhiên cũng giống như mô hình Multi-class Logistic Regression (phân lớp đa nhãn) trong Thống kê hay Học máy.

Trong bài toán phân lớp, các loại mô hình quan trọng gồm có mô hình log-tuyến tính, hồi quy logistic (McCullagh và Nelder, 1989) và mô hình decomposable (Bruce và Wiebe, 1999). Theo Zhang và Oles (2001), thử nghiệm trên bộ dữ liệu Reuters và một số bộ dữ liệu khác cho thấy Logistic Regression mang lại F1-score tốt nhất cho bài toán phân loại văn bản. Cũng

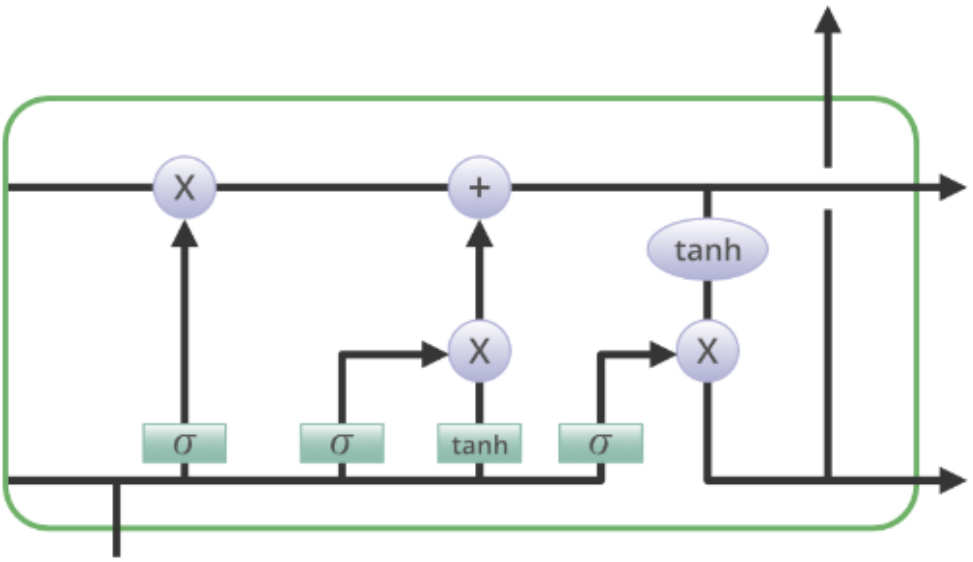
theo Zhang và Oles (2001), regularization đóng vai trò quan trọng trong kết quả của mô hình.

Cũng vì những lý do trên, nhóm đã lựa chọn mô hình phân phối đa thức hồi quy Logistic để ứng dụng cho bài toán phân lớp văn bản như một mô hình MaxEnt.

4. Mô hình học sâu

a. Giới thiệu mô hình mạng LSTM

Mô hình LSTM (Long Short-Term Memory) là một dạng đặc biệt của mạng nơ-ron hồi quy (Recurrent Neural Network - RNN), được phát triển với mục đích chính là mô hình hóa các chuỗi dữ liệu tuần tự. Mô hình này được đề xuất bởi Sepp Hochreiter và Jürgen Schmidhuber vào năm 1997, và từ đó đã trở thành một trong những mô hình RNN được sử dụng rộng rãi nhất.



Hình 1: Cấu trúc của Mạng LSTM - GeeksforGeeks

Điểm nổi bật của LSTM là khả năng xử lý hiệu quả các chuỗi dữ liệu dài, cũng như khắc phục vấn đề biến mất gradient (Vanishing Gradient Problem - VGP) - một vấn đề thường gặp trong các mạng RNN truyền thống. Để làm được điều này, LSTM sử dụng các đơn vị bộ nhớ (Memory Cells) đặc biệt, giúp lưu trữ các thông tin quan trọng từ quá khứ và đưa ra quyết định về việc "quên" hoặc "giữ" thông tin này trong quá trình dự đoán. Điều này giúp LSTM có khả năng học và nhớ được các mối liên hệ dài hạn trong dữ liệu.

Một đơn vị bộ nhớ LSTM có cấu trúc gồm các thành phần sau:

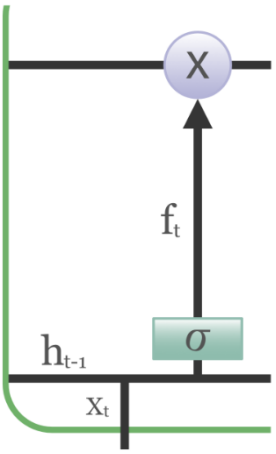
- **Cổng quên** (Forget Gate): Đây là cơ chế quyết định mức độ "quên" thông tin từ bộ nhớ trước đó, xác định thông tin nào sẽ được truyền qua và thông tin nào sẽ bị loại bỏ. Công thức xác định như sau:

$$f_t = \sigma(W_f \times [h_{t-1}, x_t] + b_f)$$

Trong đó:

- h_t là đầu ra tại thời điểm t.
- x_t là đầu vào tại thời điểm t.
- f_t là cổng quên
- W_f, b_f là các ma trận trọng số và vector điều chỉnh
- σ là hàm sigmoid, được sử dụng để tính toán giá trị của các cổng.

Hàm sigmoid biến đổi giá trị đầu vào của nó thành một giá trị trong khoảng từ 0 đến 1, thường được sử dụng để "nén" giá trị đầu vào của nó vào một phạm vi cụ thể.



Hình 2: Cổng Quên của Cấu trúc của Mạng LSTM

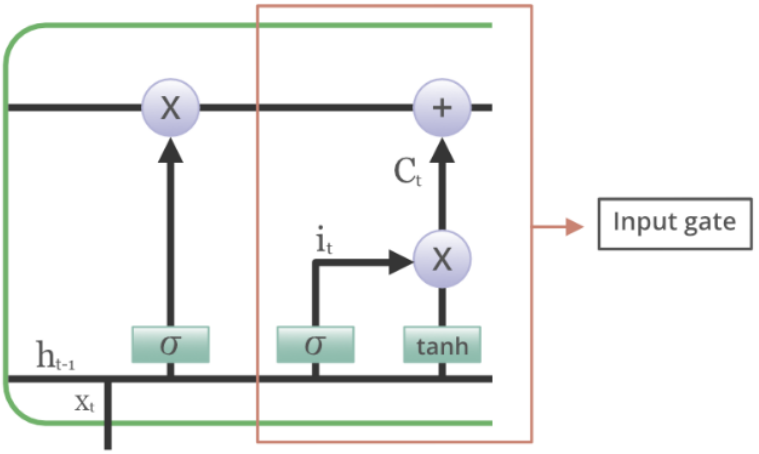
- **Cổng đầu vào** (Input Gate): Cơ chế này xác định mức độ cập nhật thông tin mới vào bộ nhớ, quyết định thông tin mới nào sẽ được cập nhật và thông tin nào sẽ bị bỏ qua. Công thức xác định như sau:

$$i_t = \sigma(W_i \times [h_{t-1}, x_t] + b_i) \; ; \; \widehat{c_t} = tanh(W_c \times [h_{t-1}, x_t] + b_c)$$

Trong đó:

- i_t là cổng vào.
- $\widehat{c}_t$ là giá trị đầu vào tại thời điểm t.
- $tanh$ là hàm hyperbolic tangent, được sử dụng để tính toán giá trị của trạng thái tế bào.

Hàm hyperbolic tangent biến đổi giá trị đầu vào của nó thành một giá trị trong khoảng từ -1 đến 1, cung cấp một phạm vi đầu ra rộng hơn so với hàm sigmoid.



Hình 3: Cổng Vào của Cấu trúc của Mạng LSTM

- **Cập nhật tế bào trạng thái** (Cell State Update)**:** Tế bào trạng thái được cập nhật dựa trên thông tin từ cổng quên và cổng đầu vào. Công thức tính toán cho việc cập nhật tế bào trạng thái là**:**

$$C_t = f_t \times C_{t-1} + i_t \times \widehat{c}_t$$

Trong đó:

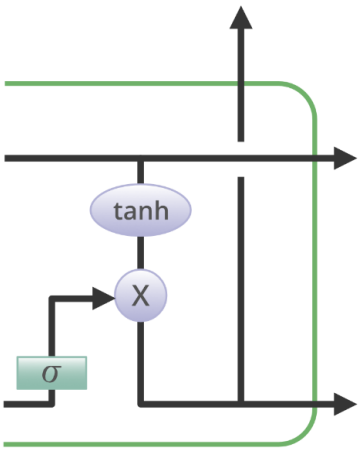
- C_t là trạng thái tế bào tại thời điểm t

- **Cổng đầu ra** (Output Gate): Cơ chế này xác định mức độ sử dụng thông tin từ bộ nhớ để tạo ra đầu ra của đơn vị bộ nhớ, quyết định thông tin nào sẽ được truyền đi và thông tin nào sẽ bị lọc.

$$o_t = \sigma(W_o \times [h_{t-1}, x_t] + b_o) \; ; \; h_t = o_t \times tanh(C_t)$$

Trong đó:

- o_t là cổng ra.



Hình 4: Cổng Ra của Cấu trúc của Mạng LSTM

Thông qua chuỗi các đơn vị bộ nhớ LSTM, mô hình có khả năng lưu trữ và truyền thông tin theo chiều thời gian trong các chuỗi dữ liệu tuần tự. Điều này cho phép LSTM hiểu và mô hình hóa các mẫu phức tạp trong dữ liệu, bao gồm sự phụ thuộc dài hạn, các tương quan ngắn hạn, và các quan hệ phi tuyến.

Mô hình LSTM đã được áp dụng rộng rãi trong nhiều lĩnh vực, từ xử lý ngôn ngữ tự nhiên (NLP), dịch máy, nhận diện giọng nói, dự báo chuỗi thời gian, đến nhiều ứng dụng khác. Trong lĩnh vực NLP, LSTM đã chứng tỏ hiệu suất tốt trong các tác vụ như phân loại văn bản, sinh văn bản, và dịch máy. Tuy nhiên ở phần này, nhóm sẽ tận dụng những ưu điểm vừa đề cập của mô hình LSTM để dự đoán nhãn cho các văn bản tiếng Việt.

b. Khái niệm liên quan

Để khởi tạo một mô hình mạng LSTM, nhóm nghiên cứu cần phải thêm vào các lớp (layers) phù hợp. Trong bài báo cáo này, nhóm sẽ thực hiện xây dựng mô hình bằng cách sử dụng các lớp thường được kết hợp với mạng LSTM, bao gồm Dense, Attention, Bidirectional và Embedding. Chi tiết như sau:

- **Dense Layer:**

  Dense layer, hay còn được gọi là fully connected layer, là một layer cơ bản trong mạng nơ-ron. Nó kết nối tất cả các đầu vào từ layer trước với tất cả các đầu ra của nó.

