BỘ GIÁO DỤC VÀ ĐÀO TẠO

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ TP. HCM**

# ĐỒ ÁN MÔN HỌC

# PHÁT HIỆN TIN GIẢ TRÊN MẠNG XÃ HỘI

Ngành: **KHOA HỌC DỮ LIỆU**

Chuyên ngành: **KHOA HỌC DỮ LIỆU**


Giảng viên hướng dẫn : TS. Lê Cung Tưởng

Sinh viên thực hiện :

2286400029 – Hồ Gia Thành

2286400015 – Huỳnh Thái Linh

2286400011 – Trương Minh Khoa

Lớp: 22DKHA1


TP. Hồ Chí Minh, 2025

BỘ GIÁO DỤC VÀ ĐÀO TẠO

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ TP. HCM**

# ĐỒ ÁN MÔN HỌC

# PHÁT HIỆN TIN GIẢ TRÊN MẠNG XÃ HỘI

Ngành: **KHOA HỌC DỮ LIỆU**

Chuyên ngành: **KHOA HỌC DỮ LIỆU**


Giảng viên hướng dẫn : TS. Lê Cung Tưởng

Sinh viên thực hiện :

2286400029 – Hồ Gia Thành

2286400015 – Huỳnh Thái Linh

2286400011 – Trương Minh Khoa

Lớp: 22DKHA1


TP. Hồ Chí Minh, 2025

# NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN

. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . . . .

TP. HCM, Ngày.....tháng.....năm 2025

**Giảng viên hướng dẫn**

(Ký tên, đóng dấu)

# LỜI CAM ĐOAN

Chúng tôi, Hồ Gia Thành, Trương Minh Khoa và Huỳnh Thái Linh, xin cam đoan rằng:

Tất cả nội dung của bài báo cáo này là kết quả từ quá trình nghiên cứu và làm việc chung của cả ba chúng tôi. Các thông tin được trình bày trong báo cáo đều được thu thập từ những nguồn đáng tin cậy và đã được xử lý cẩn thận.

Chúng tôi đảm bảo rằng không có bất kỳ hành vi sao chép hay sử dụng thông tin không chính xác nào từ các nguồn khác. Mọi tài liệu tham khảo đã được ghi nguồn rõ ràng và tuân thủ đúng các quy định về trích dẫn học thuật.

Báo cáo này là sản phẩm nghiên cứu độc lập của nhóm và chưa từng được nộp hoặc công bố ở bất kỳ nơi nào trước đây. Chúng tôi hoàn toàn chịu trách nhiệm về tính trung thực và chính xác của toàn bộ nội dung báo cáo.

Chúng tôi hy vọng rằng đề tài này sẽ đóng góp một phần nhỏ vào việc nghiên cứu và ứng dụng các công nghệ trí tuệ nhân tạo trong việc **phát hiện, ngăn chặn và giảm thiểu sự lan truyền của tin giả trên mạng xã hội**, góp phần xây dựng một môi trường thông tin an toàn và đáng tin cậy hơn.

TP. HCM, Ngày.....tháng.....năm 2025

**Sinh viên**

Hồ Gia Thành
Huỳnh Thái Linh
Trương Minh Khoa

# DANH MỤC CÁC KÝ HIỆU, TỪ VIẾT TẮT VÀ TỪ KHÓA

| | |
|---|---|
| **NLP** | Natural Language Processing (Xử lý Ngôn ngữ Tự nhiên). |
| **EDA** | Exploratory Data Analysis (Phân tích Khám phá Dữ liệu). |
| **BERT** | Bidirectional Encoder Representations from Transformers (Biểu diễn Mã hóa Hai chiều từ Mô hình Transformer). |
| **TF-IDF** | Term Frequency-Inverse Document Frequency (Tần suất Thuật ngữ - Tần suất Ngược Tài liệu). |
| **LSTM** | Long Short-Term Memory (Bộ nhớ Dài-Ngắn Hạn). |
| **CV** | Cross-Validation (Kiểm định chéo). |
| **LR** | Logistic Regression (Hồi quy Logistic). |
| **DT** | Decision Tree (Cây Quyết định). |
| **RF** | Random Forest (Rừng ngẫu nhiên). |
| **XGB** | eXtreme Gradient Boosting (Tăng cường Gradient Cực đại). |

# Mục lục

# Danh sách hình vẽ

# Chương 1

# TỔNG QUAN

## 1.1 Giới thiệu đề tài

Trong bối cảnh thông tin lan truyền nhanh trên các nền tảng mạng xã hội, tin giả trở thành vấn đề nghiêm trọng ảnh hưởng đến nhận thức và niềm tin của cộng đồng. Đề tài hướng đến việc xây dựng mô hình AI có khả năng tự động phát hiện và phân loại tin giả dựa trên phân tích nội dung văn bản hoặc hình ảnh, ứng dụng các kỹ thuật học sâu và xử lý ngôn ngữ tự nhiên (NLP)[1]. Kết quả nghiên cứu góp phần hỗ trợ kiểm chứng thông tin, hạn chế tác động tiêu cực của tin giả và nâng cao độ tin cậy của môi trường truyền thông số.

## 1.2 Nhiệm vụ của đồ án

Nhiệm vụ của đồ án là nghiên cứu và phát triển một mô hình trí tuệ nhân tạo có khả năng phát hiện và phân loại tin giả trên mạng xã hội dựa trên việc phân tích nội dung văn bản. Quá trình thực hiện bao gồm tìm hiểu lý thuyết nền tảng, thu thập và xử lý dữ liệu phù hợp, xây dựng và huấn luyện mô hình, sau đó đánh giá hiệu quả và đề xuất hướng cải thiện. Kết quả của đồ án hướng đến việc tạo ra một giải pháp ứng dụng thực tiễn, góp phần hỗ trợ kiểm chứng thông tin và hạn chế sự lan truyền của tin giả trong không gian mạng.

### 1.2.1 Tính cấp thiết của đề tài

Trong kỷ nguyên của Internet và mạng xã hội, thông tin được lan truyền với tốc độ chóng mặt và khả năng tiếp cận toàn cầu. Tuy nhiên, mặt trái của sự kết nối này là sự bùng nổ của tin giả (Fake News) – những thông tin sai lệch hoặc bịa đặt được ngụy tạo tinh vi dưới dạng tin tức chính thống. Tin giả không chỉ đơn thuần là sự nhầm lẫn; nó là một vấn đề nghiêm trọng, đe dọa trực tiếp đến sự ổn định xã hội, niềm tin cộng đồng, và thậm chí là an ninh quốc gia.

[1]Natural Language Processing: Kỹ thuật giúp máy hiểu và xử lý ngôn ngữ con người.

Tin giả có khả năng thao túng dư luận trong các cuộc bầu cử, gây hoang mang trong các cuộc khủng hoảng sức khỏe (như đại dịch), và làm suy giảm uy tín của các tổ chức báo chí chân chính. Nếu không có cơ chế lọc và phát hiện tự động, chúng ta sẽ phải đối mặt với một "đại dịch thông tin" (infodemic), khiến công chúng khó lòng phân biệt được đâu là sự thật, đâu là lời dối trá. Do đó, việc nghiên cứu và phát triển một hệ thống tự động, chính xác, có khả năng phân loại tin tức theo thời gian thực không chỉ là một nhu cầu học thuật mà còn là một yêu cầu cấp bách của đời sống hiện đại.

### 1.2.2 Ý nghĩa khoa học và thực tiễn của đề tài

**Ý nghĩa khoa học:** Đề tài góp phần vào lĩnh vực Xử lý Ngôn ngữ Tự nhiên và Học máy thông qua việc đánh giá, so sánh các phương pháp biểu diễn văn bản truyền thống và hiện đại, đặc biệt là khai thác sức mạnh của mô hình BERT và kiến trúc lai BERT-LSTM trong phát hiện tin giả, từ đó mở ra hướng nghiên cứu mới về ứng dụng mô hình lai trong phân loại văn bản.

**Ý nghĩa thực tiễn:** Đề tài mang lại một mô hình phát hiện tin giả có tính ứng dụng cao, có thể hỗ trợ kiểm chứng và sàng lọc thông tin trên mạng xã hội. Hệ thống giúp tự động hóa quy trình kiểm duyệt, giảm thiểu chi phí và thời gian, đồng thời nâng cao độ tin cậy và chất lượng thông tin trong môi trường truyền thông số.

## 1.3 Mục tiêu

### 1.3.1 Mục tiêu tổng quan

Mục tiêu tổng quan của đề tài là nghiên cứu và phát triển mô hình trí tuệ nhân tạo có khả năng phát hiện và phân loại tin giả trên mạng xã hội dựa trên phân tích nội dung văn bản. Thông qua việc ứng dụng các phương pháp học máy và học sâu, đề tài hướng đến xây dựng một hệ thống tự động nhận diện tin giả, góp phần nâng cao độ tin cậy và hỗ trợ kiểm chứng thông tin trong môi trường truyền thông số.

### 1.3.2 Mục tiêu cụ thể

Với đề tài này chúng tôi tìm hiểu và ứng dụng các kỹ thuật xử lý ngôn ngữ tự nhiên và học sâu để xây dựng mô hình trí tuệ nhân tạo phát hiện tin giả một cách chính xác và hiệu quả. Đề tài tập trung vào việc tiền xử lý và chuẩn hóa dữ liệu, thiết kế và huấn luyện mô hình phân loại, đồng thời đánh giá, so sánh hiệu suất giữa các phương pháp biểu diễn và mô hình khác nhau. Từ đó, đề xuất giải pháp tối ưu giúp nâng cao độ chính xác trong phát hiện tin giả và khả năng ứng dụng thực tế trong kiểm chứng thông tin trên mạng xã hội.

## 1.4 Đối tượng và phạm vi

### 1.4.1 Đối tượng

Đối tượng nghiên cứu của đề tài là các phương pháp và mô hình trí tuệ nhân tạo được ứng dụng trong phát hiện và phân loại tin giả, đặc biệt là những mô hình xử lý ngôn ngữ tự nhiên (NLP) và học sâu (Deep Learning) có khả năng phân tích và hiểu ngữ nghĩa văn bản. Bên cạnh đó, đề tài cũng tập trung vào khai thác và so sánh hiệu quả của các kỹ thuật biểu diễn văn bản như TF-IDF, Word2Vec và BERT trong việc mô tả đặc trưng nội dung tin tức.

### 1.4.2 Phạm vi

Phạm vi của đề tài tập trung vào phát hiện tin giả dựa trên nội dung văn bản với đối tượng chính là tiêu đề (title) của các bài báo bằng ngôn ngữ tiếng Anh, chứ không phải toàn bộ nội dung. Phạm vi dữ liệu được giới hạn trong các mẫu tin đã được kiểm chứng từ hai tập dữ liệu công khai, uy tín là *GossipCop và PolitiFact*, không bao gồm các yếu tố đa phương tiện khác như âm thanh, video hoặc bình luận người dùng. Nghiên cứu được thực hiện trong phạm vi học thuật, với mục tiêu đánh giá khả năng áp dụng các mô hình học máy và học sâu trong bài toán phân loại tin giả, làm nền tảng cho các hướng mở rộng trong tương lai.

## 1.5 Phương pháp nghiên cứu

### 1.5.1 Phương pháp nghiên cứu sơ bộ

Ban đầu, chúng tôi tiến hành thu thập và hợp nhất hai bộ dữ liệu thô là *GossipCop* và *PolitiFact* từ nguồn Github. Chúng tôi xác định bài toán là phân loại nhị phân (thật/giả) và giới hạn phạm vi nghiên cứu chỉ trong tiêu đề tin tức, đồng thời kiểm tra sơ bộ về cấu trúc và các trường dữ liệu có sẵn.

