TRƯỜNG ĐẠI HỌC QUY NHƠN

KHOA TOÁN VÀ THỐNG KÊ

BÁO CÁO

HỌC PHẦN: XỬ LÝ NGÔN NGỮ TỰ NHIÊN

ĐỀ TÀI:

Phát hiện tin giả dựa trên mô hình DistilBERT

Sinh viên thực hiện: NGUYỄN HOÀI NAM

Mã sinh viên: 4554110013

Lớp: KHOA HỌC DỮ LIỆU K45

Giảng viên hướng dẫn: TS. LÊ QUANG HÙNG

Quy Nhơn, tháng 11, năm 2025

Lời cảm ơn

Em xin bày tỏ lòng biết ơn sâu sắc đến Thầy **TS. Lê Quang Hùng**, người đã tận tâm giảng dạy, định hướng và truyền cảm hứng cho em trong suốt quá trình học tập học phần *Xử lý Ngôn ngữ Tự nhiên*. Những kiến thức quý báu và sự tận tụy của Thầy không chỉ giúp em tiếp cận vững vàng các khái niệm nền tảng, mà còn mở ra cho em cơ hội khám phá sâu hơn về các kỹ thuật và mô hình ứng dụng trong lĩnh vực phát hiện tin giả.

Em chân thành cảm ơn Thầy vì tinh thần trách nhiệm, sự hướng dẫn tận tâm và những chia sẻ đầy cảm hứng trong mỗi buổi học. Những giá trị đó sẽ luôn là hành trang quý báu giúp em tiếp tục phát triển trên con đường học tập và nghiên cứu sau này..

Trân trọng,
Nguyễn Hoài Nam

Mục lục

Chương 1

Giới thiệu tổng quan

1.1 Bối cảnh & động cơ nghiên cứu

Trong bối cảnh bùng nổ thông tin hiện nay, các nền tảng trực tuyến như mạng xã hội, diễn đàn, trang tin tức điện tử phát triển mạnh mẽ, kéo theo đó là sự lan truyền nhanh chóng của các nội dung chưa được kiểm chứng. **Tin giả (Fake News)** — những thông tin sai lệch, bóp méo hoặc có chủ đích gây hiểu lầm — đang trở thành một vấn đề đáng lo ngại, có thể gây tác động tiêu cực đến nhận thức cộng đồng, định hướng dư luận, thậm chí ảnh hưởng đến chính trị, kinh tế, y tế và an ninh xã hội.

Nhằm hạn chế tác hại của tin giả, việc xây dựng các hệ thống **phát hiện tin giả tự động** dựa trên dữ liệu văn bản trở thành một hướng nghiên cứu quan trọng trong lĩnh vực **Xử lý ngôn ngữ tự nhiên (Natural Language Processing – NLP)** và **Học sâu (Deep Learning)**. Các mô hình ngôn ngữ hiện đại, đặc biệt là họ mô hình Transformer, đã chứng minh được hiệu quả vượt trội trong việc hiểu ngữ cảnh và ngữ nghĩa của văn bản, từ đó hỗ trợ tốt cho các bài toán phân loại như phát hiện spam, phân tích cảm xúc, và phát hiện tin giả.

Trong nghiên cứu này, đề tài tập trung vào việc xây dựng một hệ thống **phát hiện tin giả sử dụng mô hình DistilBERT được fine-tune trên tập dữ liệu tin tức đã gán nhãn**. Cụ thể:

- Dữ liệu tin tức (tiêu đề và nội dung) được thu thập từ một bộ dữ liệu công khai trên Kaggle, gồm khoảng **44 898** bản ghi, được chia thành hai nhóm: *tin giả* (fake) và *tin thật* (true).
- Văn bản được tiền xử lý (làm sạch, ghép `title + text`) và mã hoá bằng **tokenizer** của mô hình `distilbert-base-uncased`.
- Mô hình **DistilBERT** được fine-tune cho nhiệm vụ *phân loại nhị phân* (fake / real) với các chỉ số đánh giá chính: *Accuracy*, *Precision*, *Recall*, *F1-score*.
- Áp dụng **Stratified K-Fold Cross Validation** để đánh giá độ ổn định mô hình, sau đó huấn luyện một **mô hình cuối cùng (final model)** và kiểm tra trên tập hold-out.
- Mô hình sau khi huấn luyện được triển khai dưới dạng ứng dụng web sử dụng **Streamlit**, kèm theo cơ chế **giải thích mô hình** bằng phương pháp **Integrated Gradients** (thuộc thư viện Captum), giúp trực quan hóa mức độ đóng góp của từng từ vào quyết định phân loại.

Do đó, đề tài **không còn tập trung so sánh nhiều mô hình như Logistic Regression, LSTM hay các kỹ thuật nhúng Word2Vec, FastText, BERT-tiny**, mà tập trung chuyên sâu vào **một pipeline end-to-end với DistilBERT**: từ xử lý dữ liệu, huấn luyện, đánh giá cho tới triển khai ứng dụng và giải thích kết quả.