Các đầu vào được trọng số hóa và ánh xạ tới các đầu ra thông qua một hàm kích hoạt phi tuyến (non-linear activation function), thường là hàm ReLU (Rectified Linear Unit) hoặc sigmoid, hay thậm chí là Softmax được dùng trong các bài toán phân loại nhiều lớp (multi-class classification) được nhóm ứng dụng trực tiếp trong báo cáo.

Mục đích của Dense layer là giúp mô hình mở rộng không gian đặc trưng và học các biểu diễn phức tạp từ dữ liệu đầu vào.

- **Attention Layer:**

Attention layer được sử dụng để tập trung vào các phần quan trọng trong dữ liệu đầu vào. Trong bối cảnh của mạng LSTM, Attention layer có thể giúp mô hình tập trung vào các phần tử quan trọng trong chuỗi đầu vào. Nó tính toán một trọng số cho mỗi phần tử trong chuỗi đầu vào, sau đó trọng số này được sử dụng để làm nổi bật các phần tử quan trọng trong quá trình tính toán.

- **Bidirectional Layer:**

Bidirectional layer cho phép mạng LSTM xem xét thông tin từ cả quá khứ và tương lai của mỗi điểm dữ liệu trong chuỗi. Thông thường, mạng LSTM tiến (forward LSTM) chỉ xem xét thông tin từ quá khứ đến hiện tại, trong khi mạng LSTM lùi (backward LSTM) chỉ xem xét thông tin từ tương lai đến hiện tại.

Bidirectional layer kết hợp cả hai mạng LSTM này để có được thông tin toàn diện từ cả quá khứ và tương lai. Điều này giúp mô hình có khả năng xử lý các phụ thuộc dài hạn và tạo ra biểu diễn tốt hơn cho dữ liệu chuỗi.

- **Embedding Layer:**

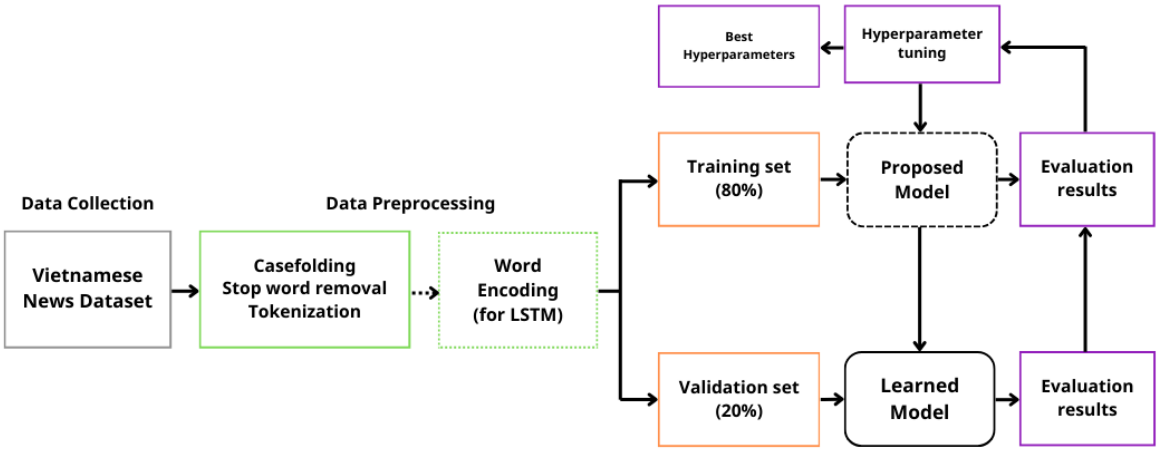
Embedding layer được sử dụng để biểu diễn từ hoặc đối tượng trong dữ liệu dưới dạng các vector có chiều thấp. Trong mạng LSTM, Embedding layer thường được sử dụng để biểu diễn từ vựng (vocabulary) hoặc chuỗi các từ. Nó ánh xạ mỗi từ hoặc đối tượng thành một vector dày đặc, trong đó các thông tin về sự tương đồng và mối quan hệ giữa các từ được bảo toàn. Embedding layer giúp mô hình hiểu được ngữ nghĩa và cấu trúc của dữ liệu ngôn ngữ và cải thiện khả năng dự đoán và xử lý các tác vụ liên quan đến ngôn ngữ.

CHƯƠNG IV. THỰC HIỆN

1. Tổng quan

Về quy trình thực hiện, đầu tiên, nhóm lựa chọn bộ dữ liệu đã có sẵn. Tiếp theo, nhóm thực hiện các phương pháp tiền xử lý chuẩn cho dữ liệu, bao gồm có chuẩn hóa từ in hoa (casefolding), loại bỏ các từ thường sử dụng (stop word removal) và tạo token các từ (tokenization). Phương pháp tiền xử lý này được thực hiện cho mọi loại mô hình, trừ mô hình LSTM, với mô hình này, nhóm có bổ sung thêm phương pháp encoding để phù hợp với cấu trúc của mô hình này. Sau đó, nhóm chia bộ dữ liệu thành hai phần: train set và validation set với tỷ lệ 80% - 20%. Nhóm thử nghiệm các tổ hợp siêu tham số khác nhau, in kết quả cho train set và test set để từ đó lựa ra mô hình tốt nhất. Mô hình tốt nhất thuộc từng nhóm sẽ được lựa chọn để trực quan hóa và thực nghiệm trong giao diện của nhóm ở phần sau.

Phương pháp nghiên cứu của nhóm có thể được tóm tắt qua sơ đồ sau:



Hình 5: Phương pháp nghiên cứu của nhóm

2. Bộ dữ liệu

Bộ dữ liệu "News Dataset Vietnameses" được thu thập từ Báo Lao Động ngày 19/05/2022, chứa các thông tin liên quan đến tin tức mỗi ngày, các xu hướng hiện hành cũng như những sự kiện nổi bật tại Việt Nam. Bộ dữ liệu bao gồm 39.000 quan sát và 8 thuộc tính được mô tả trong bảng sau:

STT	Tên thuộc tính	Mô tả
1	URL	Đường dẫn liên kết đến bài báo
2	Title	Tên (tiêu đề) bài báo

3	Summary	Tóm tắt nội dung bài báo
4	Contents	Nội dung chi tiết bài báo
5	Data	Ngày viết (xuất bản) của bài báo
6	Author(s)	Tác giả của bài báo
7	Category	Thể loại của bài báo
8	Tags	Các chủ đề liên quan đến bài báo

Bảng 1. Bảng mô tả các thuộc tính trong bộ dữ liệu

Thông tin về bộ dữ liệu "News Dataset Vietnameses":

```
# Thông tin bộ dữ liệu
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 39000 entries, 0 to 38999
Data columns (total 8 columns):
 #   Column     Non-Null Count  Dtype 
---  ------     --------------  ----- 
 0   URL        39000 non-null  object
 1   Title      39000 non-null  object
 2   Summary    38413 non-null  object
 3   Contents   36961 non-null  object
 4   Date       39000 non-null  object
 5   Author(s)  38234 non-null  object
 6   Category   39000 non-null  object
 7   Tags       39000 non-null  object
dtypes: object(8)
memory usage: 2.4+ MB
```

Quan sát 5 dòng đầu tiên trong bộ dữ liệu:

```
# Quan sát dữ liệu
df.head()
```

	URL	Title	Summary	Contents	Date	Author(s)	Category	Tags
0	https://laodong.vn/ban-doc/cach-tham-gia-bao-h...	Cách tham gia Bảo hiểm xã hội tự nguyện sau kh...	Bảo hiểm xã hội Việt Nam thông tin về mức đóng...	Bạn đọc hỏi: Tôi mới đi làm và tham gia Bảo hi...	Thứ tư, 22/09/2021 10:00 (GMT+7)	ANH THƯ	Bạn đọc	['Bảo hiểm xã hội', 'Bảo hiểm xã hội Việt Nam'...
1	https://laodong.vn/ban-doc/quang-tri-cay-co-da...	Quảng Trị: Cây cọ dầu ở nơi có dự án của Vingr...	Quảng Trị - Ở địa điểm xây dựng khu đô thị thư...	Sau khi đấu giá trúng khu đất rộng 132.415,6 m...	Thứ tư, 09/03/2022 18:13 (GMT+7)	HƯNG THƠ	Bạn đọc	['Quảng Trị', 'Dự án Vingroup ở Quảng Trị', 'V...
2	https://laodong.vn/ban-doc/lao-cai-dan-khat-nu...	Lào Cai: Dân khát nước bên công trình nước sạc...	Lào Cai – Hàng trăm hộ dân vẫn phải dùng nước ...	Phản ánh đến PV, người dân xã Cam Đường, TP.Là...	Thứ tư, 11/05/2022 16:57 (GMT+7)	Văn Đức	Bạn đọc	['Lào Cai', 'Nước sạch', 'nhà máy nước sạch', ...
3	https://laodong.vn/ban-doc/quy-dinh-ve-viec-la...	Quy định về việc làm giám định sức khoẻ để ngh...	Bạn đọc Hà My hỏi: Tôi có làm giám định sức kh...	Về vấn đề trên, Bảo hiểm xã hội Việt Nam trả l...	Chủ nhật, 20/06/2021 20:16 (GMT+7)	Minh Hương	Bạn đọc	['Bảo hiểm xã hội', 'Luật Bảo hiểm xã hội', 'N...
4	https://laodong.vn/ban-doc/iphone-12-va-dang-c...	iPhone 12 và đẳng cấp "móc túi" người dùng của...	Âm thầm tăng giá bán nhưng lại giảm phụ kiện đ...	Nếu là iFan, hẳn các bạn còn nhớ, ngày 11.9.20...	Thứ năm, 15/10/2020 06:58 (GMT+7)	Đạt Phan	Bạn đọc	['Bảo vệ môi trường', 'Apple', 'Iphone', 'Appl...