### 1.5.2 Phương pháp nghiên cứu tài liệu

Chúng tôi đã tham khảo các công trình khoa học liên quan đến phát hiện tin giả và xử lý ngôn ngữ tự nhiên. Việc này giúp chúng tôi lựa chọn các phương pháp luận đã được chứng minh: sử dụng các kỹ thuật undersampling (TomekLinks, NearMiss) để xử lý mất cân bằng, và đặc biệt là chọn BERT làm công cụ trích xuất đặc trưng chính do khả năng nắm bắt ngữ cảnh vượt trội so với TF-IDF hay Word2Vec.

### 1.5.3 Phương pháp nghiên cứu thống kê

Chúng tôi áp dụng phương pháp thống kê mô tả (EDA) để khám phá sâu về dữ liệu. Bằng cách sử dụng các thư viện như matplotlib và seaborn, chúng tôi đã trực quan hóa sự mất cân bằng nghiêm trọng của dữ liệu, kiểm tra giá trị thiếu (missing values), và phân tích tần suất từ (WordCloud, bar plots) để tìm ra các đặc điểm từ vựng khác biệt giữa tin thật và tin giả.

### 1.5.4 Phương pháp thực nghiệm

Đây là phương pháp trọng tâm. Chúng tôi xây dựng một quy trình thử nghiệm có kiểm soát: (1) Tiền xử lý văn bản đồng nhất; (2) Áp dụng các kỹ thuật undersampling; (3) Vector hóa bằng BERT; (4) Huấn luyện đồng loạt 5 mô hình (Logistic Regression, Decision Tree, Random Forest, XGBoost, LSTM) trên cùng một bộ dữ liệu. Sau đó, chúng tôi tiến hành tinh chỉnh (fine-tuning) có hệ thống qua GridSearchCV và lặp thủ công (cho LSTM) để tìm ra cấu hình tối ưu.

### 1.5.5 Phương pháp đánh giá

Để đảm bảo kết quả khách quan, chúng tôi sử dụng phương pháp Đánh giá chéo K-Fold (K=5) cho các mô hình học máy để kiểm tra sự ổn định. Hiệu suất của tất cả mô hình cuối cùng được đo lường trên tập dữ liệu kiểm tra (test set) độc lập. Chúng tôi sử dụng bộ chỉ số tiêu chuẩn (Accuracy, Precision, Recall, F1-Score) và trực quan hóa kết quả bằng Ma trận nhầm lẫn (Confusion Matrix) và biểu đồ so sánh, cho phép đưa ra kết luận chính xác về mô hình hiệu quả nhất.

## 1.6 Những đóng góp nghiên cứu của đề tài

Đề tài mang lại đóng góp cả về mặt khoa học và thực tiễn trong lĩnh vực phát hiện tin giả. Về khoa học, chúng tôi đánh giá và so sánh hiệu quả của các phương pháp biểu diễn văn bản và mô hình học sâu, qua đó làm rõ khả năng ứng dụng của mô hình lai trong xử lý ngôn ngữ tự nhiên. Về thực tiễn, đề tài xây dựng một mô hình phát hiện tin giả có tính ứng dụng cao, có thể hỗ trợ kiểm chứng thông tin và giảm thiểu sự lan truyền của tin sai lệch trên mạng xã hội.

# Chương 2

# CƠ SỞ LÝ THUYẾT

## 2.1 Tổng quan về bài toán phát hiện tin giả

### 2.1.1 Khái niệm về tin giả (Fake News)

Tin giả (fake news) là thông tin sai lệch hoặc gây hiểu nhầm, được trình bày dưới dạng tin tức thật nhằm đánh lừa người đọc. Theo Encyclopaedia Britannica, đó là "false or deceptive stories presented as legitimate news content" [1]. Một số nghiên cứu còn phân biệt misinformation (sai lệch không cố ý) và disinformation (sai lệch có chủ ý) [2]. Việc xác định rõ khái niệm là nền tảng để xây dựng hệ thống phát hiện hiệu quả.

### 2.1.2 Ảnh hưởng và thách thức của tin giả trên mạng xã hội

Tin giả lan truyền nhanh trên mạng xã hội, gây ảnh hưởng đến nhận thức cộng đồng, làm suy giảm niềm tin vào truyền thông và có thể dẫn đến hậu quả xã hội nghiêm trọng [2]. Các thách thức chính gồm: tốc độ lan truyền vượt kiểm soát; nội dung cảm xúc, giật gân dễ thu hút; khó khăn trong xác minh tính xác thực. Do đó, phát hiện tin giả vừa là vấn đề kỹ thuật vừa là thách thức xã hội.

### 2.1.3 Các hướng tiếp cận phát hiện tin giả bằng trí tuệ nhân tạo

Các phương pháp tiếp cận chính [3]:

1. **Dựa trên nội dung (Content-based):**
   - Phân tích đặc trưng ngôn ngữ (linguistic features) của bài viết.
   - Sử dụng các mô hình từ truyền thống (SVM, TF-IDF) đến các mô hình học sâu hiện đại (CNN, LSTM, và các mô hình Transformer như BERT) để nắm bắt ngữ nghĩa.
2. **Dựa trên ngữ cảnh (Context-based):**

- Phân tích các yếu tố bên ngoài văn bản, như mô hình lan truyền (propagation patterns) hoặc độ tin cậy của người dùng (user credibility).
- Thường sử dụng Mạng nơ-ron đồ thị (GNN) để mô hình hóa sự lan truyền.

**3. Phương pháp lai (Hybrid):** Kết hợp cả hai hướng tiếp cận trên để tăng hiệu quả.

### 2.1.4 Đặc điểm dữ liệu văn bản ngắn (title) trong phát hiện tin giả

Phát hiện tin giả qua tiêu đề (title) là hướng tiếp cận phổ biến do độ lan truyền cao trên mạng xã hội. Tuy nhiên, tiêu đề thường ngắn, thiếu ngữ cảnh, và mang tính cảm xúc hoặc giật gân nên khó phân tích ngữ nghĩa chính xác [4]. Nghiên cứu cho thấy việc kết hợp các đặc trưng như tương quan tiêu đề–nội dung, cảm xúc và metadata giúp cải thiện hiệu quả mô hình github.com.

## 2.2 Xử lý Ngôn ngữ Tự nhiên (NLP)

### 2.2.1 Khái niệm và vai trò của NLP

Xử lý Ngôn ngữ Tự nhiên (NLP) là một lĩnh vực của Trí tuệ Nhân tạo, cung cấp các phương pháp tính toán để máy tính có thể xử lý, phân tích và "hiểu" ngôn ngữ con người [5]. Trong bối cảnh luận văn này, NLP là nền tảng kỹ thuật bắt buộc để chuyển đổi các tiêu đề tin tức (văn bản phi cấu trúc) sang định dạng vectơ số có cấu trúc, từ đó làm đầu vào cho các mô hình học máy. [6] Trong bài toán phát hiện tin giả, vai trò của NLP là (1) Tiền xử lý văn bản để loại bỏ nhiễu và (2) Trích xuất đặc trưng để biểu diễn ý nghĩa ngữ nghĩa của văn bản dưới dạng số học.

### 2.2.2 Các bước xử lý văn bản trong NLP

Văn bản thô (raw text) luôn chứa "nhiễu" (ví dụ: chữ hoa/thường, dấu câu, từ dừng). Quá trình Tiền xử lý (Preprocessing) là bước bắt buộc để chuẩn hóa dữ liệu, cải thiện hiệu suất mô hình. Các bước chính bao gồm [7]:

- **Chuẩn hóa và Làm sạch (Normalization & Cleaning):** [8]
  - **Lowercasing:** Đồng nhất văn bản về chữ thường.
  - **Removing Punctuation/Special Characters:** Loại bỏ các ký tự không mang ngữ nghĩa
- **Tokenization (Tách từ):** Bước cơ bản nhất, tách một câu thành các đơn vị nhỏ hơn (tokens), thường là các từ.

- **Stopword Removal (Loại bỏ Stopwords):** Xóa các từ xuất hiện thường xuyên nhưng không mang nhiều ý nghĩa (ví dụ: "là", "của", "và") để giảm chiều dữ liệu.
- **Stemming & Lemmatization:** Đưa các từ biến thể (ví dụ: "chạy", "đang chạy") về một dạng gốc chung (ví dụ: "chạy") để mô hình không coi chúng là các từ khác nhau.

### 2.2.3 Ứng dụng NLP trong bài toán phân loại văn bản

Trong phát hiện tin giả, NLP được dùng để:

- Biểu diễn văn bản bằng các kỹ thuật như Bag-of-Words, TF-IDF, Word2Vec hoặc BERT embedding.
- Khai thác đặc trưng ngôn ngữ (cảm xúc, cấu trúc, n-gram, độ tương quan tiêu đề-nội dung).
- Huấn luyện các mô hình như Naive Bayes, SVM, Random Forest, CNN, BERT để phân loại tin tức thật – giả. [9]

## 2.3 Các phương pháp biểu diễn đặc trưng văn bản

### 2.3.1 TF-IDF (Term Frequency – Inverse Document Frequency)

TF-IDF (Term Frequency–Inverse Document Frequency) là một phương pháp phổ biến trong xử lý ngôn ngữ tự nhiên (NLP) nhằm xác định mức độ quan trọng của một từ trong văn bản so với toàn bộ tập dữ liệu [10].
Trọng số TF-IDF được tính bằng tích của *tần suất từ* (Term Frequency - TF) và *nghịch đảo tần suất tài liệu* (Inverse Document Frequency - IDF), như công thức sau:

$$TF\text{-}IDF(t, d, D) = TF(t, d) \times \log \frac{|D|}{|\{d \in D : t \in d\}|} \tag{2.1}$$

Trong đó:

- $TF(t, d)$ là tần suất xuất hiện của từ $t$ trong văn bản $d$;
- $|D|$ là tổng số văn bản trong tập dữ liệu;
- $|\{d \in D : t \in d\}|$ là số lượng văn bản có chứa từ $t$.

Phương pháp này giúp giảm ảnh hưởng của các từ phổ biến (như “là”, “và”, “của”), đồng thời nhấn mạnh các từ mang tính phân biệt cao trong từng văn bản. TF-IDF được ứng dụng rộng rãi trong các bài toán *phân loại văn bản*, *truy xuất thông tin* và *phân cụm tài liệu* [11].

### 2.3.2 Word2Vec

Word2Vec biểu diễn mỗi từ dưới dạng vector liên tục (word embedding) phản ánh ngữ nghĩa và quan hệ giữa các từ.
Hai kiến trúc chính là Skip-gram (dự đoán từ ngữ cảnh từ từ trung tâm) và CBOW (dự đoán từ trung tâm từ ngữ cảnh).
Nhờ học từ dữ liệu lớn, Word2Vec có thể nắm bắt các mối quan hệ ngữ nghĩa như:

$$\vec{king} - \vec{man} + \vec{woman} \approx \vec{queen} \tag{2.2}$$

Phương pháp này được triển khai hiệu quả trong thư viện Gensim [12].

### 2.3.3 BERT

BERT là mô hình học sâu dựa trên kiến trúc Transformer encoder, cho phép mô hình hiểu ngữ cảnh hai chiều (bidirectional) của câu.
BERT được huấn luyện trước với hai nhiệm vụ: Masked Language Modeling và Next Sentence Prediction, giúp tạo ra biểu diễn ngữ cảnh hóa (contextual embeddings) cho từ và câu [13].

Trong bài toán phân loại văn bản (như phát hiện tin giả), vector biểu diễn [CLS] từ BERT thường được sử dụng làm đặc trưng đầu vào cho mô hình học máy, giúp cải thiện độ chính xác so với TF-IDF hay Word2Vec.