1.2 Mục tiêu của báo cáo

Dựa trên bối cảnh và định hướng nghiên cứu nêu trên, các mục tiêu chính của báo cáo được xác định như sau:

- **Xây dựng pipeline phát hiện tin giả end-to-end** sử dụng mô hình **DistilBERT**:
  - Tiền xử lý dữ liệu văn bản (làm sạch, ghép tiêu đề và nội dung).
  - Mã hoá văn bản bằng tokenizer DistilBERT.
  - Tổ chức dữ liệu dưới dạng `Dataset` phục vụ huấn luyện.
  - Fine-tune mô hình DistilBERT cho bài toán phân loại tin thật/tin giả.
- **Đánh giá hiệu năng mô hình**:
  - Sử dụng **Stratified 3-Fold Cross Validation** để kiểm tra độ ổn định và khả năng tổng quát của mô hình.
  - Đánh giá mô hình cuối cùng trên tập hold-out bằng các chỉ số: Accuracy, Precision, Recall, F1-score, kèm theo báo cáo phân loại và ma trận nhầm lẫn.
- **Triển khai mô hình thành ứng dụng thực tế**:
  - Xây dựng giao diện web đơn giản bằng **Streamlit**, cho phép người dùng nhập nội dung tin tức bằng tiếng Anh và nhận kết quả phân loại (Real/Fake).
  - Tích hợp **Integrated Gradients** để giải thích quyết định của mô hình ở mức token: tô màu các từ/ngữ đóng góp mạnh theo hướng "Fake News" hoặc "Real News".
- **Phân tích, thảo luận kết quả**:
  - Phân tích ưu điểm và hạn chế của mô hình DistilBERT trên tập dữ liệu tin giả.
  - Đề xuất các hướng cải tiến trong tương lai: mở rộng ngôn ngữ, bổ sung dữ liệu, hoặc sử dụng mô hình Transformer lớn hơn.

Tóm lại, báo cáo này hướng tới việc **xây dựng, đánh giá và triển khai một hệ thống phát hiện tin giả dựa trên DistilBERT** một cách đầy đủ và có tính thực tiễn, đồng thời minh họa rõ ràng vai trò của các mô hình ngôn ngữ hiện đại trong bài toán an toàn thông tin trên không gian số.

Chương 2

Tổng quan về tin giả

2.1 Khái niệm tin giả

Trong bối cảnh thông tin phát triển với tốc độ chưa từng có, **tin giả (Fake News)** đã trở thành một trong những vấn đề đáng quan ngại nhất trên mạng Internet. Tin giả được hiểu là các nội dung chứa thông tin sai lệch, xuyên tạc hoặc được tạo ra với chủ đích gây hiểu lầm cho người đọc. Theo Allcott và Gentzkow (2017), tin giả là "những bài viết chứa nội dung cố ý sai sự thật, được trình bày dưới hình thức tin tức nhằm đánh lừa độc giả".

Sự phổ biến của tin giả phần lớn đến từ:

- tốc độ lan truyền nhanh trên mạng xã hội (Facebook, Twitter, TikTok),
- khả năng thao túng nhận thức cộng đồng,
- sự khó khăn trong việc kiểm chứng nguồn gốc thông tin,
- các mô hình lan truyền tự động và thuật toán đề xuất nội dung.

Những yếu tố này khiến việc phát hiện tin giả trở thành một yêu cầu cấp thiết, đặc biệt trong các lĩnh vực chính trị, sức khỏe cộng đồng, kinh tế và an ninh mạng.

2.2 Phân loại tin giả

Việc phân loại tin giả giúp định hướng phương pháp xử lý và mô hình hóa. Dựa trên tổng quan các nghiên cứu, tin giả có thể được phân chia thành ba nhóm chính:

1. Phân loại theo mục đích

- **Tin giả có chủ đích (Intentional Fake News):** được tạo ra nhằm thao túng dư luận, phục vụ mục tiêu chính trị hoặc lợi ích cá nhân.
- **Tin giả vì lợi nhuận (Profit-driven Fake News):** lợi dụng tiêu đề giật gân để kéo lượt xem và quảng cáo.
- **Tin sai lệch vô ý (Misinformation):** phát sinh từ sự hiểu nhầm hoặc chia sẻ thiếu kiểm chứng.

2. Phân loại theo nội dung

- **Tin bịa đặt hoàn toàn (Fabricated Content):** không dựa trên bất kỳ sự kiện thật nào.
- **Tin bóp méo (Manipulated Content):** sự kiện thật nhưng bị chỉnh sửa, cắt ghép hoặc thêm thắt để thay đổi ý nghĩa.
- **Tin đặt trong ngữ cảnh sai (Misleading Context):** thông tin đúng nhưng được sử dụng trong bối cảnh không phù hợp.