Có tổng cộng 39 thể loại (Category) có trong bộ dữ liệu "News Dataset Vietnameses":

```
num_categories = df['Category'].nunique()
print(f'Số lượng thể loại có trong bộ dữ liệu: {num_categories}')
```

```
Số lượng thể loại có trong bộ dữ liệu: 39
```

3. Tiền xử lý

a. Xử lý các giá trị bị thiếu và trùng lặp

Kiểm tra các giá trị trùng lặp và số giá trị bị thiếu ở mỗi cột:

```
Giá trị trùng lặp: 20687
Giá trị bị thiếu:
URL             0
Title           0
Summary       587
Contents     2039
Date            0
Author(s)     766
Category        0
Tags            0
dtype: int64
```

Sau khi kiểm tra, ta nhận thấy bộ dữ liệu có tổng cộng 20.687 giá trị trùng lặp. Ở 3 cột "Summary", "Contents", "Author(s)" , giá trị bị thiếu ở mỗi cột chiếm dưới 6%. Điều này có thể ảnh hưởng đến kết quả phân tích nên nhóm quyết định loại bỏ những dòng trùng lặp và những dòng bị thiếu giá trị. Kết quả sau khi xử lý:

```
URL          0
Title        0
Summary      0
Contents     0
Date         0
Author(s)    0
Category     0
Tags         0
dtype: int64
```

b. Xử lý nhãn

Quan sát số lượng bài báo của từng thể loại (Category):

```
print('\nSố lượng bài viết của từng thể loại:')
category_counts = df['Category'].value_counts()
category_counts
```

```
Số lượng bài viết của từng thể loại:
Pháp luật                985
Thế giới                 975
Kinh doanh               966
Thể thao                 965
Xã hội                   964
Sức khỏe                 952
Công đoàn                951
Thời sự                  948
```

```
Giáo dục                        945
Văn hóa - Giải trí              941
Media                           921
Bạn đọc                         920
Gia đình - Hôn nhân             873
Bất động sản                    869
Xe +                            826
Tấm Lòng Vàng                   727
Công nghệ                       582
Lao Động cuối tuần              495
Lưu trữ                         379
Lao Động & Đời sống             319
Diễn đàn                        278
Tin hoạt động                   216
Tin tức việc làm                166
Thông tin tiện ích               96
Sự kiện Bình luận                58
Phóng sự                         47
Tin địa phương                   31
Quỹ TLV                          28
Thông tin doanh nghiệp           15
Tin bài xem thêm                 13
Du lịch                          11
Sổ tay kinh tế                    7
Phóng sự - Điều tra               2
Người Việt tử tế                  2
Tin bài nổi bật                   1
Tin bài liên quan                 1
Tản mạn - Chuyện dọc đường        1
Video                             1
Lao Động Xuân                     1
Name: Category, dtype: int64
```

Bộ dữ liệu bao gồm 39 thể loại về tin tức, trong đó "Pháp luật", "Thế giới", "Kinh doanh", "Thể thao" và "Xã hội" là 5 thể loại có nhiều bài báo xuất hiện nhất. Có thể thấy đây là các thể loại phổ biến mà mà người đọc quan tâm và muốn theo dõi, do đó số lượng bài báo về các chủ đề này cũng chiếm phần lớn so với các thể loại còn lại. Tuy nhiên, khi quan sát ta nhận thấy phân phối của 39 lớp không đồng đều với nhau. Số lượng bài báo giữa các lớp (nhãn) có hiện tượng không cân bằng, với các nhãn có số lượng bài báo lên đến hơn 900, trong khi một vài nhãn lại có duy nhất 1 bài báo. Điều này có thể tạo ra những khó khăn cũng như thách thức cho việc ứng dụng mô hình học máy, vì chúng có thể trở nên thiên vị hơn đối với lớp đa số và có thể hoạt động kém trên các lớp thiểu số. Do đó nhóm tiến hành cập nhật nhãn để có sự phân loại mới cho các bài báo.

```
category_mapping = {
    'Lao Động Xuân': 'Lao Động & Đời sống',
    'Video': 'Media',
    'Tản mạn - Chuyện dọc đường': 'Bạn đọc',
    'Phóng sự - Điều tra': 'Thời sự',
    'Thông tin doanh nghiệp': 'Kinh doanh',
    'Lao Động cuối tuần': 'Lao Động & Đời sống',
    'Người Việt tử tế': 'Tấm Lòng Vàng',
    'Sổ tay kinh tế': 'Kinh doanh',
    'Quỹ TLV': 'Tấm Lòng Vàng',
    'Tin địa phương': 'Thời sự',
    'Du lịch': 'Văn hóa - Giải trí',
    'Phóng sự': 'Thời sự'}
# Áp dụng việc thay đổi nhãn
df['Category'] = df['Category'].replace(category_mapping)
df['Category'].value_counts()
```

Kết quả sau khi cập nhật nhãn mới:

```
Thời sự                1028
Kinh doanh              988
Pháp luật               985
Thế giới                975
Thể thao                965
Xã hội                  964
Sức khỏe                952
Văn hóa - Giải trí      952
Công đoàn               951
Giáo dục                945
Media                   922
Bạn đọc                 921
Gia đình - Hôn nhân     873
Bất động sản            869
Xe +                    826
Lao Động & Đời sống     815
Tấm Lòng Vàng           757
Công nghệ               582
Lưu trữ                 379
Diễn đàn                278
Tin hoạt động           216
Tin tức việc làm        166
Thông tin tiện ích       96
Sự kiện Bình luận        58
Tin bài xem thêm         13
Tin bài liên quan         1
Tin bài nổi bật           1
Name: Category, dtype: int64
```

Tuy nhiên, bộ dữ liệu vẫn tồn tại những dòng chứa nội dung không quy theo một thể loại nhất định, nhóm sẽ tiến hành loại bỏ những dòng này, việc loại bỏ được thực hiện như sau:

```
df.drop(df[df['Category'] == 'Lưu trữ'].index,inplace= True)
df.drop(df[df['Category'] == 'Tin hoạt động'].index,inplace=
True)
df.drop(df[df['Category'] == 'Diễn đàn'].index,inplace= True)
df.drop(df[df['Category'] == 'Sự kiện Bình
luận'].index,inplace= True)
df.drop(df[df['Category'].isin(['Tin bài nổi bật','Tin bài
liên quan','Tin bài xem thêm','Lao Động & Đời sống','Bạn
đọc','Media','Công đoàn','Xã hội','Kinh doanh','Thời
sự'])].index,inplace= True)
df.reset_index(drop = True, inplace = True)
```

Sau khi xử lý nhãn, còn tổng cộng 13 thể loại có trong bộ dữ liệu. Tên thể loại và số lượng bài viết thuộc từng thể loại được liệt kê bên dưới bao gồm:

```
Pháp luật              985
Thế giới               975
Thể thao               965
Văn hóa - Giải trí     952
Sức khỏe               952
Giáo dục               945
Gia đình - Hôn nhân    873
Bất động sản           869
Xe +                   826
Tấm Lòng Vàng          757
Công nghệ              582
Tin tức việc làm       166
Thông tin tiện ích      96
Name: Category, dtype: int64
```

Nhóm quyết định giữ lại 2 thể loại là “Tin tức việc làm” và “Thông tin tiện ích” bởi sau khi xem xét kỹ lưỡng, nhóm nhận thấy rằng với nhãn “Tin tức việc làm” có sự thống nhất về nội dung với hầu hết các bài báo là về tuyển dụng nhân sự. Còn với nhãn “Thông tin tiện ích”, đây là 1 nhãn với nội dung là cập nhật giá vàng theo các mốc thời gian nhất định. Đây đều là 2 thể loại có nội dung đồng nhất nên có thể sẽ tăng hiệu quả cho việc phân loại.

c. Text Mining

Vì mục tiêu nghiên cứu của đồ án là xây dựng hệ thống dự đoán nhãn cho văn bản tiếng Việt. Vì vậy ta sẽ tạo DataFrame mới 'machine_df' để xử lý trên các biến 'Category' và 'Summary'. Với biến 'Category' là biến target phân loại.

```
machine_df = df[['Category','Summary']]
machine_df.head(5)
```

	Category	Summary
0	Bất động sản	Bước sang quý III.2019, giao dịch bất động sản...
1	Bất động sản	Tin tưởng vào việc hùn vốn với giám đốc công t...
2	Bất động sản	Chủ tịch UBND tỉnh Bắc Ninh vừa phê duyệt quy ...
3	Bất động sản	Chuyển đổi quyền sử dụng đất (đổi đất) chỉ áp ...
4	Bất động sản	Không ai phủ nhận ý nghĩa của Đề án giãn dân p...

Trong bước tiếp theo, nhóm tiến hành khai thác văn bản (Text Mining) bằng cách sử dụng thư viện "pyvi" để thực hiện việc tách từ tiếng Việt và sử dụng một danh sách từ dừng (stop words) từ file 'vietnamese.txt' để loại bỏ các từ không quan trọng. Đầu tiên, tạo hàm "text_preprocessing" cho bộ dữ dữ liệu chứa các lệnh: loại bỏ dấu câu, chữ số và ký tự đặc biệt, tách một câu thành các từ và đồng thời chuyển chúng về chữ thường, tách thành từng từ và loại bỏ stop words.

```
def text_preprocessing(sent):
    with open('vietnamese.txt', 'r') as f:
        stop_words = f.read().split('\n')
    sent = re.sub(f'[{string.punctuation}\d\n]', '', sent)
    sent = re.sub(r'[^\w\s]', '', sent)
    sent = ViTokenizer.tokenize(sent.lower())
    sent = [w for w in sent.split()]
    sent = [w for w in sent if w not in stop_words]
    return ' '.join(sent)
```

Thêm một cột mới vào DataFrame "machine_df" có tên là 'tokenized_contents'. Cột này được tạo ra bằng cách áp dụng hàm "text_preprocessing" lên cột "Summary" của DataFrame "df":

```
machine_df['tokenized_contents'] = df['Summary'].apply(lambda
x: text_preprocessing(x))
machine_df
```

Kết quả thu được từ DataFrame "machine_df" như sau:

	Category	Summary	tokenized_contents
0	Bất động sản	Bước sang quý III.2019, giao dịch bất động sản...	quý iii giao_dịch bất_động_sản thị_trường đà_n...
1	Bất động sản	Tin tưởng vào việc hùn vốn với giám đốc công t...	tin_tưởng hùn vốn giám_đốc công_ty bất_động_sả...
2	Bất động sản	Chủ tịch UBND tỉnh Bắc Ninh vừa phê duyệt quy ...	chủ_tịch ubnd tỉnh bắc ninh phê_duyệt quy_hoạc...
3	Bất động sản	Chuyển đổi quyền sử dụng đất (đổi đất) chỉ áp ...	chuyển_đổi quyền sử_dụng đất đổi đất áp_dụng đ...
4	Bất động sản	Không ai phủ nhận ý nghĩa của Đề án giãn dân p...	phủ_nhận ý_nghĩa đề_án giãn dân_phố cổ hà_nội ...
...	...	...	...
9938	Xe +	Mới đây, trên mạng xã hội giao thông đăng tải ...	mới_đây mạng xã_hội giao_thông đăng_tải clip b...
9939	Xe +	Theo thống kê của Uỷ ban ATGT Quốc gia, trong ...	thống_kê uỷ_ban atgt quốc_gia đầu toàn_quốc xả...
9940	Xe +	Ca sĩ Lily Chen tậu xe Mercedes G63 AMG có giá...	ca_sĩ lily chen tậu xe mercedes g amg có_giá l...
9941	Xe +	Xe máy sau một thời gian vận hành, một số bộ p...	xe_máy thời_gian vận_hành một_số bộ_phận rơ rã...
9942	Xe +	Xe hơi là phương tiện di chuyển tối ưu trong n...	xe_hơi phương_tiện di_chuyển tối_ưu nắng_nóng ...