## 2.4 Xử lý mất cân bằng dữ liệu (Imbalanced Data)

### 2.4.1 Giới thiệu về vấn đề mất cân bằng dữ liệu

Mất cân bằng dữ liệu (data imbalance) xảy ra khi một hoặc nhiều lớp trong tập dữ liệu có số lượng mẫu vượt trội so với các lớp khác, gây sai lệch trong quá trình huấn luyện mô hình học máy. Mô hình có xu hướng dự đoán theo lớp chiếm đa số, dẫn đến hiệu suất thấp đối với lớp thiểu số [14]. Đây là một thách thức phổ biến trong các bài toán như phát hiện gian lận, chẩn đoán y khoa và phát hiện tin giả, nơi dữ liệu "thật" thường áp đảo dữ liệu "giả".

### 2.4.2 Các kỹ thuật Undersampling (Giảm thiểu mẫu đa số)

Undersampling là nhóm phương pháp nhằm giảm số lượng mẫu của lớp đa số để tạo ra tập dữ liệu cân bằng hơn. Mục tiêu là duy trì tính đại diện của dữ liệu trong khi giảm độ chênh lệch giữa các lớp.

**a) Tomek Links:** Tomek Links được giới thiệu bởi Tomek (1976) [15] nhằm loại bỏ các cặp mẫu gần nhau nhưng thuộc hai lớp khác nhau. Nếu

một cặp như vậy được phát hiện, mẫu thuộc lớp đa số sẽ bị loại bỏ. Cách này giúp làm sạch ranh giới phân lớp và giảm nhiễu trong dữ liệu [16].

**b) NearMiss:** NearMiss là thuật toán lựa chọn một tập con của lớp đa số dựa trên khoảng cách giữa các điểm của lớp đa số và lớp thiểu số. Có ba biến thể chính: NearMiss-1, NearMiss-2 và NearMiss-3, khác nhau ở cách đo và chọn các điểm gần nhất. Phương pháp này giúp cân bằng dữ liệu mà vẫn giữ được các mẫu quan trọng [17].

## 2.5 Các mô hình học máy và học sâu

### 2.5.1 Các mô hình học máy

#### a) Hồi quy Logistic (Logistic Regression)

Mặc dù có tên là "hồi quy", Hồi quy Logistic thực chất là một mô hình phân loại nhị phân (binary classification) tuyến tính và hiệu quả [18].
Thay vì dự đoán một giá trị liên tục, nó dự đoán xác suất (probability) để một mẫu đầu vào thuộc về một lớp cụ thể (ví dụ: "Tin giả"). Để làm điều này, nó sử dụng hàm sigmoid (hay còn gọi là hàm logistic).
Hàm sigmoid nhận một giá trị đầu vào $z$ (là một tổ hợp tuyến tính của các đặc trưng đầu vào) và "ép" giá trị đó vào một khoảng $(0, 1)$:

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

Giá trị đầu ra $\sigma(z)$ này được diễn giải là xác suất. Một ngưỡng (threshold) thường là 0.5 được chọn: nếu xác suất $> 0.5$, mẫu được gán nhãn "Tin giả" (Lớp 1); ngược lại, nó được gán nhãn "Tin thật" (Lớp 0) [19].

#### b) Cây quyết định (Decision Tree)

Cây quyết định là một mô hình phi tuyến tính, trực quan, mô phỏng quá trình ra quyết định của con người dưới dạng một cấu trúc "if-then-else" [20].
Cách thức phân chia dữ liệu của nó như sau:

**1. Chọn đặc trưng tốt nhất:** ắt đầu từ nút gốc (root node), thuật toán tìm ra đặc trưng (ví dụ: sự xuất hiện của từ "sốc") và một ngưỡng giá trị để chia tập dữ liệu thành hai tập con.

**2. Đo lường độ "tinh khiết":** Việc lựa chọn này dựa trên một tiêu chí đo lường, chẳng hạn như *Gini Impurity* hoặc *Information Gain (Entropy)*. Mục tiêu là tạo ra các tập con "tinh khiết" nhất có thể (tức là các tập con chứa phần lớn các mẫu từ chỉ một lớp).

**3. Đệ quy:** Quá trình này được lặp lại (đệ quy) cho mỗi tập con (tạo ra các nhánh và nút con), cho đến khi đạt đến một điều kiện dừng (ví dụ: cây đạt độ sâu tối đa, hoặc các nút lá đã đủ "tinh khiết").

#### c) Rừng ngẫu nhiên (Random Forest)

Rừng ngẫu nhiên là một phương pháp học tập tổ hợp (ensemble learning) nhằm cải thiện hiệu suất và giải quyết vấn đề lớn nhất của một cây quyết định đơn lẻ: *overfitting* (học quá khớp) [21].
Nó "kết hợp nhiều cây quyết định" bằng hai kỹ thuật chính:

**1. Bagging (Bootstrap Aggregating):** Thuật toán tạo ra $N$ cây quyết định độc lập. Mỗi cây được huấn luyện trên một mẫu *bootstrap* (một tập dữ liệu con được lấy ngẫu nhiên, có lặp lại, từ tập huấn luyện gốc).

**2. Random Feature Subspace:** Khi xây dựng mỗi cây, tại mỗi nút, thay vì xem xét tất cả các đặc trưng để tìm ra phép chia tốt nhất, cây chỉ được phép chọn từ một tập con ngẫu nhiên của các đặc trưng.

Kết quả cuối cùng được quyết định bằng "bỏ phiếu" (majority vote) từ tất cả các cây trong rừng. Sự ngẫu nhiên này giúp tạo ra các cây không bị tương quan (uncorrelated), làm cho mô hình tổng thể trở nên mạnh mẽ và tổng quát hóa tốt hơn.

#### d) XGBoost (Extreme Gradient Boosting)

XGBoost cũng là một mô hình ensemble, nhưng thuộc họ *Boosting* (Tăng cường), khác với Bagging của Random Forest [22].

**Boosting:** Thay vì xây dựng các cây song song và độc lập, các thuật toán boosting xây dựng các cây một cách tuần tự *(sequentially)*. Cây thứ hai được huấn luyện để sửa chữa các lỗi mà cây thứ nhất mắc phải. Cây thứ ba sửa lỗi của cây thứ hai, và cứ thế tiếp diễn.

**Gradient Boosting:** Cụ thể, mô hình học các "lỗi" bằng cách sử dụng phương pháp tối ưu hóa Gradient Descent. Mỗi cây mới được huấn luyện trên phần dư (residuals) của mô hình trước đó.

**Extreme (XGBoost):** XGBoost là một triển khai tối ưu hóa cao của Gradient Boosting. Nó nổi bật nhờ tốc độ vượt trội (do xử lý song song và tối ưu hóa phần cứng) và khả năng kiểm soát overfitting hiệu quả thông qua việc tích hợp sẵn điều chuẩn hóa *(Regularization)* L1 và L2 vào hàm mục tiêu.

### 2.5.2 Mô hình học sâu

**Mạng LSTM (Long Short-Term Memory)**

Văn bản (như một tiêu đề) về bản chất là một chuỗi *(sequence)*. Mạng Nơ-ron Hồi quy (RNN) được thiết kế để xử lý chuỗi, nhưng các RNN đơn giản gặp phải vấn đề *vanishing gradient* (gradient tiêu biến), khiến chúng "quên" mất thông tin ở các bước thời gian (time steps) xa (ví dụ: quên từ đầu tiên của một câu dài).
LSTM là một kiến trúc RNN đặc biệt được thiết kế để giải quyết vấn đề này [23].

- **Kiến trúc:** LSTM duy trì một "trạng thái tế bào" (cell state) $C_t$, hoạt động giống như một "băng chuyền" thông tin, cho phép thông tin quan trọng chạy dọc theo chuỗi mà ít bị thay đổi.
- **Các cổng (Gates):** Khả năng ghi nhớ/quên của LSTM được điều khiển bởi ba "cổng" (gates) — là các lớp nơ-ron nhỏ (thường dùng hàm sigmoid) quyết định xem bao nhiêu thông tin được đi qua:

  **1. Forget Gate (Cổng Quên):** Quyết định thông tin nào cần loại bỏ khỏi trạng thái tế bào $C_{t-1}$ của bước trước.

  **2. Input Gate (Cổng Đầu vào):** Quyết định thông tin mới nào (từ đầu vào hiện tại $x_t$) cần được thêm vào trạng thái tế bào.

  **3. Output Gate (Cổng Đầu ra):** Quyết định xem trạng thái tế bào hiện tại $C_t$ sẽ được sử dụng để tính toán đầu ra (hidden state $h_t$) của bước này như thế nào.

Kiến trúc này cho phép LSTM nắm bắt các phụ thuộc dài hạn, một yếu tố quan trọng để hiểu ngữ nghĩa của các tiêu đề tin tức.

## 2.6 Đánh giá và tối ưu hóa mô hình

### 2.6.1 Các độ đo đánh giá (Evaluation Metrics)

Trong các bài toán phân loại (classification) của học máy và học sâu, việc đánh giá hiệu năng mô hình là bước thiết yếu để đảm bảo mô hình không chỉ học tốt trên dữ liệu huấn luyện mà còn có khả năng tổng quát hóa tốt với dữ liệu mới. Dưới đây trình bày các độ đo phổ biến [24]:

#### Accuracy

Accuracy (độ chính xác chung) được định nghĩa là tỷ lệ mẫu được phân loại đúng trên tổng số mẫu xét tới:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.3}$$

Trong đó:

- $TP$ (True Positive): số mẫu dương tính được dự đoán đúng;
- $TN$ (True Negative): số mẫu âm tính được dự đoán đúng;
- $FP$ (False Positive): số mẫu âm bị dự đoán nhầm thành dương;
- $FN$ (False Negative): số mẫu dương bị dự đoán nhầm thành âm.

Mặc dù dễ hiểu và hay sử dụng, accuracy có thể gây hiểu nhầm khi dữ liệu bị mất cân bằng mạnh – ví dụ nếu lớp đa số chiếm tỷ lệ rất lớn thì mô hình "luôn dự đoán lớp đa số" có thể cho accuracy cao nhưng lại không thực sự phân loại tốt lớp thiểu số. [25]

#### Precision (Độ chính xác của dự đoán dương)

Precision được định nghĩa là tỷ lệ các mẫu dự đoán là dương (positive) mà thực sự là dương:

$$Precision = \frac{TP}{TP + FP} \tag{2.4}$$

Precision phản ánh khả năng "khi mô hình nói dương, thì đúng bao nhiêu" [26, 27].

#### Recall

Recall (hay còn gọi sensitivity) là tỷ lệ các mẫu thực sự dương mà mô hình dự đoán đúng:

$$Recall = \frac{TP}{TP + FN} \tag{2.5}$$

Recall phản ánh khả năng "mô hình bắt được hết bao nhiêu mẫu dương thực sự" [24, 26, 27].

#### F1-Score

F1-score là trung bình hài hòa (harmonic mean) của precision và recall:

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{2.6}$$

F1 giúp cân bằng giữa precision và recall, đặc biệt hữu ích khi dữ liệu mất cân bằng hoặc khi ưu tiên cả hai yếu tố: không bỏ sót mẫu dương (high recall) và không có quá nhiều dự đoán dương sai (high precision) [24, 27].

### 2.6.2 Kỹ thuật kiểm định chéo (Cross-Validation)

Cross-validation (kiểm định chéo) là thủ tục "chia dữ liệu thành nhiều phần (fold)", huấn luyện mô hình trên một phần và đánh giá trên phần còn lại, lặp lại cho tất cả các fold, sau đó lấy trung bình kết quả để ước tính khả năng tổng quát hóa của mô hình. [28, 29]

#### Stratified K-Fold

Đối với bài toán phân loại – đặc biệt khi các lớp không cân bằng – việc giữ nguyên tỷ lệ phân bố lớp trong mỗi fold là rất quan trọng. Đó là lý do xuất hiện kỹ thuật *Stratified K-Fold*: mỗi fold sẽ có phân phối các lớp (positive/negative) tương đương với phân phối của toàn bộ tập dữ liệu.
Việc này giúp tránh trường hợp một fold không chứa hoặc chứa quá ít mẫu của lớp thiểu số, dẫn tới đánh giá mô hình thiếu công bằng hoặc thiên lệch [30].