3. Phân loại theo hình thức thể hiện

- **Tin văn bản (Text-based Fake News):** dạng bài viết, tiêu đề hoặc mô tả sai lệch. Đây là dạng được xử lý trong đề tài.
- **Tin đa phương tiện (Multimodal Fake News):** kết hợp hình ảnh, video hoặc âm thanh giả mạo để tăng sức thuyết phục.

2.3 Các phương pháp phát hiện tin giả

Nghiên cứu về phát hiện tin giả thường xoay quanh ba hướng chính:

(1) Phương pháp dựa trên đặc trưng nội dung

Đây là hướng truyền thống, tập trung xử lý văn bản thuần túy. Các kỹ thuật phổ biến gồm:

- Biểu diễn văn bản: Bag-of-Words, TF–IDF, n-grams.
- Mô hình học máy tuyến tính: Naive Bayes, SVM, **Logistic Regression**.
- Phân tích ngôn ngữ: mức độ chủ quan, cảm xúc, cấu trúc câu, độ mạch lạc.

Tuy hiệu quả với tập dữ liệu nhỏ và ngôn ngữ đơn giản, nhưng phương pháp này khó nắm bắt được ngữ nghĩa sâu và ngữ cảnh hai chiều.

(2) Phương pháp dựa trên đặc trưng xã hội

Một số nghiên cứu phân tích:

- hành vi lan truyền thông tin (retweet, chia sẻ),
- đặc điểm tài khoản đăng tin,
- động lực tương tác của người dùng.

Hướng này phù hợp với dữ liệu mạng xã hội nhưng không áp dụng cho dữ liệu văn bản thuần như trong đề tài.

(3) Phương pháp dựa trên học sâu và mô hình ngôn ngữ

Với sự phát triển của kiến trúc Transformer, các mô hình ngôn ngữ tiền huấn luyện (PLM) như BERT, RoBERTa, XLNet, và **DistilBERT** đã tạo ra bước đột phá trong xử lý văn bản:

- Cho phép mô hình *hiểu ngữ cảnh hai chiều* và quan hệ phụ thuộc dài.
- Khả năng tổng quát hóa mạnh trên nhiều tập dữ liệu khác nhau.
- Giảm phụ thuộc vào kỹ thuật tiền xử lý thủ công.

Trong đề tài này, **DistilBERT** được lựa chọn để fine-tuning cho nhiệm vụ phân loại tin giả nhờ ưu điểm:

- nhẹ hơn 40% so với BERT,
- tốc độ nhanh hơn 60%,
- giữ lại 97% chất lượng mô hình gốc.

2.4 Tóm tắt

Chương này trình bày những kiến thức quan trọng làm nền tảng cho đề tài, bao gồm khái niệm, phân loại tin giả và tổng quan các phương pháp phát hiện phổ biến. Những nội dung này là cơ sở quan trọng để hiểu rõ vì sao mô hình DistilBERT và phương pháp fine-tuning được lựa chọn làm hướng tiếp cận chính trong các chương tiếp theo.

Chương 3

Định nghĩa bài toán

3.1 Mô tả bài toán

Bài toán **phát hiện tin giả (Fake News Detection)** là một bài toán phân loại văn bản trong lĩnh vực **Xử lý ngôn ngữ tự nhiên (NLP)**. Mục tiêu chính là xây dựng một mô hình có khả năng **phân biệt giữa tin thật (real)** và **tin giả (fake)** dựa trên nội dung văn bản hoặc tiêu đề bài báo.

Cho trước một tập dữ liệu gồm các bài viết hoặc tiêu đề đã được gán nhãn:

- **Đầu vào (Input):** Văn bản tin tức $X = \{x_1, x_2, ..., x_n\}$, trong đó mỗi x_i là một câu hoặc đoạn văn bản.
- **Đầu ra (Output):** Nhãn $y_i \in \{\text{real}, \text{fake}\}$ thể hiện loại tin tức tương ứng.

Bài toán có thể được mô tả dưới dạng một hàm học f_θ:

$$f_\theta : X \to Y$$

Trong đó f_θ là mô hình học máy hoặc học sâu được huấn luyện trên dữ liệu đã gán nhãn, với mục tiêu cực tiểu hóa sai số giữa dự đoán và thực tế.

3.2 Mục tiêu và yêu cầu của bài toán

- Xây dựng mô hình có khả năng nhận diện và phân loại chính xác tin thật – tin giả.
- Đánh giá, so sánh hiệu quả giữa các mô hình học máy truyền thống và mô hình học sâu.
- Ứng dụng các kỹ thuật biểu diễn văn bản hiện đại như **Word2Vec**, **FastText**, và **BERT** để cải thiện hiệu suất mô hình.
- Đảm bảo mô hình có khả năng tổng quát tốt, tránh overfitting, và có thể mở rộng cho dữ liệu thực tế.