Lưu DataFrame "machine_df" vào tệp Excel đặt tên là 'data.xlsx':

```
machine_df.to_excel('data.xlsx')
```

4. Huấn luyện

Sử dụng train_test_split từ thư viện sklearn, nhóm chia dữ liệu thành 80% train set và 20% validation set.

a. Mô hình học máy

Đối với các mô hình học máy thông thường, nhóm đã sử dụng các mô hình phân lớp dựa trên xác suất (Naive Bayes, Decision Tree, Random Forest) cũng như phân lớp nhị phân (SVM) để tiến hành huấn luyện

Nhóm đã sử dụng TFIDF đối với những mô hình học máy đơn giản để biểu diễn các câu sau khi tiền xử lý thành dạng vector số. Trong đó, ứng với mỗi câu, vector sẽ được tính bằng cách tính toán tần suất xuất hiện của từ sau khi fit_transform với tập train

	SVM	Naive Bayes	Decision Tree	Random Forest
Accuracy	0.6938	0.4439	0.5268	0.6219
Precision	0.7177	0.5054	0.5528	0.6512
Recall	0.6904	0.4727	0.5517	0.6307
F1-Score	0.7008	0.4651	0.5512	0.6381

Bảng 2. Chỉ số chính xác của các mô hình học máy

Mô hình SVM có hiệu suất cao nhất trên tất cả các độ đo, đặc biệt là độ chính xác (Accuracy) và F1-Score. Đây là một phương pháp được đánh giá cao cho các mô hình phân loại văn bản, nó hoạt động tốt trên trong không gian vector với số chiều cao. Trong khi đó, đối với các phương pháp còn lại gặp phải một hạn chế khá lớn khi phải đối mặt với lượng dữ liệu lớn và sự đa dạng về từ của ngôn ngữ tiếng Việt khi thực hiện việc tính toán tần suất.

b. Maxent Classifier

Nhóm sử dụng mô hình Logistic Regression từ thư viện sklearn (LogisticRegression). Nhóm sử dụng các tham số mặc định để đánh giá đầu tiên là {C=1.0, penalty='l2', solver='lbfgs'}.

Giải thích các tham số được sử dụng trong mô hình:

Hàm mất mát của Logistic Regression với điều chuẩn có công thức tổng quát như sau:

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}[y^{(i)}\log(h_\theta(x^{(i)})) + (1-y^{(i)})\log(1-h_\theta(x^{(i)}))] + \frac{1}{2C}\sum_{j=1}^{n}\theta_j^2$$

Trong đó:

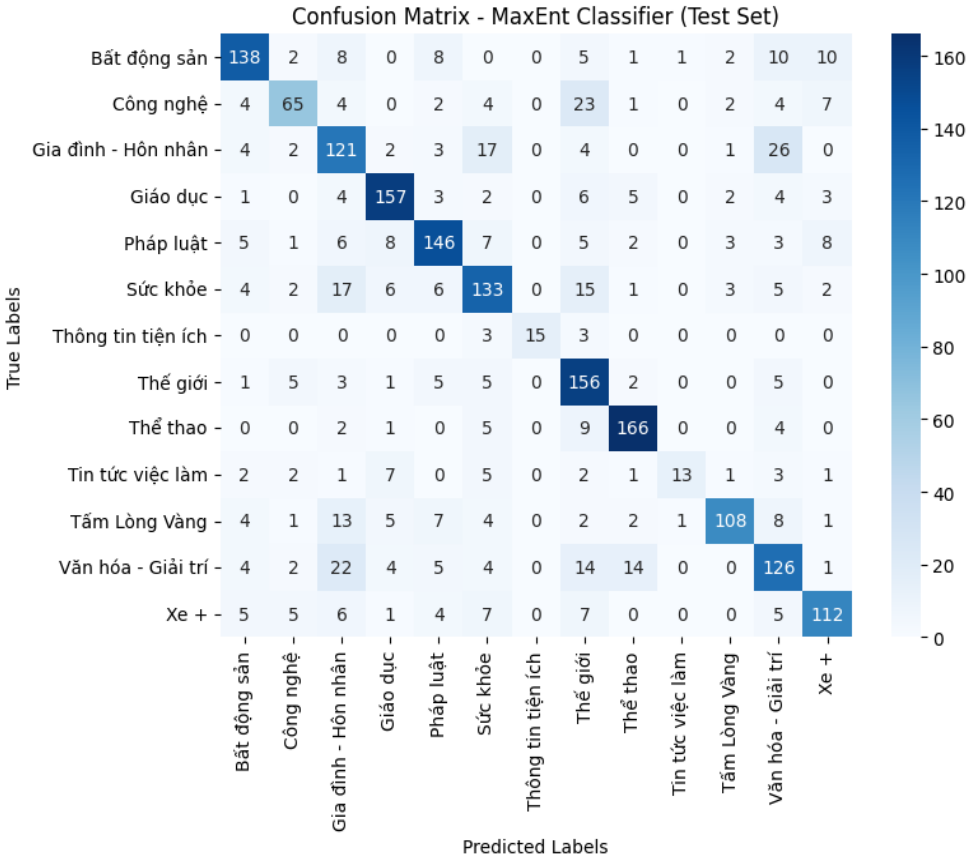
- **'C'** là nghịch đảo độ mạnh của hàm điều chuẩn (regularization). Khi C càng nhỏ, độ mạnh của điều chuẩn càng lớn.
- **'penalty'** cụ thể hóa hàm điều chuẩn (regularization term). Các giá trị có thể gồm 'l1', 'l2', và 'elasticnet'.
    - Khi penalty = 'l1', thuật toán tương đương với Ridge Regression - thêm các giá trị tuyệt đối của hệ số vào hàm mất mát để tạo sự thưa thớt cho mô hình.
    - Khi penalty = 'l2', thuật toán tương đương với Lasso Regression - thêm các giá trị bình phương của trọng số vào hàm mất mát để loại trừ ảnh hưởng của hệ số lớn.
    - Khi penalty = 'l2', thuật toán tương đương với Elastic Net Regression.
- '**solver**' lựa chọn thuật toán tối ưu hóa để sử dụng
    - 'solver'='liblinear' phù hợp với các bộ dữ liệu nhỏ.
    - 'solver'='lbfgs' phù hợp với các bộ dữ liệu vừa và nhỏ.

Kết quả thu được:

Thời gian huấn luyện trung bình	Test Accuracy
22.15309334	0.690896

Bảng 3. Chỉ số chính xác và thời gian thực hiện của mô hình Maxent

Ngoài ra, nhóm cũng đã thực hiện dự đoán cho test set (chiếm 20% dữ liệu ban đầu) và biểu diễn kết quả bằng ma trận nhầm lẫn như sau:



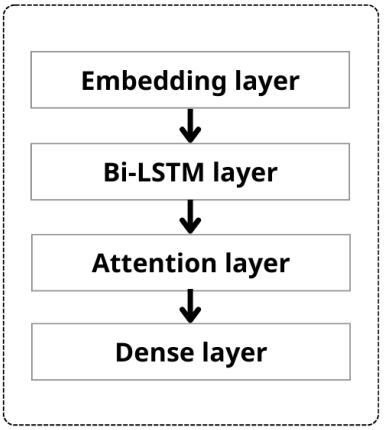
Hình 6: Ma trận nhầm lẫn - Maxent Classifier

Nhận xét:

- **Một số nhãn dễ bị nhầm lẫn**: Nhãn "Gia đình - Hôn nhân" có khả năng bị nhầm lẫn với nhãn "Sức khỏe" và "Giải trí". Tương tự, nhãn "Văn hóa - Giải trí" cũng có thể bị nhầm lẫn thành "Thế giới", "Thể thao" và "Gia đình - Hôn nhân". Điều này có thể được giải thích vì nội dung của các nhãn này có chứa nhiều từ khóa của nhãn khác, gây nên hiện tượng nhầm lẫn khi dự đoán bởi mô hình.

- **Siêu tham số khởi điểm khá tốt**: Có thể thấy, mô hình mang lại kết quả và thời gian thực hiện tương đối tốt với các siêu tham số được lựa chọn ban đầu. Để cải thiện hiệu quả của mô hình, nhóm sẽ thực hiện điều chỉnh siêu tham số để tìm ra các giá trị tốt nhất.

c. Mô hình học sâu (LSTM)



Hình 7: Cấu trúc của mạng Bidirectional LSTM

Định nghĩa lớp **Attention** để sử dụng trong mô hình. Lớp **Attention** có nhiệm vụ tạo trọng số cho các đầu vào dựa trên sự chú ý của mô hình. Nó có hai phương thức quan trọng là **build** và **call**.