### 2.6.3 Tối ưu hóa siêu tham số (Hyperparameter Tuning)

Hyperparameters là các tham số của mô hình mà không được học trực tiếp từ dữ liệu huấn luyện (không phải là w, b trong hồi quy) mà được đặt trước quá trình huấn luyện và có ảnh hưởng lớn đến hiệu năng mô hình. [31, 32]

#### GridSearchCV

Một trong những phương pháp phổ biến để tìm bộ hyperparameter "tốt nhất" là GridSearchCV: phương pháp tìm kiếm toàn diện (exhaustive search) trên một lưới (grid) các giá trị hyperparameter đã định sẵn và sử dụng cross-validation để đánh giá mỗi tổ hợp. [33]
Cách thức hoạt động:

- Xác định một từ điển hyperparameters với các giá trị thử nghiệm.
- Với mỗi tổ hợp giá trị, sử dụng cross-validation (ví dụ Stratified K-Fold) để huấn luyện và đánh giá mô hình.
- Chọn tổ hợp cho kết quả tốt nhất (theo metric đã chọn) [34].

Phương pháp này giúp tìm ra cấu hình hyperparameter phù hợp với dữ liệu cụ thể, giảm khả năng overfitting hoặc underfitting, nâng cao khả năng chung hóa của mô hình [33].

# Chương 3

# PHƯƠNG PHÁP THỰC NGHIỆM

Chương này trình bày chi tiết về phương pháp luận mà chúng tôi đã áp dụng để giải quyết bài toán phát hiện tin giả. Quy trình của chúng tôi bao gồm các bước: giới thiệu và thu thập dữ liệu, phân tích dữ liệu khám phá (EDA), tiền xử lý văn bản, xử lý mất cân bằng dữ liệu, trích xuất đặc trưng, và cuối cùng là huấn luyện, tinh chỉnh và đánh giá các mô hình học máy.

## 3.1 Giới thiệu về nguồn dữ liệu

Để thực hiện nghiên cứu này, chúng tôi sử dụng bộ dữ liệu FakeNewsNet: *This is a dataset for fake news detection research* được công bố bởi `KaiDMML` trên nền tảng Github. Đây là một tập hợp các bài báo và thông tin liên quan được thu thập từ hai trang web kiểm chứng thông tin lớn là *GossipCop* và *PolitiFact*, và được dán nhãn là Tin giả (Fake News) hoặc Tin thật (Real News).

Tập dữ liệu này cung cấp một nguồn tài nguyên quan trọng để phát triển và đánh giá các mô hình phát hiện tin giả, bao gồm các biến thể tin tức đa phương tiện. Trong phạm vi nghiên cứu này, chúng tôi tập trung khai thác trường Tiêu đề (title) của các bài báo để thực hiện phân loại.

#### Cấu trúc dữ liệu

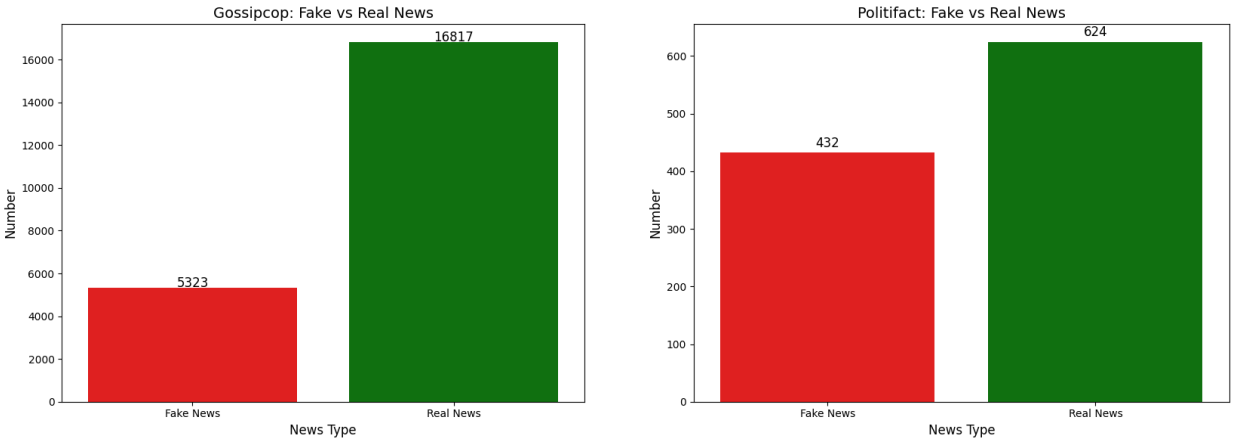
Bộ dữ liệu được phân chia thành 4 tệp CSV chính:

- `gossipcop_fake.csv`
- `gossipcop_real.csv`
- `politifact_fake.csv`
- `politifact_real.csv`

## 3.2 Phân tích khám phá dữ liệu (EDA)

Sau khi đọc dữ liệu, chúng tôi thực hiện phân tích khám phá (EDA) để nắm bắt các thông tin cơ bản của bộ dữ liệu.

- **Kiểm tra tổng quan:** Chúng tôi đã kiểm tra các thông tin cơ bản (thông qua `.info()`), thống kê mô tả (`.describe()`), và tỷ lệ giá trị bị thiếu (`.isnull().sum()`) của cả bốn bộ dữ liệu.
- **Phân bố nhãn:** Chúng tôi trực quan hóa số lượng tin "fake" và "real" cho cả *GossipCop* và *PolitiFact*. Phân tích này cho thấy sự mất cân bằng dữ liệu rõ rệt ở cả hai nguồn, với số lượng tin thật (real) chiếm đa số đáng kể so với tin giả (fake).




(a) GossipCop (b) PolitiFact

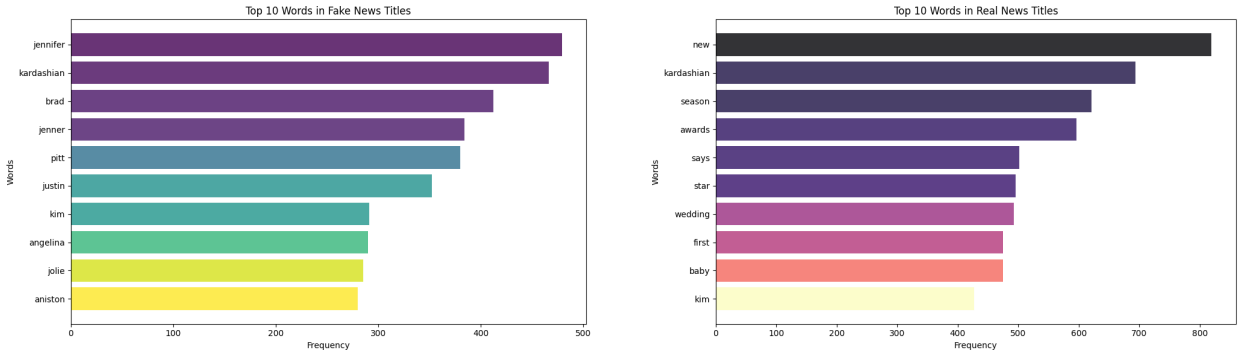
Hình 3.1: Phân bố nhãn của GossipCop và PolitiFact

- **Phân tích từ vựng:**
  - **WordCloud:** Chúng tôi đã tạo WordCloud cho các tiêu đề tin giả và tin thật để quan sát các từ khóa nổi bật nhất trong mỗi nhóm.


(a) WordCloud for Fake News Titles (b) WordCloud for Real News Titles

Hình 3.2: WordCloud của Fake và Real

– **Top-N Words:** Chúng tôi trích xuất 10 từ có tần suất xuất hiện cao nhất (sau khi loại bỏ stop words) và vẽ biểu đồ cột để so sánh các từ khóa đặc trưng giữa hai nhãn.




(a) Top 10 Words in Fake News Titles

(b) Top 10 Words in Real News Titles

Hình 3.3: Top 10 Words của Fake và Real

## 3.3 Tiền xử lý dữ liệu

Quy trình tiền xử lý được thiết kế để làm sạch và chuẩn hóa dữ liệu văn bản, chuẩn bị cho bước trích xuất đặc trưng.

### 3.3.1 Gộp dữ liệu và gán nhãn:

Chúng tôi gộp các tiêu đề từ `gossipcop_fake` và `politifact_fake` thành một nhóm và gán nhãn là 0. Tương tự, chúng tôi gộp `gossipcop_real` và `politifact_real` và gán nhãn là 1.

### 3.3.2 Làm sạch văn bản:

Mỗi tiêu đề đều trải qua một quy trình làm sạch `(preprocess_text)` bao gồm các bước:

- Chuyển đổi thành chữ thường (to_lowercase)
- Loại bỏ tất cả các ký tự số `(remove_numbers)`.
- Loại bỏ dấu câu `(remove_punctuation)`.
- Xóa các khoảng trắng thừa `(remove_extra_spaces)`.
- Tách từ (Tokenization) bằng `word_tokenize` của NLTK.
- Loại bỏ các từ dừng `(Stopwords)` trong tiếng Anh.
- Chuẩn hóa từ `(Stemming)` bằng thuật toán `PorterStemmer`.

## 3.4 Phân chia tập dữ liệu và xử lý mất cân bằng

Để đảm bảo tính khách quan của mô hình, chúng tôi thực hiện các bước phân chia và cân bằng dữ liệu một cách cẩn thận.

- **1. Phân chia Train/Test:** Chúng tôi chia toàn bộ dữ liệu đã xử lý thành hai tập: 80% cho huấn luyện `(X_train, y_train)` và 20% cho kiểm thử `(X_test, y_test)`. Chúng tôi sử dụng tham số `stratify=y` để đảm bảo tỷ lệ phân bố giữa các lớp "fake" và "real" trong tập huấn luyện và tập kiểm thử là tương đồng với tập dữ liệu gốc.
- **2. Xử lý mất cân bằng (Trên tập Train):** Do EDA cho thấy dữ liệu mất cân bằng nghiêm trọng, chúng tôi đã áp dụng chiến lược *undersampling* kết hợp trên chỉ tập huấn luyện để tránh rò rỉ thông tin sang tập kiểm thử.
  - **Vector hóa tạm thời:** Chúng tôi sử dụng `TfidfVectorizer` để chuyển đổi văn bản thành vector.
  - **Bước 1: Tomek Links `(TomekLinks)`:** Chúng tôi áp dụng `Tomek Links` để loại bỏ các mẫu nhiễu hoặc nằm trên ranh giới phân loại thuộc lớp đa số.
  - **Bước 2: NearMiss`NearMiss`:** Sau khi làm sạch bằng `Tomek Links`, chúng tôi tiếp tục sử dụng `NearMiss` để giảm số lượng mẫu của lớp đa số (tin thật) cho đến khi số lượng mẫu của hai lớp trở nên cân bằng. Tập dữ liệu `(X_train_resampled_nm, y_train_resampled_nm)` là kết quả cuối cùng cho việc huấn luyện.

Bảng 3.1: Phân bố lớp trước và sau khi xử lý mất cân bằng dữ liệu

| **Lớp** | **Số lượng (Trước khi xử lý)** | **Số lượng (Sau khi xử lý)** |
|---|---|---|
| 0 (Fake) | 4,604 | 4,604 |
| 1 (Real) | 13,952 | 4,604 |
| **Tổng** | **18,556** | **9,208** |

## 3.5 Trích xuất đặc trưng (Vectorization)

Sau khi có tập huấn luyện đã được cân bằng, chúng tôi tiến hành trích xuất đặc trưng bằng ba phương pháp khác nhau để so sánh hiệu quả.