3.3 Dữ liệu đầu vào và đầu ra

Dữ liệu đầu vào

Dữ liệu được trích xuất từ các nguồn tin tức có gán nhãn **tin thật (real)** và **tin giả (fake)**. Các trường thông tin chính bao gồm:

- `title`: Tiêu đề bài báo.
- `text`: Nội dung bài viết (tùy chọn, có thể rút gọn).
- `label`: Nhãn phân loại ("real" hoặc "fake").

Văn bản được tiền xử lý nhằm loại bỏ nhiễu như ký tự đặc biệt, từ dừng, dấu câu và chuyển toàn bộ sang chữ thường để chuẩn hóa dữ liệu.

Dữ liệu đầu ra

Mô hình sẽ trả về:

- Nhãn dự đoán cho mỗi bài viết: `real` hoặc `fake`.
- Xác suất tương ứng (độ tin cậy) của mô hình cho từng dự đoán.

3.4 Không gian đặc trưng (Feature Space)

Mỗi văn bản sau khi tiền xử lý sẽ được biểu diễn dưới dạng vector số thông qua các kỹ thuật nhúng từ (*word embedding*):

- **Word2Vec:** Học biểu diễn từ dựa trên ngữ cảnh xuất hiện.
- **FastText:** Cải tiến Word2Vec bằng cách xem xét cấu trúc bên trong từ (subword).
- **BERT:** Mô hình ngôn ngữ hai chiều dựa trên kiến trúc Transformer, giúp hiểu ngữ cảnh sâu hơn.

Các vector đặc trưng này sẽ là đầu vào cho mô hình học máy hoặc học sâu để thực hiện phân loại.

3.5 Hàm mất mát và tiêu chí đánh giá

Để đánh giá hiệu quả của mô hình, sử dụng các thước đo phổ biến trong bài toán phân loại:

- **Accuracy (Độ chính xác):** Tỷ lệ dự đoán đúng trên toàn bộ mẫu.
- **Precision:** Tỷ lệ tin giả được dự đoán đúng trên tổng số tin được gán là giả.
- **Recall:** Tỷ lệ tin giả được phát hiện đúng trên tổng số tin giả thực tế.
- **F1-Score:** Trung bình điều hòa giữa Precision và Recall.

Hàm mất mát thường dùng trong quá trình huấn luyện:

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log(\hat{y}_i) + (1-y_i)\log(1-\hat{y}_i)\right]$$

Trong đó y_i là nhãn thực tế, $\hat{y}_i$ là xác suất dự đoán của mô hình.

3.6 Kỳ vọng của mô hình

Mô hình được kỳ vọng đạt được:

- Độ chính xác cao và khả năng phân biệt rõ giữa tin thật và tin giả.
- Hiểu được ngữ nghĩa sâu và ngữ cảnh hai chiều của văn bản.
- Tính ổn định và khả năng mở rộng khi áp dụng cho các bộ dữ liệu lớn hơn trong tương lai.

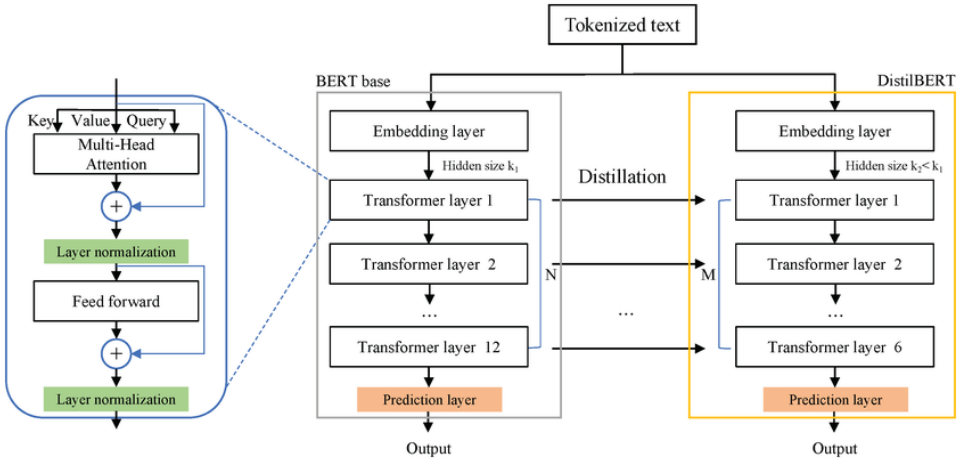
Tóm lại, bài toán phát hiện tin giả là một bài toán phân loại nhị phân trong lĩnh vực NLP, đòi hỏi kết hợp giữa **kỹ thuật biểu diễn ngôn ngữ hiện đại** và **mô hình học máy/học sâu** để đạt được hiệu quả cao trong việc nhận diện và ngăn chặn sự lan truyền của các thông tin sai lệch trên môi trường số.