- ***Phương thức build***:
  - **build** được gọi khi mô hình được khởi tạo. Nó có nhiệm vụ xác định kích thước đầu vào và khởi tạo các trọng số cần thiết cho lớp **Attention**.
  - Trong phương thức **build**, nhóm định nghĩa các biến trọng số cho lớp **Attention**. Các biến trọng số này sẽ được sử dụng để tính toán trọng số chú ý trong phương thức **call**.
  - Trong trường hợp này, nhóm sử dụng hai biến trọng số là **W** và **b**, với kích thước được xác định dựa trên đầu vào của lớp **Attention**.
- ***Phương thức call***:
  - **call** được gọi khi một đối tượng **Attention** được sử dụng trong mô hình chính.
  - Phương thức **call** nhận đầu vào là đầu ra của lớp trước đó trong mô hình và tính toán trọng số chú ý dựa trên đầu vào đó.

- Công thức tính toán trọng số chú ý trong lớp **Attention** thường dựa trên mô hình chú ý như **Bahdanau** hoặc **Luong**.

- Trong trường hợp này, nhóm sử dụng mô hình chú ý **Bahdanau**. Trong phương thức **call**, nhóm sẽ tính toán trọng số chú ý bằng cách sử dụng hàm **activation** (thường là hàm softmax) với đầu vào là một ma trận tương tác giữa các đầu vào và trọng số **W** đã được khởi tạo trong phương thức **build**.

- Sau đó, nhóm sẽ tính toán đầu ra của lớp **Attention** bằng cách lấy tổng có trọng số của các đầu vào ban đầu. Đầu ra này sẽ được truyền cho lớp tiếp theo trong mô hình.

Nhóm sử dụng mô hình mạng nơ-ron được xây dựng bằng cách sử dụng các lớp **Embedding** (lớp nhúng), **LSTM** (lớp LSTM), **Attention** (lớp Attention), **Bidirectional** (lớp Bidirectional) và **Dense** (lớp kết nối đầy đủ). Mô hình được xây dựng bằng cách xếp các lớp theo thứ tự và kết nối chúng với nhau.

Giải thích các tham số được sử dụng trong mô hình:

- **'output_dim'** được sử dụng để xác định số chiều của không gian nhúng (embedding space) cho các từ trong dữ liệu văn bản.

  - **Khi 'output_dim' ở ngưỡng rất nhỏ** (0.1):

    Khiến cho không gian nhúng sẽ có số chiều rất nhỏ làm mất mát thông tin và khả năng biểu diễn các đặc trưng của mô hình

    ➔ Mô hình sẽ gặp khó khăn trong việc học và trích xuất các mẫu phức tạp từ dữ liệu.

  - **Khi 'output_dim' tăng lên** (10.0):

    Không gian nhúng gia tăng làm mô hình có khả năng biểu diễn phức tạp hơn các mẫu từ và cấu trúc ngữ nghĩa.

    ➔ Mô hình sẽ có khả năng học sâu và trích xuất thông tin dữ liệu tốt hơn.

- **'lstm_units'** là một tham số của lớp LSTM và chỉ định số đơn vị (units) trong mạng LSTM. Mỗi đơn vị đại diện cho một tế bào LSTM, và các đơn vị này thực hiện các phép tính để xử lý dữ liệu chuỗi đầu vào.

- **Khi số đơn vị trong mạng nhỏ** (64):

  Mô hình LSTM có ít thông tin để học và xử lý, dẫn đến khả năng hạn chế trong việc học các mẫu phức tạp và trích xuất thông tin từ dữ liệu chuỗi

  ➔ Mô hình có thể bị Underfitting, tức là không đủ mạnh để học các quan hệ phức tạp trong dữ liệu.

- **Khi số đơn vị trong mạng lớn hơn** (128, 256):

  Mô hình có thể biểu diễn các mẫu phức tạp hơn và trích xuất thông tin từ dữ liệu chuỗi một cách hiệu quả hơn.

  Tuy nhiên, mô hình có khả năng Overfitting nếu số đơn vị quá lớn so với kích thước dữ liệu và tính chất của bài toán.

- **'optimizer'** quyết định thuật toán tối ưu hóa được sử dụng để điều chỉnh các trọng số của mạng và giảm thiểu hàm mất mát trong quá trình huấn luyện. Với hai giá trị chính là **'adam'** và **'rmsprop'**

  - **Adam** (Adaptive Moment Estimation) là một thuật toán tối ưu hóa tỷ lệ học tự động. Nó kết hợp các ưu điểm của hai thuật toán tối ưu hóa khác là RMSprop và Momentum. Adam thích ứng với tốc độ học tự động dựa trên gradient và moment thứ nhất và thứ hai của gradient. Adam được sử dụng phổ biến trong học sâu vì hiệu suất và tính ổn định của nó.

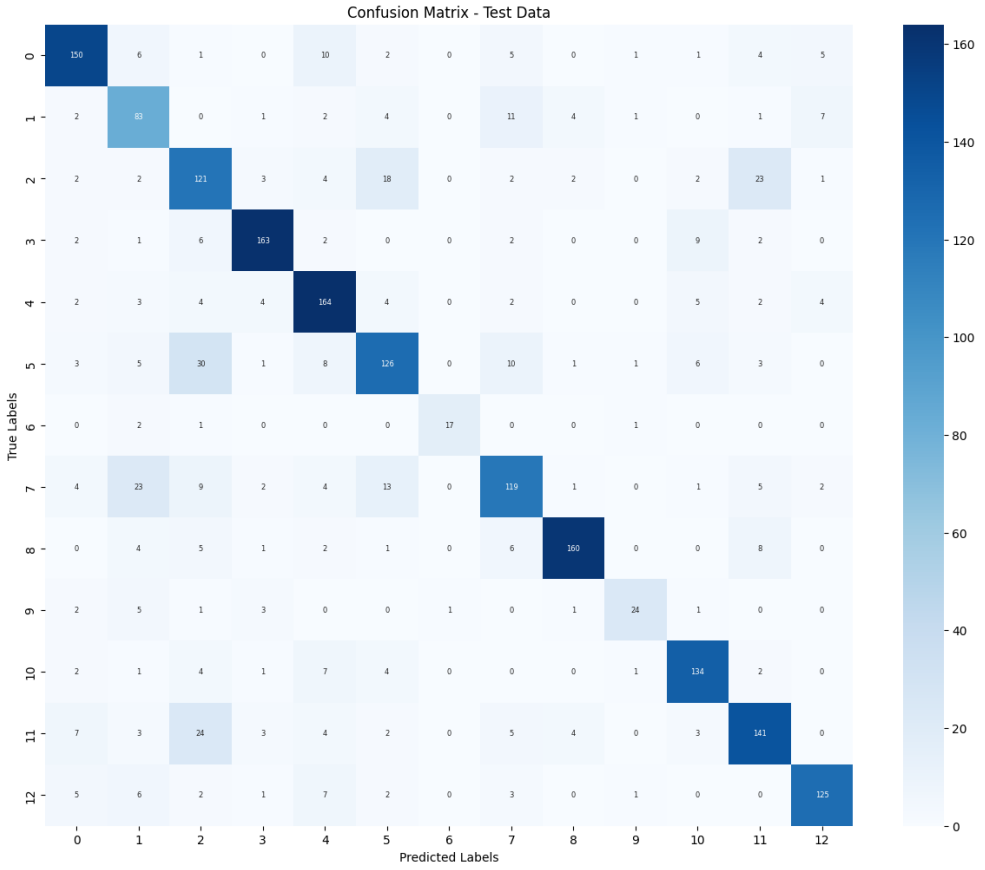
  - **RMSprop** (Root Mean Square Propagation) là một thuật toán tối ưu hóa dựa trên gradient. Nó cũng điều chỉnh tốc độ học tự động, nhưng chỉ dựa trên moment thứ hai của gradient. RMSprop sử dụng một tỷ lệ học riêng cho từng trọng số. Nó giúp giảm các thay đổi không đều đặn trong quá trình huấn luyện và có khả năng tìm được điểm tối ưu cục bộ tốt hơn.

Các tham số mặc định để đánh giá đầu tiên là {lstm_units=256, optimizer='adam'}. Nhóm thực hiện huấn luyện trên 5 epochs với batch_size = 32.

Kết quả thu được:

Thời gian huấn luyện trung bình	Test Accuracy
439.935480	0.6317

Bảng 4. Chỉ số chính xác và thời gian thực hiện của mô hình LSTM



Hình 8: Ma trận nhầm lẫn - LSTM

Nhận xét:

- Dựa trên phân tích từ ma trận nhầm lẫn, nhóm nhận thấy rằng các vấn đề mà mô hình Maxent gặp phải cũng xuất hiện trong mô hình LSTM. Điều này chỉ ra rằng, để cải thiện hiệu suất của mô hình, nhóm cần áp dụng thêm các phương pháp tối ưu hóa cho quá trình huấn luyện dữ liệu.

Đề xuất:

Có thể áp dụng các kỹ thuật tăng cường dữ liệu và tinh chỉnh siêu tham số khác nhau nhằm cải thiện hiệu suất của mô hình, có thể kể đến các kỹ thuật tăng cường dữ liệu như thay đổi từ vựng, đảo ngược thứ tự câu, thêm nhiễu, …. Ngoài ra, cũng có thể tinh chỉnh các siêu tham số của mô hình như số chiều Embedding, số lớp LSTM, số nút ẩn, hàm kích hoạt, tốc độ học, ….

5. Điều chỉnh siêu tham số

a. Mô hình học máy

Nhóm thực hiện điều chỉnh siêu tham số bằng phương pháp Grid Search sử dụng thư viện sklearn (GridSearchCV), với số kfold = 5.