**1. TF-IDF:** Chúng tôi sử dụng `TfidfVectorizer` (với `max_features=5000`) trên tập huấn luyện đã xử lý. Sau đó, chúng tôi áp dụng `StandardScaler` (với `with_mean=False`) để chuẩn hóa dữ liệu.

**2. Word2Vec:** Chúng tôi huấn luyện một mô hình `Word2Vec` của Gensim (với `vector_size=100`) trên tập huấn luyện. Đặc trưng của mỗi tiêu đề được biểu diễn bằng cách lấy vector trung bình của tất cả các từ trong tiêu đề đó. Dữ liệu này sau đó được chuẩn hóa bằng `StandardScaler`.

**3. BERT:** Chúng tôi sử dụng mô hình `bert-base-uncased` đã được huấn luyện trước. Mỗi tiêu đề (đã được làm sạch) được đưa qua mô hình BERT, và chúng tôi lấy *vector trung bình* của `last_hidden_state` (kích thước 768) làm đặc trưng đại diện cho tiêu đề đó. Đây là phương pháp trích xuất đặc trưng chính được sử dụng trong các thí nghiệm so sánh mô hình.

## 3.6 Huấn luyện mô hình

Chúng tôi tập trung vào việc sử dụng các đặc trưng BERT (768 chiều) làm đầu vào cho các mô hình.

### 3.6.1 Các mô hình cơ sở

Chúng tôi đã triển khai 5 mô hình khác nhau:

1. Logistic Regression (LR)
2. Decision Tree (DT)
3. Random Forest (RF)
4. XGBoost (XGB
5. LSTM (BERT-LSTM): Một mô hình học sâu tùy chỉnh `(BERT_LSTM)` bao gồm các lớp LSTM hai chiều (bidirectional) và một lớp fully-connected, nhận đầu vào là đặc trưng BERT.

### 3.6.2 Đánh giá chéo (Cross-Validation)

Đối với 4 mô hình học máy cổ điển (LR, DT, RF, XGB), chúng tôi sử dụng kỹ thuật *Đánh giá chéo 5-fold (5-Fold Stratified Cross-Validation)* trên tập huấn luyện đã cân bằng `(X_train_bert, y_train_resampled)`.
Để đánh giá trên tập kiểm thử `(X_test_bert)`, chúng tôi đã tạo một hàm dự đoán `ensemble_predict`. Hàm này lấy trung bình xác suất dự đoán từ cả 5 mô hình được huấn luyện trong 5 fold, sau đó đưa ra dự đoán cuối cùng. Điều này giúp tăng tính ổn định và giảm phương sai của kết quả.
Đối với mô hình LSTM, chúng tôi chia tập huấn luyện thành 80% train và 20% validation để huấn luyện và đánh giá trên tập test.

### 3.6.3 Tinh chỉnh siêu tham số (Hyperparameter Tuning)

Để tối ưu hóa hiệu suất, chúng tôi đã tiến hành tinh chỉnh siêu tham số cho tất cả các mô hình trên tập huấn luyện sử dụng đặc trưng BERT.

- **Với LR, DT, RF, XGB:** Chúng tôi sử dụng *GridSearchCV* với đánh giá chéo 5-fold để tự động tìm kiếm bộ tham số tốt nhất từ một không gian tham số (`param_grid`) được định nghĩa trước.
- **Với LSTM:** Chúng tôi thực hiện một quy trình tìm kiếm thủ công (`tune_and_save_lstm`) qua một lưới các tham số (`lstm_grid`), bao gồm `hidden_dim, num_layers, learning_rate`, v.v. Mô hình tốt nhất được chọn dựa trên độ chính xác (accuracy) trên tập validation.

Các mô hình với bộ tham số tốt nhất đã được lưu lại và sử dụng để đánh giá hiệu suất cuối cùng trên tập kiểm thử `X_test`.

### 3.6.4 Các chỉ số đánh giá

Để đánh giá hiệu quả của các mô hình, chúng tôi đã sử dụng các chỉ số (metrics) phân loại tiêu chuẩn:

- **Accuracy:** Tỷ lệ dự đoán đúng trên tổng số mẫu.
- **Precision:** Tỷ lệ các mẫu được dự đoán là "Positive" (Real) thực sự là "Positive".
- **Recall:** Tỷ lệ các mẫu "Positive" (Real) trong thực tế được mô hình dự đoán đúng.
- **F1-Score:** Trung bình điều hòa của Precision và Recall.

Chúng tôi cũng sử dụng `classification_report` để xem chi tiết các chỉ số cho từng lớp và `confusion_matrix` (Ma trận nhầm lẫn) để trực quan hóa hiệu suất của mô hình trong việc phân biệt giữa tin giả và tin thật

# Chương 4

# KẾT QUẢ THỰC NGHIỆM

## 4.1 Đặc trưng hóa văn bản (Vectorization)

Chúng tôi đã thử nghiệm ba phương pháp trích xuất đặc trưng khác nhau từ dữ liệu văn bản đã được làm sạch. Dữ liệu vector hóa sau đó được chuẩn hóa bằng `StandardScaler`.

- TF-IDF
- Word2Vec
- BERT

Để đánh giá sơ bộ, chúng tôi đã huấn luyện mô hình Logistic Regression trên cả ba loại đặc trưng này.

Bảng 4.1: Hiệu suất Logistic Regression với các phương pháp biểu diễn đặc trưng

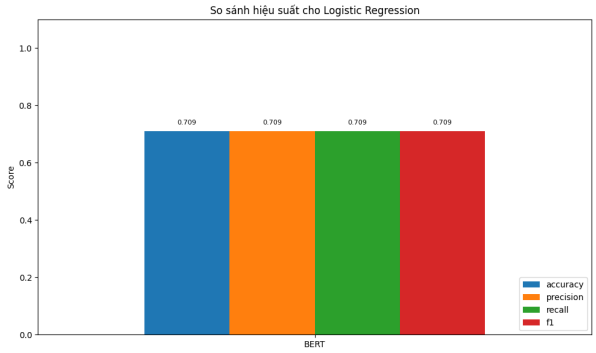
| **Phương pháp** | **Lớp** | **Precision** | **Recall** | **F1-Score** |
|---|---|---|---|---|
| TF-IDF | 0 (Fake) | 0.67 | 0.70 | 0.69 |
| | 1 (Real) | 0.69 | 0.66 | 0.67 |
| | **Accuracy** | | | 0.68 |
| Word2Vec | 0 (Fake) | 0.74 | 0.64 | 0.68 |
| | 1 (Real) | 0.68 | 0.77 | 0.72 |
| | **Accuracy** | | | 0.71 |
| BERT | 0 (Fake) | 0.70 | 0.72 | 0.71 |
| | 1 (Real) | 0.71 | 0.70 | 0.70 |
| | **Accuracy** | | | 0.71 |

Kết quả sơ bộ cho thấy đặc trưng BERT (`X_train_bert`) mang lại hiệu suất vượt trội, đạt được 0.71 accuracy và F1-Score là 0.70 cho lớp 1 (Real)). Do đó, trong các thí nghiệm tiếp theo, chúng tôi quyết định tập trung sử dụng đặc trưng *BERT (768 chiều)* làm đầu vào cho tất cả các mô hình học máy.

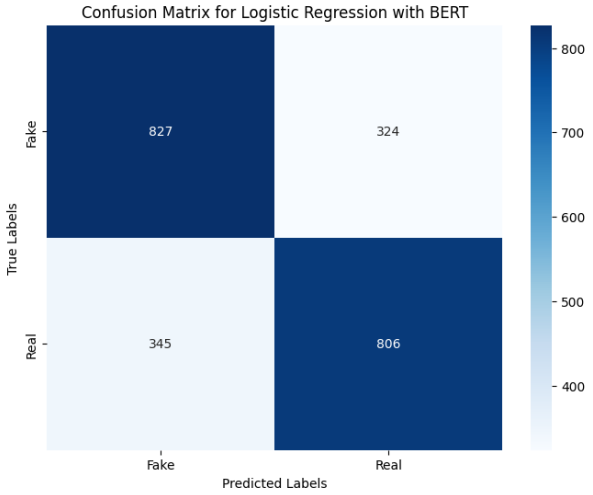
## 4.2 Kết quả mô hình

Kết quả đánh giá của 5 mô hình trên tập kiểm tra (Test set) với cấu hình mặc định (trước khi tinh chỉnh) được trình bày dưới đây.