Chương 4

Phương pháp nghiên cứu

Chương này trình bày chi tiết phương pháp nghiên cứu được sử dụng trong đề tài, bao gồm mô hình ngôn ngữ chính DistilBERT và quy trình biểu diễn văn bản trước khi đưa vào mô hình. Các bước được xây dựng dựa trên mã nguồn đã triển khai, bảo đảm tính nhất quán giữa lý thuyết và thực nghiệm.

4.1 Mô hình DistilBERT



Hình 4.1: Tổng quan về mô hình DistilBERT.

1. Khái niệm

DistilBERT là mô hình ngôn ngữ dựa trên kiến trúc *Transformer*, được xây dựng bằng kỹ thuật **knowledge distillation** từ BERT. Mục tiêu của DistilBERT là giảm kích thước và tăng tốc độ suy luận, trong khi vẫn giữ lại phần lớn hiệu năng của BERT gốc.

So với $\text{BERT}_{\text{base}}$ (12 tầng encoder), DistilBERT chỉ gồm 6 tầng, nhẹ hơn khoảng 40% tham số nhưng đạt khoảng 95% chất lượng. Nhờ đó, DistilBERT phù hợp cho các bài toán phân tích văn bản cần tốc độ và tài nguyên thấp như phát hiện tin giả.

Trong đề tài này, DistilBERT đóng vai trò là mô hình chính trong nhiệm vụ **phân loại nhị phân** (tin thật / tin giả).

2. Kiến trúc Transformer và Self-Attention

DistilBERT kế thừa kiến trúc encoder của Transformer, trong đó thành phần cốt lõi là cơ chế **Self-Attention**. Với chuỗi gồm n token, mỗi token i được ánh xạ thành ba vector:

$$q_i = x_i W^Q, \quad k_i = x_i W^K, \quad v_i = x_i W^V$$

Điểm chú ý giữa token i và token j được tính như sau:

$$\alpha_{ij} = \text{softmax}\left(\frac{q_i k_j^\top}{\sqrt{d_k}}\right)$$

Biểu diễn ngữ cảnh của token i:

$$h_i = \sum_{j=1}^{n} \alpha_{ij} v_j$$

Self-attention cho phép mô hình học được quan hệ dài hạn giữa các từ, giúp phát hiện các dấu hiệu tinh vi thường xuất hiện trong tin giả.

3. Kiến trúc DistilBERT và hàm mất mát

DistilBERT được huấn luyện bằng ba thành phần hàm mất mát:

$$\mathcal{L}_{\text{total}} = \lambda_1 \mathcal{L}_{\text{KD}} + \lambda_2 \mathcal{L}_{\text{MLM}} + \lambda_3 \mathcal{L}_{\text{cos}}$$

Trong đó:

- **Knowledge Distillation Loss**:

$$\mathcal{L}_{\text{KD}} = \text{KL}\left(\sigma\left(\frac{z^{(T)}}{T}\right) \middle\| \sigma\left(\frac{z^{(S)}}{T}\right)\right)$$

  với teacher là BERT, student là DistilBERT.

- **Masked Language Modeling Loss (MLM)**: dự đoán từ bị che như trong BERT.
- **Cosine Embedding Loss**: đảm bảo không gian biểu diễn của student giống teacher.

Nhờ đó, DistilBERT duy trì được hiệu năng cao dù có số tham số ít hơn.

4. DistilBERT trong bài toán phân loại tin giả

Khi fine-tuning cho nhiệm vụ phân loại, DistilBERT sinh ra biểu diễn của token `[CLS]`:

$$h_{\text{CLS}} \in \mathbb{R}^d$$

Sau đó đi qua tầng phân loại:

$$z = W h_{\text{CLS}} + b$$

Xác suất dự đoán lớp được tính bằng softmax:

$$P(y = k|x) = \frac{\exp(z_k)}{\sum_{j=1}^{2} \exp(z_j)}$$

Toàn bộ mô hình được tối ưu bằng **Cross-Entropy loss**.

5. Lý do lựa chọn DistilBERT

- Nhẹ hơn BERT 40%, suy luận nhanh hơn đáng kể.
- Hiệu năng cao (≈ 95% BERT-base).
- Khả năng nắm bắt ngữ cảnh mạnh, vượt trội so với TF–IDF và Bag-of-Words.
- Tích hợp trực tiếp trong thư viện `HuggingFace Transformers`, dễ fine-tune.
- Phù hợp triển khai thực tế (web app, API, Streamlit).

DistilBERT đáp ứng tốt yêu cầu về độ chính xác, tốc độ và tài nguyên của đề tài.

4.2 Kỹ thuật biểu diễn văn bản và Pipeline xử lý dữ liệu

Phần này trình bày quy trình xử lý dữ liệu và mã hoá văn bản dựa trên pipeline triển khai trong mã nguồn của đề tài. Đây là bước trung gian quan trọng giúp mô hình DistilBERT hiểu và học từ dữ liệu.