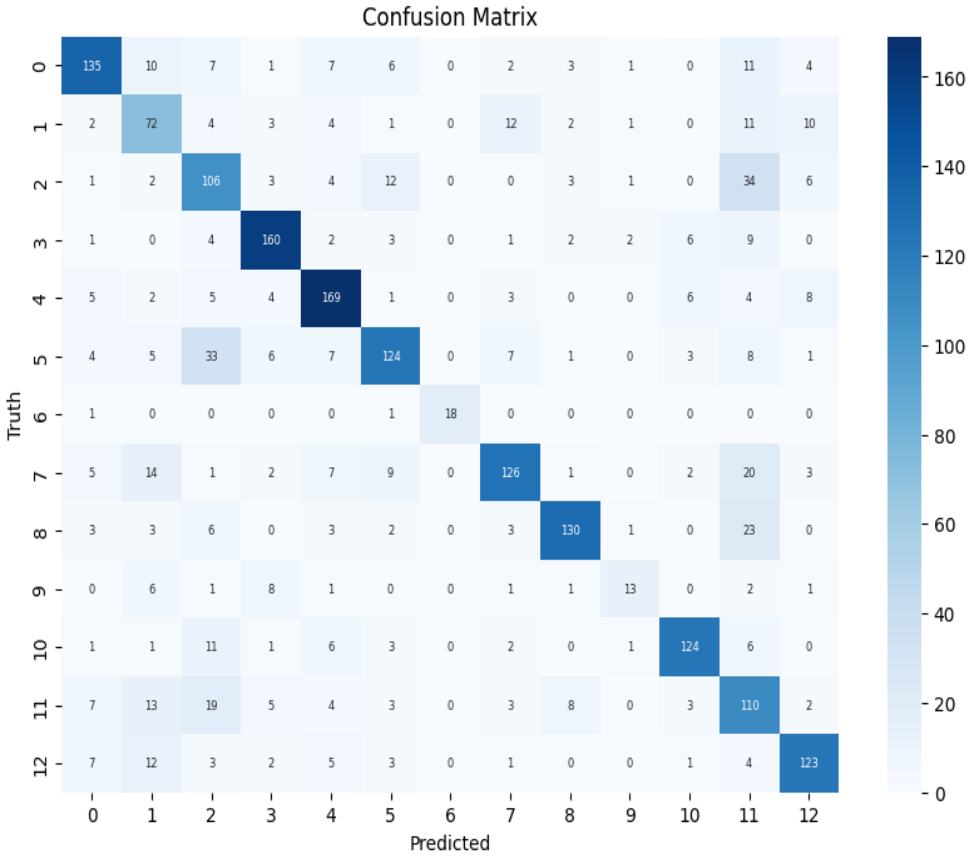
Sau khi thực hiện điều chỉnh siêu tham số và đánh giá trên tập validation, nhóm thu được tham số tốt nhất của mô hình SVM thông qua bảng dưới đây:

Giá trị các tham số	Validation Accuracy
{'C': 10, 'kernel': 'linear'}	0.70035

Bảng 5. Kết quả điều chỉnh siêu tham số của mô hình SVM

Để kiểm tra độ chính xác của siêu tham số tốt nhất vừa thu được, nhóm tiếp tục đánh giá mô hình trên tập test để xem kết quả cuối cùng ghi nhận được:

	precision	recall	f1-score
0	0.784884	0.721925	0.752089
1	0.514286	0.590164	0.549618
2	0.530000	0.616279	0.569892
3	0.820513	0.842105	0.831169
4	0.771689	0.816425	0.793427
5	0.738095	0.623116	0.675749
6	1.000000	0.900000	0.947368
7	0.782609	0.663158	0.717949
8	0.860927	0.747126	0.800000
9	0.650000	0.382353	0.481481
10	0.855172	0.794872	0.823920
11	0.454545	0.621469	0.525060
12	0.778481	0.763975	0.771160
accuracy	0.708899	0.708899	0.708899
macro avg	0.733939	0.698690	0.710683
weighted avg	0.725484	0.708899	0.713375



Hình 9: Ma trận nhầm lẫn - SVM

b. Maxent Classifier

Nhóm thực hiện điều chỉnh siêu tham số bằng phương pháp Grid Search sử dụng thư viện sklearn (GridSearchCV), với số kfold = 3.

Sau khi khởi tạo mô hình mặc định, nhóm tiến hành điều chỉnh một số các siêu tham số của Maxent Classifier.

Siêu tham số	Giá trị
Phương pháp điều chuẩn ('C')	[0.1, 1.0, 10.0]
Thành phần điều chuẩn ('penalty')	['l1', 'l2', 'elasticnet'*]
Thuật toán tối ưu hóa ('solver')	['lbfgs', 'liblinear','saga'*]
Tỷ lệ l1 ('l1_ratio')*	l1_ratio = 0.5*

Bảng 6. Giá trị tham số được sử dụng trong mô hình Maxent

* **Lưu ý:** Nhằm so sánh các thành phần điều chuẩn, cụ thể là l1, l2, l1 + l2, nhóm chọn giá trị penalty phù hợp. Trong đó, để có được l1 + l2, phải chọn penalty = 'elasticnet' và solver = 'saga'. Vậy nên nhóm thực hiện Grid Search 2

lần cho thuật toán này, trong đó có một lần để giá trị vừa rồi là mặc định. Ngoài ra, solver = 'lbfgs' không được sử dụng với penalty = 'l1'.

Kết quả về thời gian huấn luyện cũng như accuracy cho tập test của các tổ hợp siêu tham số như sau:

Thời gian huấn luyện trung bình	Siêu tham số	Test Accuracy
8.731500	{'C': 0.1, 'penalty': 'elasticnet', 'solver': 'saga'}	0.409279
1.660587311	{'C': 0.1, 'penalty': 'l1', 'solver': 'liblinear'}	0.277323
15.20737934	{'C': 0.1, 'penalty': 'l2', 'solver': 'lbfgs'}	0.675608
0.5186280409	{'C': 0.1, 'penalty': 'l2', 'solver': 'liblinear'}	0.677888
95.636692	{'C': 1.0, 'penalty': 'elasticnet', 'solver': 'saga'}	0.656163
1.364799579	{'C': 1.0, 'penalty': 'l1', 'solver': 'liblinear'}	0.606545
22.15309334	{'C': 1.0, 'penalty': 'l2', 'solver': 'lbfgs'}	0.690896
0.9027745724	{'C': 1.0, 'penalty': 'l2', 'solver': 'liblinear'}	0.694919
4.717586756	{'C': 10.0, 'penalty': 'l1', 'solver': 'liblinear'}	0.617676
24.62988504	{'C': 10.0, 'penalty': 'l2', 'solver': 'lbfgs'}	0.692773
1.884392659	{'C': 10.0, 'penalty': 'l2', 'solver': 'liblinear'}	0.697199
410.910006	{'C': 10.0, 'penalty': 'elasticnet', 'solver': 'saga'}	0.690493

Bảng 7. Kết quả điều chỉnh siêu tham số của mô hình Maxent

Nhận xét:

Có thể thấy, dẫn suất Regularized là l2 mang lại kết quả tương đối tốt hơn các phương pháp khác. Siêu tham số nhóm dùng để huấn luyện ban đầu cũng đem lại kết quả rất tốt, tuy nhiên thời gian huấn luyện trung bình lại kém hơn hai tổ hợp siêu tham số tốt nhất. Ngoài ra, tổ hợp siêu tham số {'C': 10.0, 'penalty': 'elasticnet', 'solver': 'saga'} có thời gian huấn luyện chậm đáng kể so với những tổ hợp còn lại.

c. Mô hình học sâu (LSTM)

Các tham số nhóm điều chỉnh cho mô hình này như sau:

Siêu tham số	Giá trị
Số đơn vị của lớp Bi-LTSM ('lstm_units')	[64, 128, 256]
Hàm tối ưu hóa ('optimizer')	['adam', 'rmsprop', 'sgd']

Bảng 8. Giá trị tham số được sử dụng trong mô hình LSTM

Kết quả về thời gian huấn luyện cũng như accuracy cho tập test của các tổ hợp siêu tham số như sau:

Thời gian huấn luyện trung bình	Siêu tham số	Test Accuracy
437.587810	'lstm_units': 256, 'optimizer': 'rmsprop'	0.4559
439.935480	'lstm_units': 256, 'optimizer': 'adam'	0.6317
475.518242	'lstm_units': 256, 'optimizer': 'sgd'	0.0998
180.581716	'lstm_units': 128, 'optimizer': 'rmsprop'	0.3760
174.628663	'lstm_units': 128, 'optimizer': 'adam'	0.6356
182.881877	'lstm_units': 128, 'optimizer': 'sgd'	0.0943
64.744225	'lstm_units': 64, 'optimizer': 'rmsprop'	0.3192

64.405346	'lstm_units': 64, 'optimizer': 'sgd'	0.1073
77.452629	'lstm_units': 64, 'optimizer': 'adam'	0.6914

Bảng 9. Kết quả điều chỉnh siêu tham số của mô hình LSTM

Nhận xét:

Hàm tối ưu hóa Stochastic Gradient Descent (SGD) thường đem lại accuracy rất thấp (khoảng 10%) so với các hàm khác. Thuật toán tối ưu Adam nhìn chung mang lại kết quả tốt, cùng với 64 đơn vị cho lớp Bi-LSTM, tổ hợp siêu tham số này mang lại accuracy cao nhất, với 0.6914.

6. Đánh giá

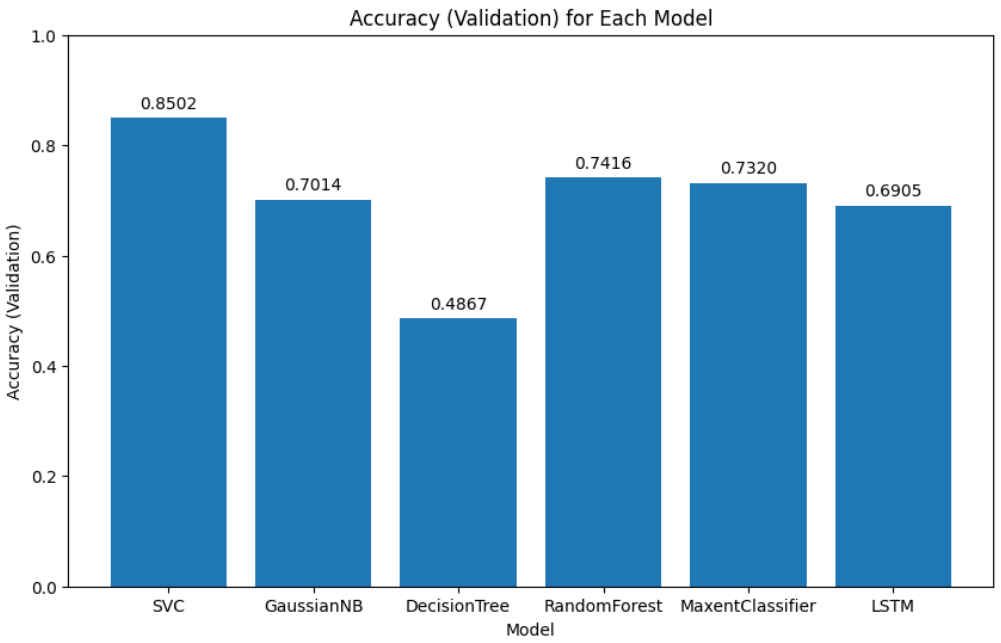
Sau khi lựa chọn các siêu tham số tốt nhất cho các mô hình, nhóm thực hiện so sánh các mô hình này với nhau qua một số tiêu chí. Các tham số nhóm lựa chọn cho từng mô hình như sau:

Mô hình	Giá trị
SVM	{'C': 10, 'kernel': 'linear'}
Naive Bayes	{'var_smoothing': 1e-09}
Decision Tree	{'criterion': 'gini', 'max_depth': 30, 'min_samples_leaf': 4, 'min_samples_split': 2}
Random Forest	{'max_depth': 20, 'min_samples_leaf': 1, 'min_samples_split': 6, 'n_estimators': 150}
Maxent Classifier	{'C': 10.0, 'penalty': 'l2', 'solver': 'liblinear'}
LSTM	'embedding_dim': 50, 'lstm_units': 64, 'epochs': 10, 'batch_size': 32, 'optimizer': 'rmsprop'

Bảng 10. Bảng tham số được sử dụng ban đầu của từng mô hình

a. Về accuracy

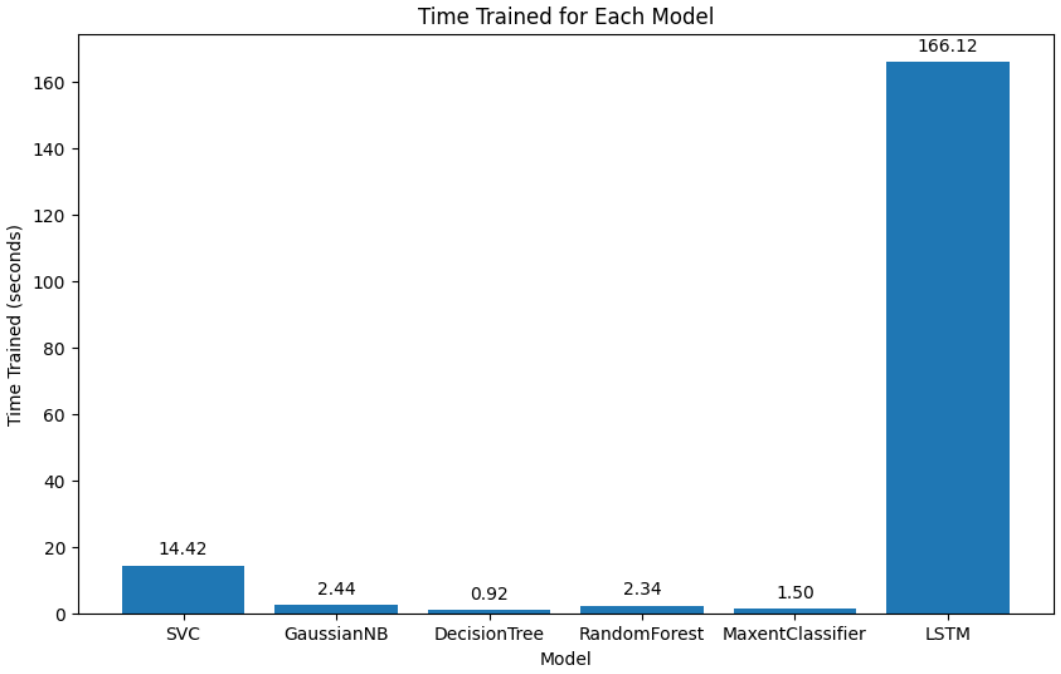
Có thể thấy, mô hình Support Vector Classifier mang lại kết quả vượt trội về accuracy so với các mô hình khác (0.8502). Trong khi đó, Decision Tree là mô hình có kết quả thấp nhất (0.4872).



Hình 10: So sánh về accuracy của các mô hình

b. Về thời gian huấn luyện

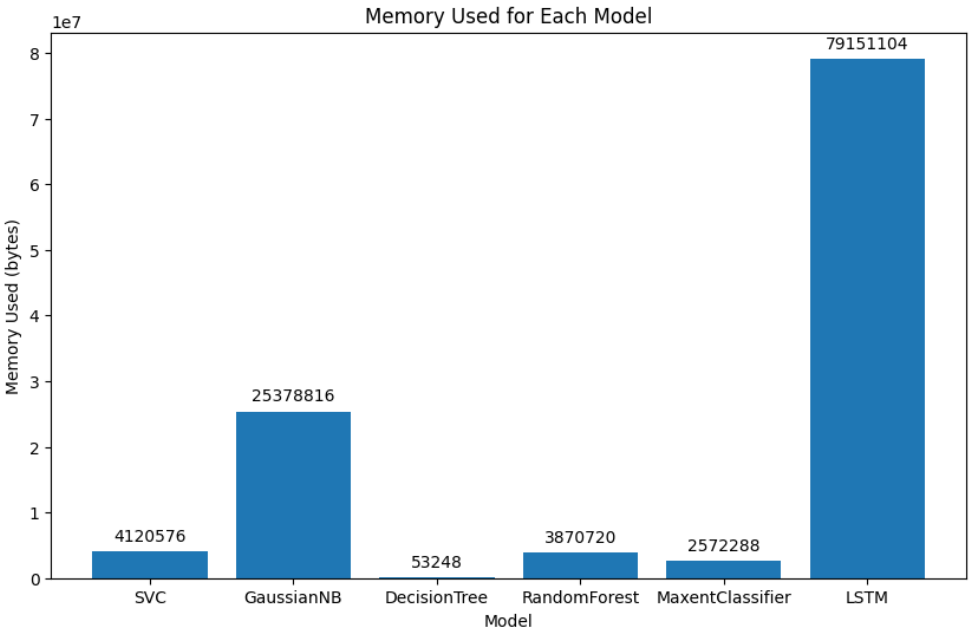
Tuy có hiệu quả cao nhưng tốc độ của Support Vector Classifier có phần nhỉnh hơn so với các mô hình học máy khác. Đối thời mô hình LSTM, thời gian huấn luyện được hiển thị là thời gian trung bình của một epoch, nhưng vẫn lớn đáng kể so với các mô hình khác.



Hình 11: So sánh về thời gian huấn luyện của các mô hình

c. Về bộ nhớ sử dụng

Không chỉ có thời gian chạy lâu hơn, LSTM cũng đặc biệt tiêu tốn về bộ nhớ sử dụng. Tuy có accuracy thấp hơn so với các mô hình còn lại, Decision Tree chiếm ít dung lượng bộ nhớ nhất.



Hình 12: So sánh về bộ nhớ sử dụng của các mô hình

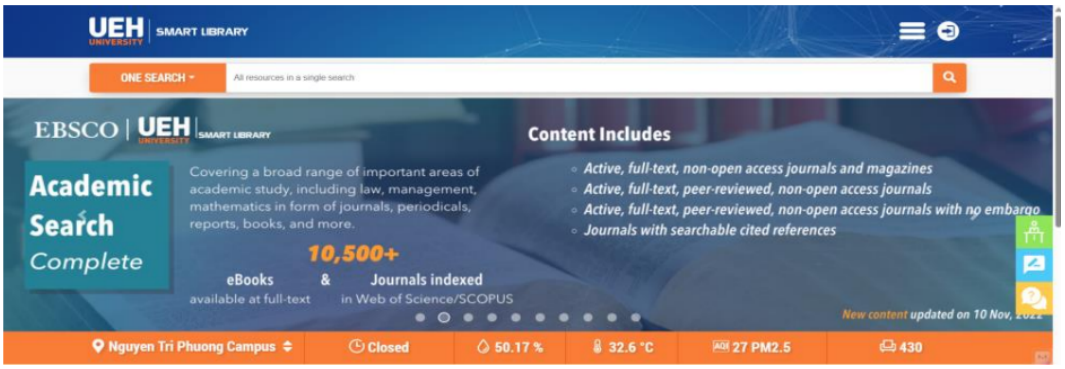
CHƯƠNG V. THIẾT KẾ GIAO DIỆN CHO HỆ THỐNG

1. Xây dựng giao diện cho hệ thống

Trong chương này, nhóm sử dụng Qt Designer và thư viện PyQt5 để thiết kế giao diện cho hệ thống trong môi trường Python.

a. Ý tưởng thiết kế cho hệ thống

Nhóm nghiên cứu áp dụng các phương pháp Máy học để phát triển Hệ thống dự đoán nhãn cho các nội dung tìm kiếm bằng văn bản tiếng Việt, đặc biệt trong môi trường UEH Smart Digital Library của Đại học Kinh tế TP. HCM. Mục tiêu là gia tăng khả năng tiếp cận đúng nhu cầu sách cho người dùng. Giao diện chính của bài nghiên cứu được thiết kế dựa trên UEH Digital Library, nhằm tạo sự thân thiện và quen thuộc cho người dùng, đặc biệt là sinh viên, từ đó nâng cao khả năng tiếp nhận và sử dụng của hệ thống.



Hình 13: UEH Smart Digital Library

b. Nút chức năng

- ***Thanh nhập nội dung cần dự đoán nhãn***

- ***Tab***

- **Chức năng**: Giúp người dùng di chuyển đến các Tab chức năng khác có trong chương trình. Bao gồm ba chức năng chính: Introduction (Giới thiệu bài nghiên cứu, cách sử dụng), Preview (So sánh hiệu quả giữa các mô hình học máy, bao gồm: thời gian và chỉ số đánh giá các siêu tham số) và Predict (Dự đoán nhãn cho văn bản bản tiếng Việt

cần tìm, hiển thị dự đoán của cả ba phương pháp: SVM, Maxent và LSTM).