### 4.2.1 Logistic Regression



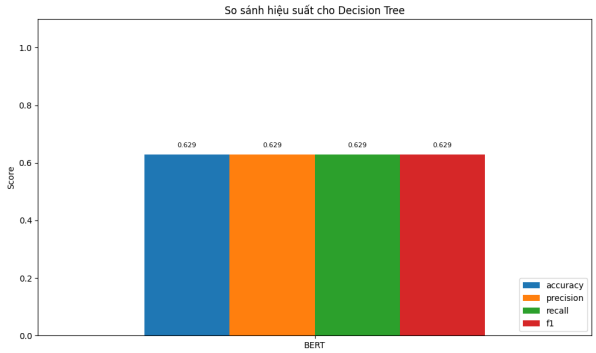

(a) Classification Report



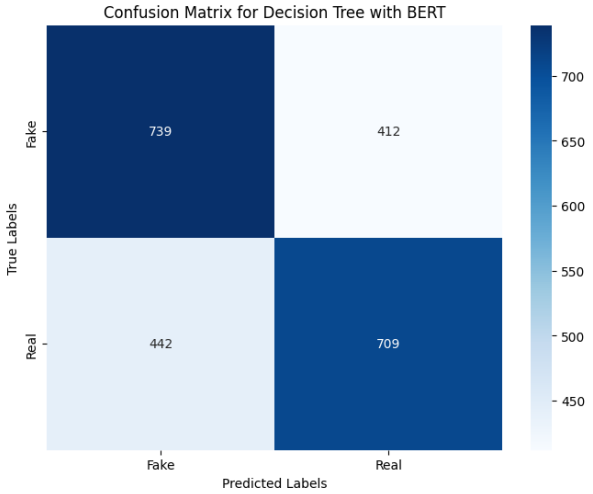

(b) Confusion Matrix

Hình 4.1: Kết quả của mô hình Logistic Regression

### 4.2.2 Decision Tree



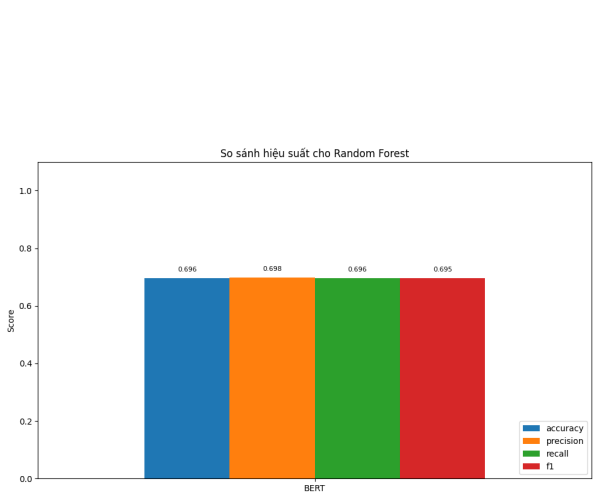

(a) Classification Report



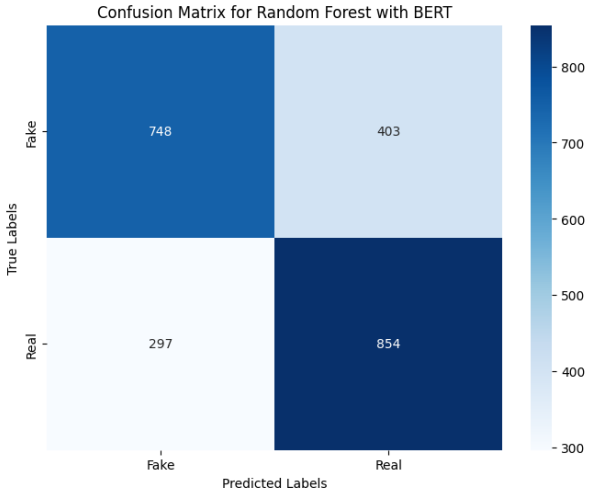

(b) Confusion Matrix

Hình 4.2: Kết quả của mô hình Decision Tree

### 4.2.3 Random Forest



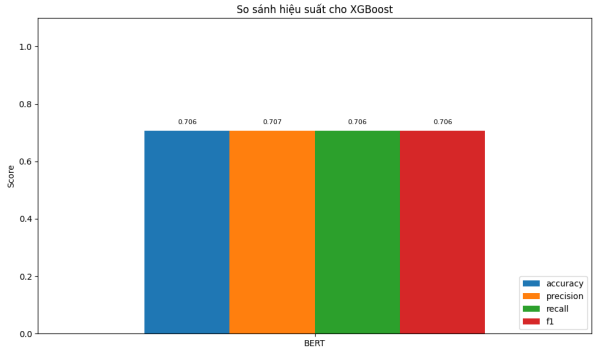

(a) Classification Report



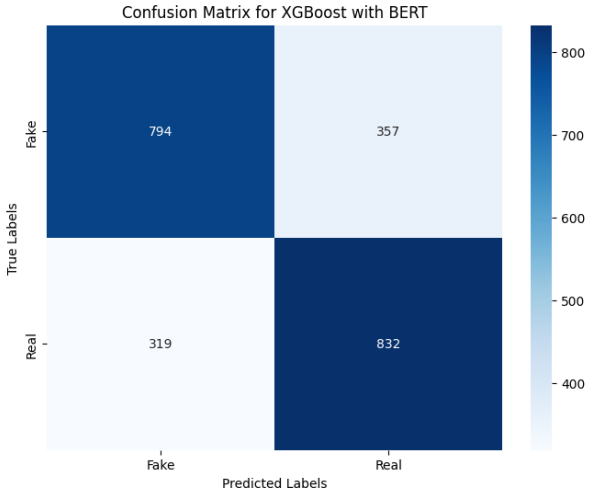

(b) Confusion Matrix

Hình 4.3: Kết quả của mô hình Random Forest

### 4.2.4 XGBoost



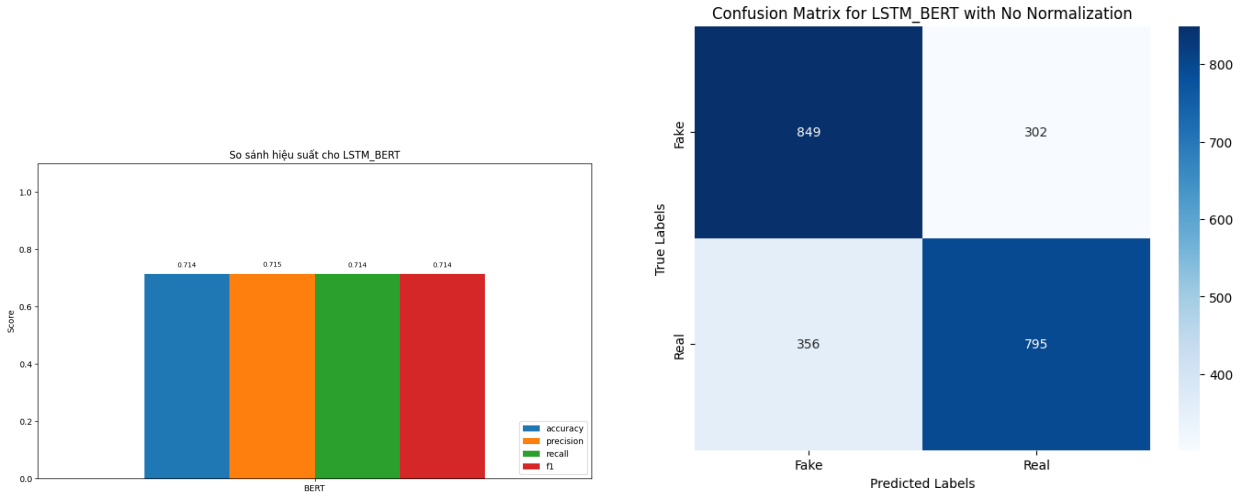

(a) Classification Report




(b) Confusion Matrix

Hình 4.4: Kết quả của mô hình XGBoost

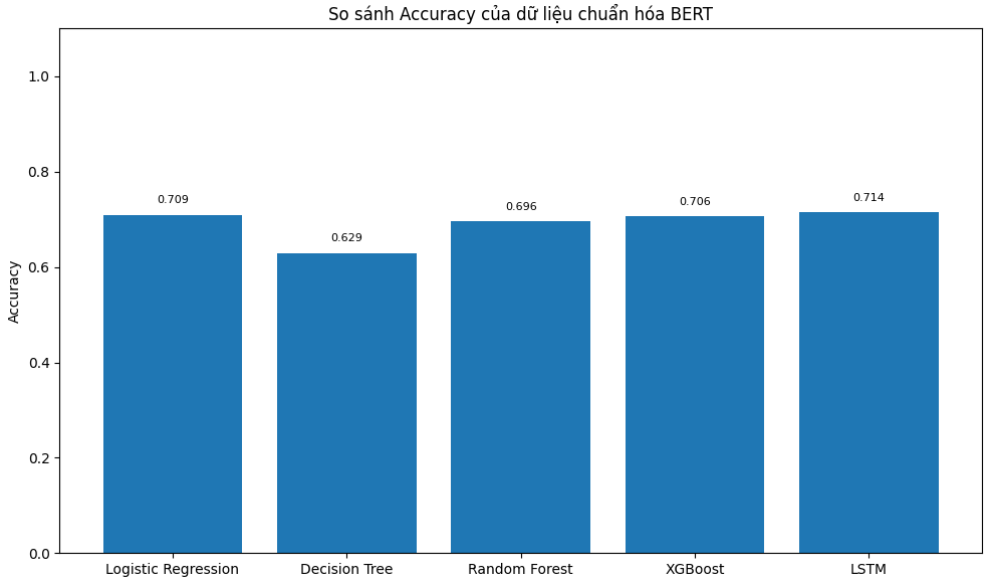
### 4.2.5 LSTM




(a) Classification Report (b) Confusion Matrix

Hình 4.5: Kết quả của mô hình LSTM

Để so sánh tổng quan, chúng tôi trực quan hóa độ chính xác (Accuracy) của cả 5 mô hình.




Hình 4.6: Hiệu suất của 5 mô hình

Biểu đồ trên thể hiện sự so sánh độ chính xác (Accuracy) của năm mô hình học máy gồm Logistic Regression, Decision Tree, Random Forest, XGBoost và LSTM trên dữ liệu văn bản đã được chuẩn hóa và biểu diễn bằng embedding từ BERT. Kết quả cho thấy các mô hình đạt Accuracy trong khoảng từ *0.629* đến *0.714*, cho thấy đặc trưng đầu ra của BERT mang thông tin ngữ nghĩa tốt và giúp các mô hình học được mối quan hệ giữa các đặc trưng một cách ổn định. Trong đó, **LSTM** có độ chính xác cao nhất *(0.714)*, nhờ khả năng tận

dụng thông tin ngữ cảnh tuần tự trong embedding. *Logistic Regression (0.709)* và *XGBoost (0.706)* đạt kết quả tương đương, cho thấy đặc trưng BERT có khả năng phân tách tuyến tính khá rõ. *Random Forest (0.696)* và *Decision Tree (0.629)* cho kết quả thấp hơn, có thể do chưa được tối ưu siêu tham số hoặc chưa thích ứng tốt với không gian đặc trưng có chiều cao. Nhìn chung, kết quả này cho thấy dữ liệu sau khi được biểu diễn bằng BERT đã hỗ trợ tốt cho nhiều loại mô hình khác nhau, tạo nền tảng để tiếp tục cải thiện hiệu năng qua các bước tinh chỉnh siêu tham số (`Grid Search`) ở gian bước tiếp theo.

## 4.3 Tinh chỉnh siêu tham số (Hyperparameter Tuning)

Để tối ưu hóa hiệu suất, chúng tôi đã tiến hành tinh chỉnh siêu tham số cho cả 5 mô hình.

- **LR, DT, RF, XGB:** Chúng tôi sử dụng GridSearchCV với `5-fold cross-validation`, tối ưu hóa theo độ đo accuracy trên tập huấn luyện. Không gian tìm kiếm (`param_grid`) được định nghĩa chi tiết trong code.
- **LSTM:** Chúng tôi thực hiện tìm kiếm thủ công (`manual grid` search) qua một danh sách các cấu hình (`lstm_grid`) bao gồm các thay đổi về `hidden_dim, epochs, batch _size, lr, dropout,` và `num_layers`. Mô hình tốt nhất được chọn dựa trên validation accuracy.

Các siêu tham số tốt nhất tìm được cho mỗi mô hình đã được lưu lại để sử dụng trong đánh giá cuối cùng.

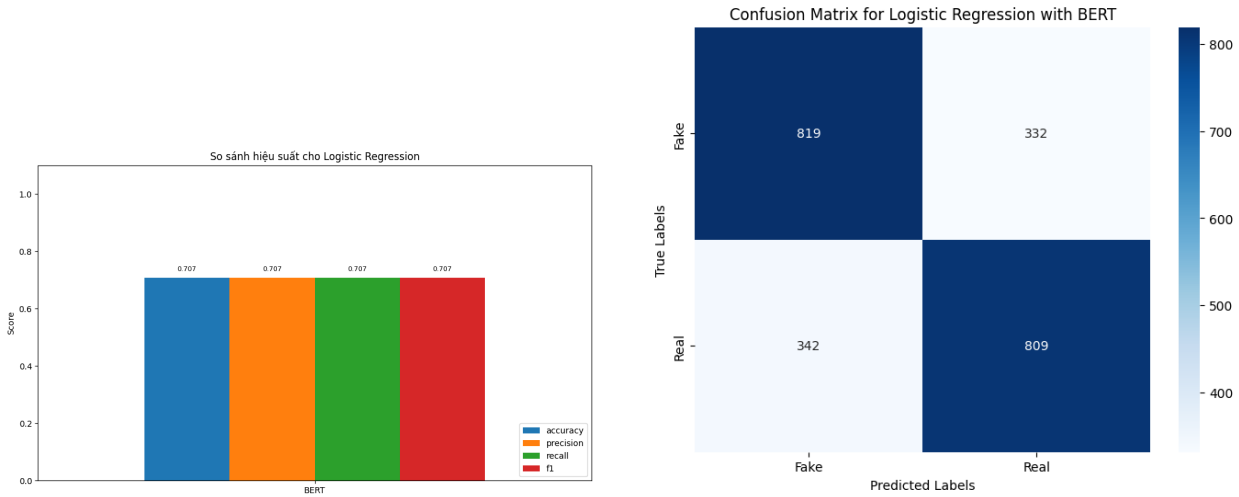
Bảng 4.2: Các siêu tham số tốt nhất (Best Params) cho 5 mô hình với BERT

| **Mô hình** | **Chi tiết Best Params** |
|---|---|
| Random Forest | `{'bootstrap': True, 'ccp_alpha': 0.0, 'class_weight': None, 'max_depth': 30, 'max_features': 0.8, 'min_samples_leaf': 2, 'min_samples_split': 5, 'n_estimators': 500, 'random_state': 42}` |
| Decision Tree | `{'ccp_alpha': 0.0, 'criterion': 'gini', 'max_depth': 10, 'max_features': None, 'min_samples_leaf': 1, 'min_samples_split': 5, 'random_state': 42}` |
| Logistic Regression | `{'C': 0.1, 'class_weight': None, 'max_iter': 100, 'penalty': 'l2', 'random_state': 42, 'solver': 'saga'}` |
| XGBoost | `{'colsample_bytree': 0.8, 'eval_metric': 'logloss', 'learning_rate': 0.05, 'max_depth': 7, 'min_child_weight': 3, 'n_estimators': 400, 'random_state': 42, 'reg_alpha': 0.1, 'reg_lambda': 1.5, 'scale_pos_weight': 1, 'subsample': 0.8}` |
| LSTM | `{'hidden_dim': 256, 'epochs': 15, 'batch_size': 32, 'lr': 0.0005, 'dropout': 0.5, 'num_layers': 2}` |