1. Pipeline tổng quan

`(title, text)` → Tiền xử lý → `full_text` → Tokenizer DistilBERT → (`input_ids`, `attention_mask`)

Pipeline được triển khai đúng theo code thực nghiệm:

- Đọc dữ liệu từ `fake_news.csv`, `true_news.csv`.
- Ghép `title` + `text` thành `full_text`.
- Làm sạch bằng hàm `clean_text()`.
- Mã hoá bằng tokenizer `distilbert-base-uncased`.
- Tạo dataset `FakeNewsDataset` cho huấn luyện.

2. Tiền xử lý văn bản

Hàm `clean_text()` thực hiện:

1. Loại bỏ URL.
2. Loại bỏ thẻ HTML.
3. Chuẩn hoá khoảng trắng.

Đây là bước quan trọng để loại nhiễu và tăng tính nhất quán của dữ liệu.

3. Tokenizer DistilBERT

Tokenizer sử dụng kiến trúc **WordPiece**. Quá trình mã hoá:

1. Chia văn bản thành token.
2. Tách subword khi gặp từ dài/hiếm.
3. Thêm `[CLS]`, `[SEP]`.
4. Chuyển thành `input_ids`.
5. Tạo `attention_mask`.

Ví dụ mã hoá (theo code):

```
enc = tokenizer(
    text,
    truncation=True,
    max_length=128,
    padding="max_length",
    return_tensors="pt"
)
```

4. Biểu diễn embedding

Mỗi token id_i được ánh xạ:

$$e_i = E(id_i)$$

Chuỗi vector sau đó đi qua 6 tầng encoder để tạo ra chuỗi biểu diễn ngữ cảnh:

$$h_1, h_2, \ldots, h_L$$

Vector quan trọng nhất:

$$h_{[\text{CLS}]} = \text{biểu diễn toàn văn bản}$$

5. Dataset và sinh batch dữ liệu

Lớp `FakeNewsDataset` được cài đặt nhằm kết nối dữ liệu đã mã hoá với `Trainer`. Mỗi item trả về:

- `input_ids`
- `attention_mask`
- `labels`

Đây là định dạng bắt buộc cho mô hình sequence classification của Transformers.

6. So sánh với các phương pháp biểu diễn truyền thống

- **TF–IDF** và **Bag-of-Words** không nắm bắt ngữ cảnh.
- **Word2Vec** và **FastText** tạo embedding cố định, không phụ thuộc câu.

Ngược lại:

DistilBERT tạo embedding phụ thuộc ngữ cảnh từng token

Do đó mô hình học được:

- Mối quan hệ xa trong câu
- Các dấu hiệu mỉa mai, giật tít
- Ngụy tạo nội dung tinh vi trong tin giả

Đây là lý do DistilBERT vượt trội trong nhiệm vụ phát hiện tin giả.

Chương 5

Thực nghiệm và đánh giá

Chương này trình bày chi tiết các thực nghiệm được tiến hành để đánh giá hiệu quả mô hình DistilBERT trong bài toán phát hiện tin giả. Toàn bộ quy trình thực nghiệm được triển khai theo pipeline: chuẩn bị dữ liệu, tiền xử lý, tokenization, huấn luyện bằng kỹ thuật K-fold cross validation, đánh giá mô hình cuối (hold-out test), và giải thích mô hình bằng LIME.

5.1 Dataset

1. Nguồn dữ liệu

Bộ dữ liệu được sử dụng trong nghiên cứu được lấy từ Kaggle, thuộc nhóm các tập dữ liệu phổ biến cho bài toán phát hiện tin giả. Dataset bao gồm hai tệp riêng biệt:

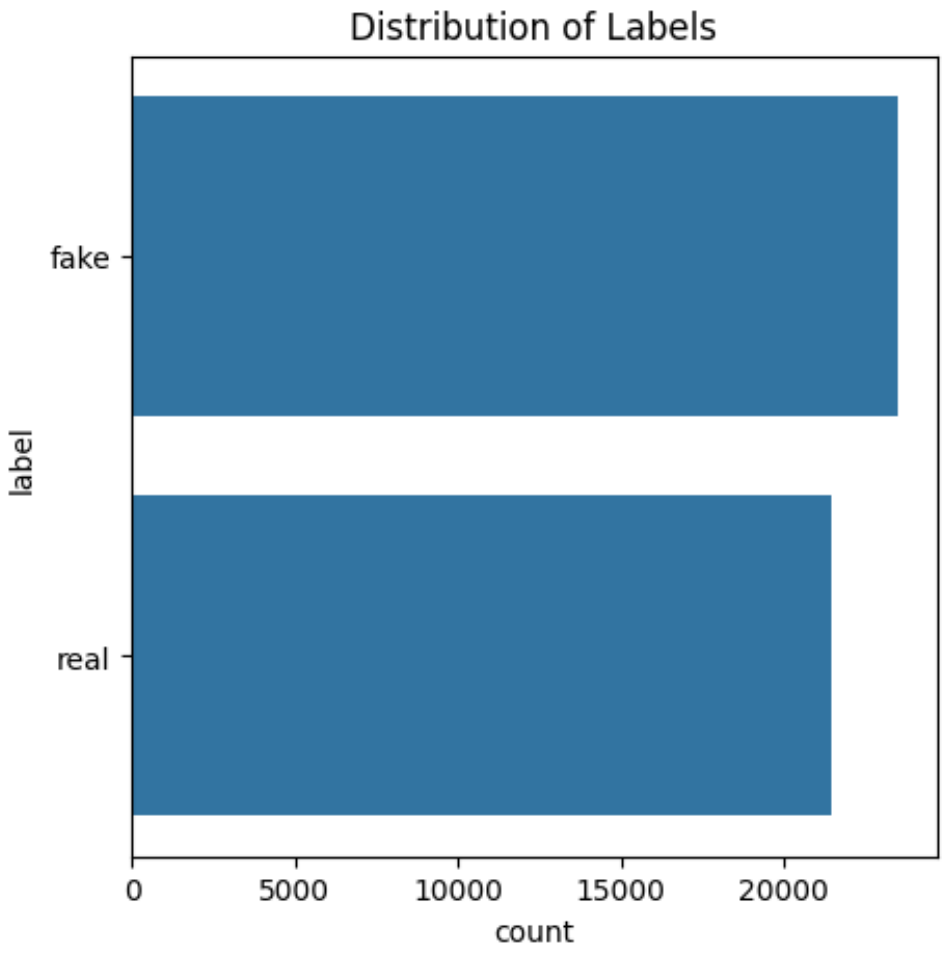
- `Fake.csv` — tập các bài viết bị gán nhãn **tin giả** (`label = 1`)
- `True.csv` — tập các bài viết được gán nhãn **tin thật** (`label = 0`)

Tổng cộng bộ dữ liệu chứa **44.898 mẫu**, thuộc hai lớp (Real/Fake) đã được cân bằng tương đối tốt và sẵn sàng để huấn luyện mô hình học sâu.

2. Phân bố nhãn

Phân bố nhãn được thống kê bằng:

```
df["label"].value_counts()
df["label"].value_counts(normalize=True)
```

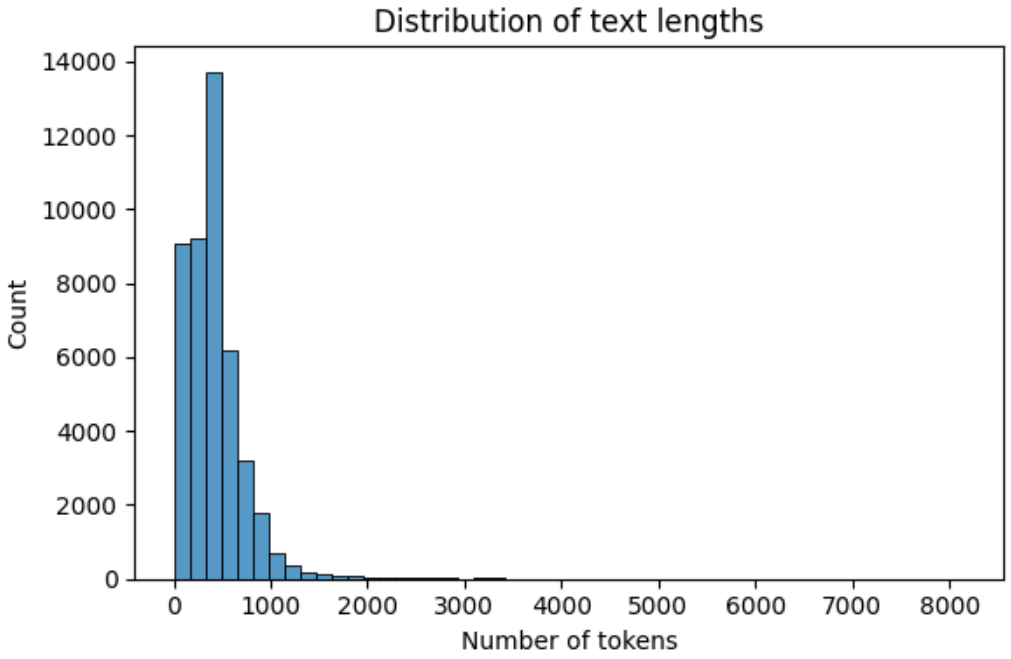


Hình 5.1: Phân bố nhãn `fake` và `real` trong tập dữ liệu.

Kết quả cho thấy hai lớp *fake* và *real* có số lượng tương đối cân bằng, giúp mô hình tránh được hiện tượng mất cân bằng dữ liệu (*class imbalance*). Điều này tạo điều kiện thuận lợi cho quá trình huấn luyện và giúp mô hình học được ranh giới phân loại ổn định hơn.