- ***Insert***

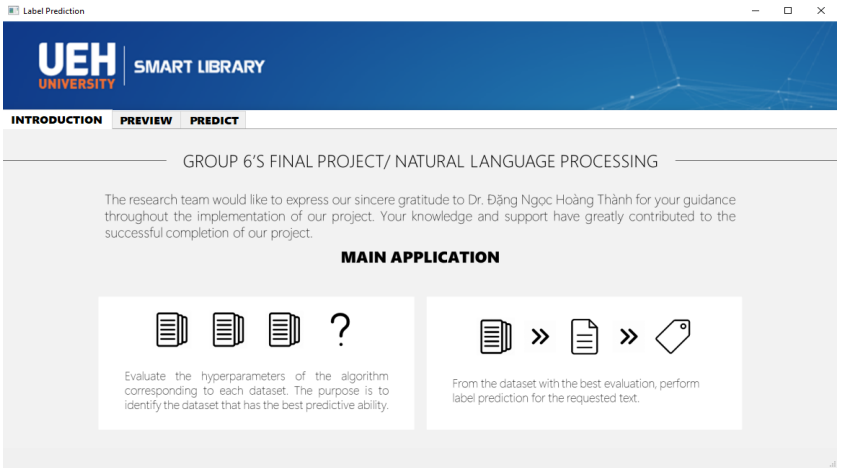
  - **Chức năng**: Khi nhấn nút, người dùng có thể lựa chọn bộ dữ liệu làm tập huấn luyện cho các phương pháp học máy có sẵn, bao gồm: SVM, Maxent và LSTM.
  - **Triển khai**: Hệ thống sẽ đưa ra các chỉ số đánh giá cho các siêu tham số của các mô hình học máy. Đồng thời, hiển thị biểu đồ đường thể hiện tốc độ xử lý của các mô hình qua 10 Epoch.

- ***Find***

  - **Chức năng**: Khi nhấn hút, hệ thống sẽ đưa ra dự đoán nhãn cho văn bản được người dùng nhập trước đó trên thành tìm kiếm.
  - **Triển khai**: Hiển thị nội dung dự đoán bao gồm: Nhãn dự đoán của ba phương pháp học máy vừa nêu, thời gian dự đoán.

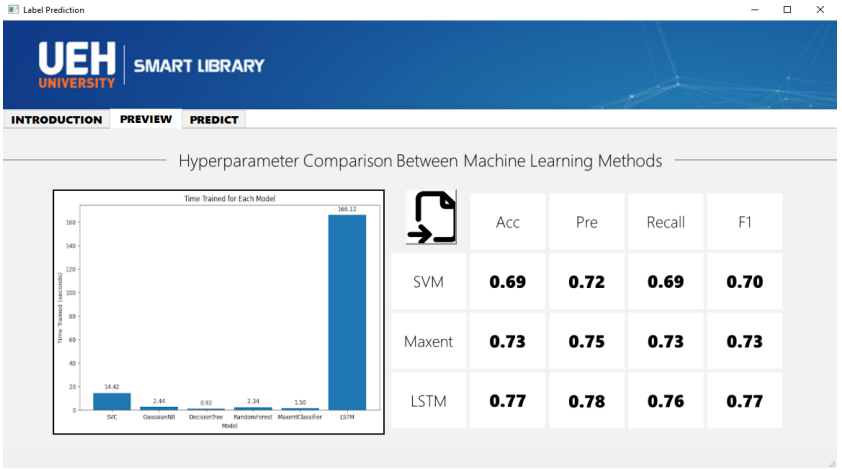
c. Kết quả giao diện

- ***Giao diện Giới thiệu (Introduction)***



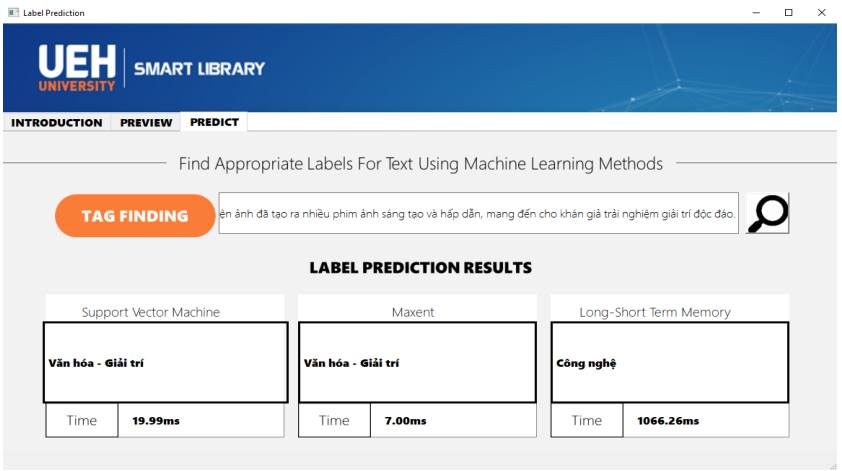
Hình 14. Mô phỏng giao diện Giới thiệu của hệ thống

- ***Giao diện So sánh (Preview)***



Hình 15. Mô phỏng giao diện So sánh của hệ thống

- ***Giao diện Dự đoán (Predict)***



Hình 16. Mô phỏng giao diện Dự đoán của hệ thống

CHƯƠNG VI. KẾT LUẬN

1. Kết quả đạt được

Thông qua quá trình tìm hiểu và nghiên cứu, nhóm đã ứng dụng được những kiến thức đã học được trong lĩnh vực "Xử lý Ngôn Ngữ Tự Nhiên" để thực hiện khai thác và xử lý dữ liệu văn bản. Từ đó, xây dựng một hệ thống dự đoán nhãn cho văn bản tiếng Việt với độ chính xác tương đối cao thông qua việc triển khai không chỉ các mô hình học máy truyền thống như SVM, Naive Bayes, Decision Tree, Random Forest, mà còn tích hợp các mô hình phức tạp hơn như Maxent và LSTM. Việc đa dạng hóa các mô hình giúp nhóm hiểu rõ hơn về hiệu suất của từng phương pháp đối với ngôn ngữ tiếng Việt và tìm ra mô hình phù hợp nhất với bài toán phân loại văn bản.

Sau khi đã huấn luyện thành công các mô hình học máy, nhằm tối ưu hóa hiệu suất của các mô hình, nhóm đã vận dụng phương pháp Grid Search để điều chỉnh siêu tham số và tìm ra các giá trị tối ưu. Kết quả đánh giá hiệu suất của các mô hình trên tập kiểm tra đã được thực hiện bằng cách sử dụng các chỉ số đánh giá chính như độ chính xác, Recall, Precision, F1-score, và ma trận nhầm lẫn.

Ngoài ra, để giới thiệu kết quả và sử dụng hệ thống một cách thuận tiện, nhóm đã triển khai một giao diện ứng dụng sử dụng Qt Designer và thư viện PyQt5. Giao diện này cho phép người dùng nhập văn bản cần dự đoán, đánh giá hiệu suất của mỗi mô hình, và nhận dự đoán nhãn cho văn bản đó.

Tóm lại, đồ án không chỉ giải quyết bài toán Text Classification mà còn đề cập đến nhiều khía cạnh khác nhau của quá trình nghiên cứu và triển khai học máy trong ngôn ngữ tự nhiên, đặc biệt là trong ngữ cảnh tiếng Việt. Thành công của đồ án mở ra những hướng phát triển tiềm năng để nghiên cứu và cải thiện hệ thống trong tương lai.

2. Những hạn chế và hướng phát triển

Bên cạnh những kết quả đạt được, đồ án "Ứng dụng phương pháp học máy trong dự đoán nhãn cho văn bản tiếng Việt" vẫn còn tồn tại một số hạn chế trong quá trình triển khai và thực hiện. Các mô hình học máy truyền thống có thể gặp khó khăn khi phải đối mặt với lượng dữ liệu lớn. Sự đa dạng về từ vựng và ngữ cảnh trong tiếng Việt là một thách thức đối với các mô hình học máy. Các mô hình truyền thống có thể không linh hoạt đủ để hiểu và xử lý sự phong phú và đa dạng của ngôn ngữ, đặc biệt là khi gặp phải từ ngữ đặc biệt và cấu trúc ngữ pháp phức tạp.

Nhóm đã xác định một số hướng phát triển quan trọng để tối ưu hóa và mở rộng tính năng của hệ thống dự đoán nhãn văn bản tiếng Việt sử dụng các mô hình học máy. Trước hết là tăng cường dữ liệu huấn luyện bằng cách thu thập thêm dữ liệu hoặc áp dụng kỹ thuật tăng cường dữ liệu. Có thể mở rộng nguồn dữ liệu bằng cách thu

thập từ nhiều nguồn khác nhau, bao gồm các nguồn tin tức, diễn đàn, blog và văn bản chuyên ngành khác nhau, nhằm giúp mô hình học được nhiều ngữ cảnh và đa dạng văn bản. Điều này giúp cải thiện khả năng học của mô hình đối với các nhãn ít xuất hiện và đối mặt với các trường hợp đa dạng trong ngôn ngữ tiếng Việt. Điều này đồng thời đòi hỏi cần kiểm soát chất lượng của dữ liệu mới để đảm bảo tính chính xác và đáng tin cậy của dữ liệu được thu thập, bao gồm kiểm tra ngữ pháp, ngữ cảnh, và đảm bảo sự đại diện đúng đắn của mọi nhãn.

Bên cạnh đó, đồ án cũng cần phải cải thiện phần giao diện để tăng cường trải nghiệm người dùng. Bằng cách hiển thị đồ thị thống kê, cải thiện tính thẩm mỹ, và thêm các chức năng lọc và sắp xếp, người dùng có thể dễ dàng theo dõi và đánh giá hiệu suất của các mô hình một cách thuận lợi. Chức năng nhập văn bản sẽ được tối ưu hóa để giúp người dùng nhập liệu một cách thuận tiện và nhanh chóng.

PHỤ LỤC

1. Mã nguồn

- Link Github: janettruong/Vietnamese-News-Classification---NLP-SEM5: A project for our NLP class at UEH. (github.com)

2. Bảng phân công

THÀNH VIÊN	PHÂN CÔNG	ĐÁNH GIÁ
Trương Thị Hồng Mai	Lý thuyết Maxent, Viết báo cáo tổng hợp, Điều chỉnh siêu tham số	100%
Võ Minh Nguyên	Lý thuyết LSTM, Xây dựng mã nguồn (Maxent, LSTM), Thiết kế giao diện	100%
Hoàng Đức Dân	Xây dựng mã nguồn (3 phương pháp học máy), Tiền xử lý dữ liệu, Xây dựng giao diện	100%
Nguyễn Thanh Vy	Tổng quan đề tài, Xác định bài toán, Tiền xử lý dữ liệu, EDA, Kết luận	100%

Bảng 11. Bảng phân công & Đánh giá

3. Tài liệu tham khảo

[1] Machine Learning cơ bản (machinelearningcoban.com)

[2] Understanding of LSTM Networks - GeeksforGeeks

[3] Deep Learning using Rectified Linear Units (ReLU) (arxiv.org)

[4] Tổng hợp nguồn tham khảo được cung cấp đầy đủ trong Github