## 4.4 Đánh giá mô hình sau tinh chỉnh

Chúng tôi đã huấn luyện lại các mô hình bằng cách sử dụng các siêu tham số tối ưu đã tìm thấy ở bước 4.3. Phương pháp đánh giá (5-fold ensemble cho sklearn, train/val split cho LSTM) được giữ nguyên. Kết quả đánh giá cuối cùng trên tập kiểm tra (Test set) như sau:

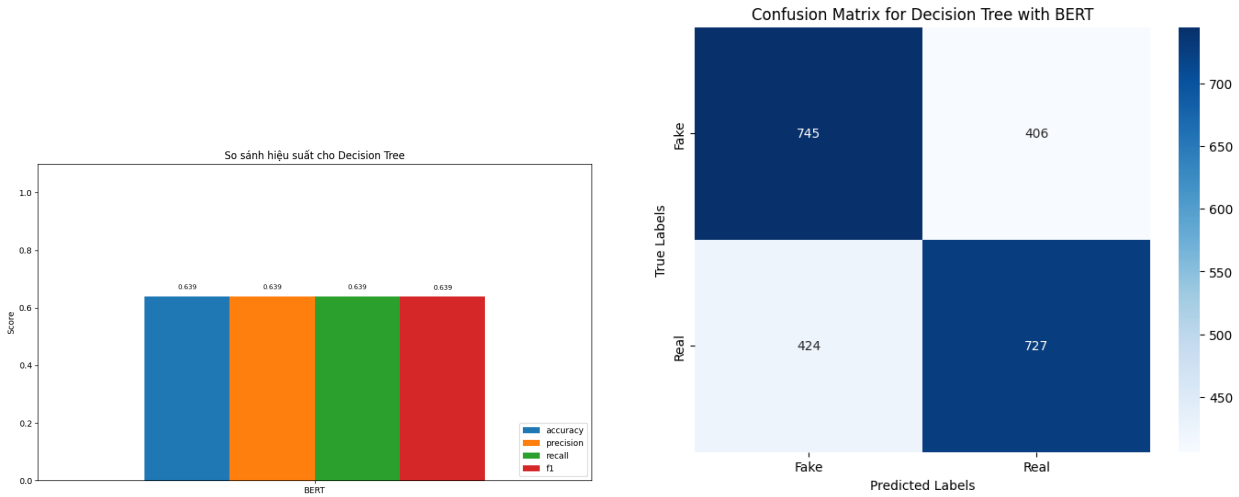
### 4.4.1 Logistic Regression




(a) Classification Report (b) Confusion Matrix

Hình 4.7: Kết quả của mô hình Logistic Regression sau Grid Search

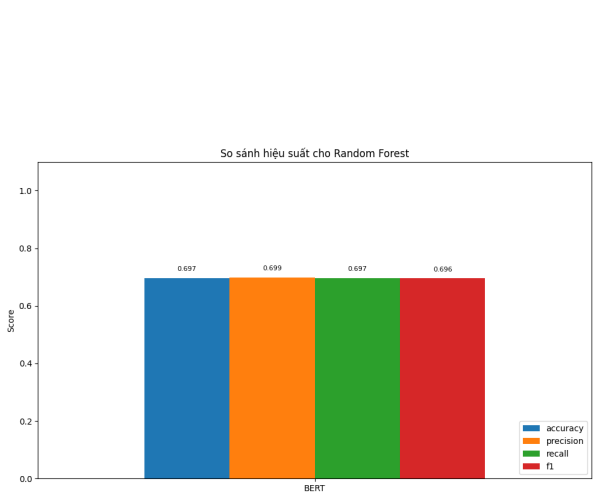
### 4.4.2 Decision Tree



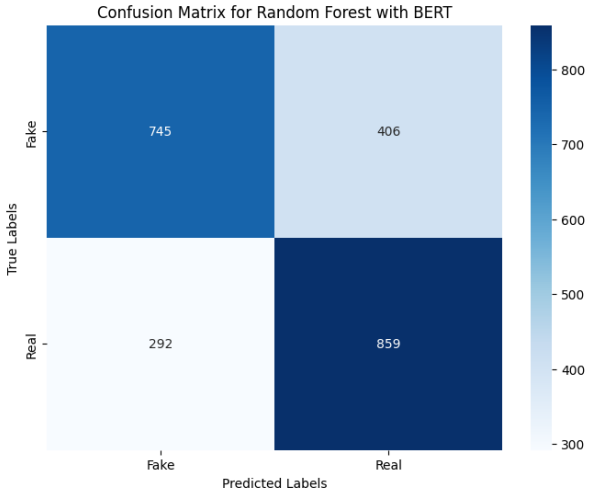

(a) Classification Report (b) Confusion Matrix

Hình 4.8: Kết quả của mô hình Decision Tree sau Grid Search

### 4.4.3 Random Forest



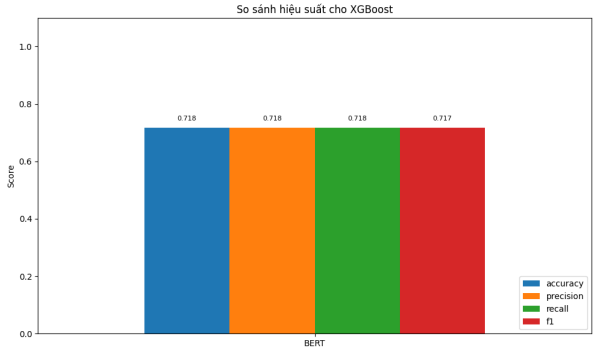

(a) Classification Report



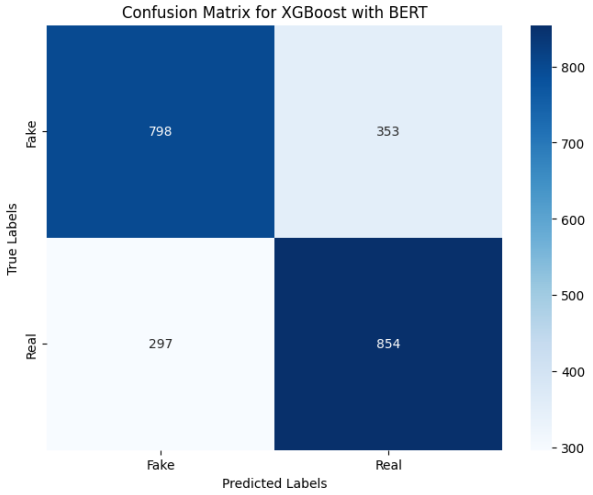

(b) Confusion Matrix

Hình 4.9: Kết quả của mô hình Random Forest sau Grid Search

### 4.4.4 XGBoost



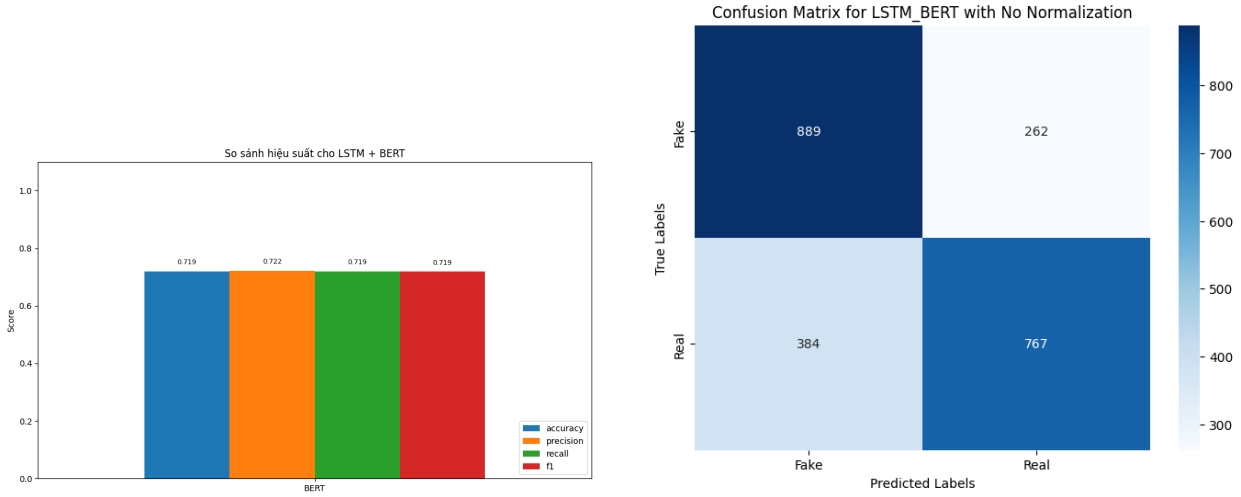

(a) Classification Report




(b) Confusion Matrix

Hình 4.10: Kết quả của mô hình XGBoost sau Grid Search

### 4.4.5 LSTM



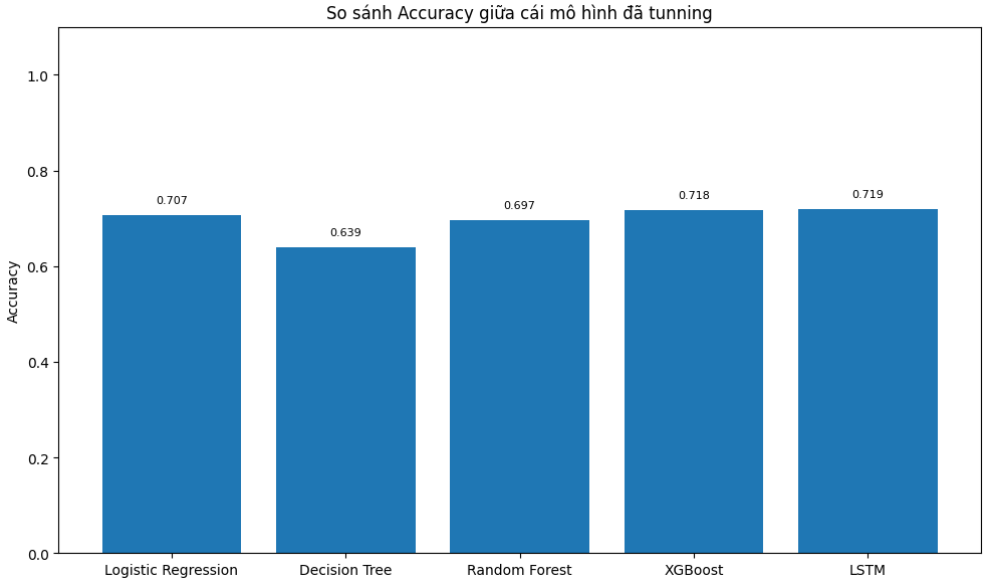

(a) Classification Report (b) Confusion Matrix

Hình 4.11: Kết quả của mô hình LSTM sau tinh chỉnh

Sau khi tinh chỉnh, chúng tôi quan sát thấy các mô hình đạt Accuracy dao động từ *0.639* đến *0.719*, trong đó *LSTM (0.719)* và *XGBoost (0.718)* tiếp tục duy trì hiệu năng cao nhất. So với kết quả trước khi tinh chỉnh, độ chính xác của hầu hết mô hình chỉ tăng nhẹ hoặc giữ ổn định, cho thấy các mô hình ban đầu đã có thiết lập khá phù hợp với đặc trưng của dữ liệu BERT. Đáng chú ý, XGBoost có mức cải thiện rõ nhất *(từ 0.706 lên 0.718)*, phản ánh việc tinh chỉnh tham số đã giúp mô hình khai thác tốt hơn các mối quan hệ phi tuyến trong không gian embedding. LSTM chỉ tăng rất nhẹ *(từ 0.714 lên 0.719)*, cho thấy mô hình này đã gần đạt mức tối ưu ở lần huấn luyện đầu. Trong khi đó, Decision Tree *(0.639)* và Random Forest *(0.697)* hầu như không cải thiện đáng kể, gợi ý rằng cấu trúc cây quyết định khó tận dụng tốt đặc trưng biểu diễn ngữ nghĩa từ BERT.

## 4.5 Tổng kết và so sánh Kết quả

Chúng tôi tổng hợp lại độ chính xác (Accuracy) của tất cả các mô hình sau khi đã được tinh chỉnh để có cái nhìn so sánh cuối cùng.




Hình 4.12: Hiệu suất của 5 mô hình sau Grid Search

Từ biểu đồ so sánh trên, chúng tôi rút ra các kết luận sau:

- Mô hình LSTM đạt được hiệu suất cao nhất trên tập kiểm tra, với độ chính xác *0.719*. Các chỉ số Precision, Recall và F1-score cũng xác nhận đây là mô hình có hiệu năng vượt trội so với các mô hình còn lại.
- Việc sử dụng đặc trưng BERT kết hợp với các kỹ thuật xử lý mất cân bằng dữ liệu (Tomek Links và NearMiss) đã cung cấp một nền tảng biểu diễn mạnh mẽ và ổn định cho bài toán phân loại tin giả. Các đặc trưng ngữ nghĩa từ BERT giúp mô hình dễ dàng nhận biết sự khác biệt tinh vi giữa tin thật và tin giả, trong khi các kỹ thuật cân bằng dữ liệu giúp giảm thiểu hiện tượng thiên lệch trong huấn luyện.
- Quá trình tinh chỉnh siêu tham số đã giúp cải thiện hiệu suất của các mô hình, đặc biệt là XGBoost, khi mô hình này tận dụng tốt hơn mối quan hệ phi tuyến trong không gian embedding. Tuy mức cải thiện tổng thể không quá lớn do dữ liệu đã được mã hóa ngữ nghĩa tốt từ trước, nhưng kết quả đạt được cho thấy việc tinh chỉnh vẫn đóng vai trò quan trọng trong việc tối ưu hóa hiệu năng, giúp mô hình đạt độ ổn định cao hơn và phù hợp hơn với đặc trưng của bộ dữ liệu FakeNewsNet.

- Mô hình XGBoost (Tuned) cho thấy khả năng nhận diện tin giả (nhãn 0) và tin thật (nhãn 1) khá cân bằng, với số lượng dự đoán sai ở mức tối thiểu so với các mô hình còn lại. Cụ thể, mô hình *dự đoán đúng 798 tin giả và 854 tin thật*, chỉ *nhầm lẫn 353 trường hợp tin giả bị nhận diện sai* và *297 trường hợp tin thật bị phân loại nhầm*. Điều này phản ánh rằng XGBoost đã tận dụng tốt đặc trưng ngữ nghĩa từ BERT để học được ranh giới phi tuyến giữa hai lớp dữ liệu. So sánh với LSTM và Random Forest, mô hình XGBoost không chỉ duy trì độ chính xác cao mà còn đạt được sự ổn định trong khả năng phân biệt hai loại tin tức, giúp giảm thiểu hiện tượng thiên lệch mô hình và đảm bảo hiệu năng nhất quán trên bộ dữ liệu FakeNewsNet.

# Chương 5

# KẾT LUẬN VÀ KIẾN NGHỊ

## 5.1 Kết luận

Dự án đã hoàn thành mục tiêu nghiên cứu và phát triển mô hình trí tuệ nhân tạo có khả năng phát hiện tin giả dựa trên phân tích tiêu đề văn bản, thông qua một quy trình nghiên cứu có hệ thống, bao gồm xử lý dữ liệu mất cân bằng bằng kỹ thuật Undersampling. Thử nghiệm cho thấy việc sử dụng đặc trưng BERT Embedding mang lại hiệu suất vượt trội và ổn định cho tất cả các mô hình cơ sở so với TF-IDF và Word2Vec. Sau quá trình tinh chỉnh siêu tham số, mô hình LSTM sử dụng đầu vào từ BERT đã đạt hiệu suất cao nhất trên tập kiểm tra với độ chính xác Accuracy là 0.719, tận dụng tốt thông tin ngữ cảnh tuần tự. Mô hình XGBoost cũng cho thấy hiệu suất gần như tương đương (0.718), đồng thời thể hiện khả năng phân loại cân bằng giữa hai lớp tin giả và tin thật. Dự án đã xây dựng được một mô hình có tính ứng dụng cao, có thể hỗ trợ tự động hóa quy trình kiểm chứng thông tin, góp phần giảm thiểu sự lan truyền của tin sai lệch trên mạng xã hội.

## 5.2 Kiến nghị

Mặc dù mô hình đạt được kết quả khả quan, đề tài vẫn có những hạn chế nhất định về phạm vi dữ liệu. Để tiếp tục nâng cao tính ứng dụng và độ chính xác của hệ thống, chúng tôi kiến nghị mở rộng nghiên cứu và phát triển theo các hướng sau: (1) Mở rộng phạm vi dữ liệu bao gồm toàn bộ nội dung bài báo thay vì chỉ tiêu đề, đồng thời tích hợp Ngữ cảnh và Đa phương tiện (mô hình lan truyền, độ tin cậy người dùng, hình ảnh) bằng kiến trúc như Mạng Nơ-ron Đồ thị. (2) Tập trung vào Tiếng Việt bằng cách triển khai các mô hình BERT đã được tiền huấn luyện trên ngôn ngữ này (ví dụ: PhoBERT). (3) Về Mô hình, cần tiến hành tinh chỉnh end-to-end toàn bộ mô hình BERT trên nhiệm vụ cụ thể để học các đặc trưng tin giả hiệu quả hơn, và khám phá các mô hình lai (Hybrid) cùng các kỹ thuật xử lý mất cân bằng tiên tiến hơn (SMOTE).

# Tài liệu tham khảo


[1] Fake news | history, examples, great moon hoax, war of the worlds, election of 2016, & possible solutions | britannica. [Online]. Available: https://www.britannica.com/topic/fake-news?utm_source=chatgpt.com

[2] "Article | fake news on social media: the impact on society | university of stirling." [Online]. Available: https://www.stir.ac.uk/research/hub/publication/1879543

[3] X. Zhou and R. Zafarani, "A survey of fake news: Fundamental theories, detection methods, and opportunities," *ACM Computing Surveys*, vol. 53, no. 5, p. 1–40, Sep. 2020. [Online]. Available: http://dx.doi.org/10.1145/3395046

[4] Q. Guo, Z. Kang, L. Tian, and Z. Chen, "Tiefake: Title-text similarity and emotion-aware fake news detection," 2023. [Online]. Available: https://arxiv.org/abs/2304.09421

[5] "Natural language processing," page Version ID: 1316068589. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Natural_language_processing&oldid=1316068589

[6] D. Jurafsky, J. Martin, A. Kehler, K. Linden, and N. Ward, *Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition*, 01 2000.

[7] Text preprocessing in NLP. Section: NLP. [Online]. Available: https://www.geeksforgeeks.org/nlp/text-preprocessing-for-nlp-tasks/

[8] An introduction to text data cleaning and normalization in python. [Online]. Available: https://www.datacamp.com/tutorial/textacy-text-data-cleaning-normalization-python?utm_source=chatgpt.com

[9] Y. Shen, Q. Liu, N. Guo, J. Yuan, and Y. Yang, "Fake news detection on social networks: A survey," *Applied Sciences*, vol. 13, no. 21, 2023. [Online]. Available: https://www.mdpi.com/2076-3417/13/21/11877

[10] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[11] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Information Processing & Management*, vol. 24, no. 5, pp. 513–523, 1988.

[12] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," 2013. [Online]. Available: https://arxiv.org/abs/1301.3781

[13] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," 2019. [Online]. Available: https://arxiv.org/abs/1810.04805

[14] H. He and E. Garcia, "Learning from imbalanced data," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 21, pp. 1263 – 1284, 10 2009.

[15] "Two modifications of cnn," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-6, no. 11, pp. 769–772, 1976.

[16] G. Batista, R. Prati, and M.-C. Monard, "A study of the behavior of several methods for balancing machine learning training data," *SIGKDD Explorations*, vol. 6, pp. 20–29, 06 2004.

[17] Zhang, j.p. and mani, i. (2003) KNN approach to unbalanced data distributions a case study involving information extraction. proceeding of international conference on machine learning (ICML 2003), workshop on learning from imbalanced data sets, washington DC, 21 august 2003. - references - scientific research publishing. [Online]. Available: https://www.scirp.org/reference/referencespapers?referenceid=1603053

[18] D. R. Cox, "The regression analysis of binary sequences," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 20, no. 2, pp. 215–242, 1958. [Online]. Available: http://www.jstor.org/stable/2983890

[19] S. Lemeshow and D. Hosmer, *Logistic Regression*, 07 2005.

[20] J. R. Quinlan, "Induction of decision trees," vol. 1, no. 1, pp. 81–106. [Online]. Available: https://doi.org/10.1007/BF00116251

[21] L. Breiman, "Random forests," vol. 45, no. 1, pp. 5–32. [Online]. Available: https://doi.org/10.1023/A:1010933404324

[22] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," 08 2016, pp. 785–794.

[23] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, pp. 1735–1780, 11 1997.

[24] Classification: Accuracy, recall, precision, and related metrics | machine learning. [Online]. Available: https://developers.google.com/machine-learning/crash-course/classification/accuracy-precision-recall

[25] Accuracy vs. precision vs. recall in machine learning: What is the difference? [Online]. Available: https://encord.com/blog/classification-metrics-accuracy-precision-recall/

[26] "Precision and recall," page Version ID: 1316777840. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Precision_and_recall&oldid=1316777840

[27] S. Kumar. Evaluation metrics for classification model. [Online]. Available: https://www.analyticsvidhya.com/blog/2021/07/metrics-to-evaluate-your-classification-model-to-take-the-right-decisions/

[28] J. Brownlee. A gentle introduction to k-fold cross-validation. [Online]. Available: https://www.machinelearningmastery.com/k-fold-cross-validation/

[29] 3.1. cross-validation: evaluating estimator performance. [Online]. Available: https://scikit-learn/stable/modules/cross_validation.html

[30] StratifiedKFold. [Online]. Available: https://scikit-learn/stable/modules/generated/sklearn.model_selection.StratifiedKFold.html

[31] "Hyperparameter (machine learning)," page Version ID: 1316566390. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Hyperparameter_(machine_learning)&oldid=1316566390

[32] H. J. P. Weerts, A. C. Mueller, and J. Vanschoren, "Importance of tuning hyperparameters of machine learning algorithms," 2020. [Online]. Available: https://arxiv.org/abs/2007.07588

[33] R. Shah. Tune hyperparameters with GridSearchCV. [Online]. Available: https://www.analyticsvidhya.com/blog/2021/06/tune-hyperparameters-with-gridsearchcv/

[34] G. L. E. Team. Hyperparameter tuning with GridSearchCV. [Online]. Available: https://www.mygreatlearning.com/blog/gridsearchcv/