3. Thống kê độ dài văn bản

Sau khi ghép `title + text` thành trường `full_text`, độ dài văn bản được tính bằng số lượng token sau khi tách từ. Kết quả được trực quan hóa bằng biểu đồ histogram dưới đây.



Hình 5.2: Phân bố độ dài văn bản trong tập dữ liệu.

Biểu đồ cho thấy:

- Phần lớn văn bản có độ dài từ **200–800 token**, tập trung dày nhất quanh mức **300–600 token**.
- Một số văn bản có độ dài rất lớn (lên đến trên 8000 token), nhưng đây là các trường hợp ngoại lệ và chiếm tỷ lệ rất nhỏ.
- Phân bố nghiêng phải (right-skewed), đặc trưng của dữ liệu văn bản tin tức.

Nhận xét:

- Độ dài trung bình khá cao, không phù hợp để đưa toàn bộ văn bản vào mô hình Transformer (giới hạn 512 token).
- Vì vậy, trong quá trình tiền xử lý, mô hình DistilBERT chỉ lấy tối đa `max_length = 128` token đầu tiên — đủ để mô hình nắm bắt ngữ cảnh chính, đồng thời đảm bảo hiệu năng tính toán.

5.2 Metric đánh giá

Trong đề tài, các chỉ số đánh giá được tính thông qua hàm `compute_metrics()` như sau:

```
def compute_metrics(eval_pred):
    logits, labels = eval_pred
    preds = np.argmax(logits, axis=-1)
    acc = accuracy_score(labels, preds)
    precision, recall, f1, _ = precision_recall_fscore_support(
        labels, preds, average="binary"
    )
    return {"accuracy": acc, "precision": precision, "recall": recall, "f1": f1}
```

Hàm này nhận đầu vào là:

- `logits`: đầu ra thô từ mô hình (chưa qua softmax)
- `labels`: nhãn thật tương ứng

Dự đoán cuối cùng được tính bằng:

$$\hat{y} = \arg\max_k(\text{logits}_k)$$

Trong đó:

- $\hat{y}$: nhãn mô hình dự đoán
- $k \in \{0, 1\}$
- lớp **1** được xem là lớp dương (`fake news`)

1. Accuracy

Tỷ lệ dự đoán đúng trên toàn bộ mẫu:

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

2. Precision

Xác suất dự đoán `fake` là đúng:

$$\text{Precision} = \frac{TP}{TP + FP}$$

3. Recall

Khả năng phát hiện đúng các mẫu tin giả:

$$\text{Recall} = \frac{TP}{TP + FN}$$

4. F1-score

Trung bình điều hoà giữa Precision và Recall (giảm ảnh hưởng của mất cân bằng nhãn):

$$F1 = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

5. Giải thích các toán tử và ký hiệu

- TP (True Positive): tin giả được dự đoán đúng là giả.
- TN (True Negative): tin thật được dự đoán đúng là thật.
- FP (False Positive): tin thật bị dự đoán nhầm thành giả.
- FN (False Negative): tin giả bị dự đoán nhầm thành thật.
- arg max: toán tử chọn chỉ số của phần tử có giá trị lớn nhất.
- `average="binary"`: chỉ tính metric cho lớp dương (label = 1).

6. Các chỉ số bổ sung

Ngoài các chỉ số trên, trong phần đánh giá mô hình còn sử dụng:

- **Confusion Matrix** – phân tích lỗi và phân bố dự đoán.
- **Classification Report** – bảng chi tiết Precision, Recall, F1 của từng lớp.

5.3 Kết quả thực nghiệm

5.3.1 1. K-fold Cross Validation

Để đánh giá độ ổn định của mô hình và khả năng tổng quát hóa trên dữ liệu mới, kỹ thuật **Stratified K-Fold** với `n_splits = 3` được sử dụng:

```
skf = StratifiedKFold(n_splits=3, shuffle=True)
```

Ở mỗi fold:

- Mô hình DistilBERT được fine-tune trong **1 epoch**.
- Tập validation thay đổi ở mỗi fold nhưng vẫn giữ nguyên tỉ lệ lớp.
- Các chỉ số Accuracy, Precision, Recall, F1 được tính bằng hàm `compute_metrics()`.

Toàn bộ kết quả được lưu trong danh sách `fold_metrics`, sau đó tính trung bình:

```
for m in fold_metrics[0].keys():
    print(mean, std)
```

Kết quả tổng hợp Cross-Validation

Kết quả trung bình sau 3 fold như sau:

$$\text{Accuracy}_{avg} = 0.9994 \pm 0.0002$$
$$\text{Precision}_{avg} = 0.9991 \pm 0.0004$$
$$\text{Recall}_{avg} = 0.9998 \pm 0.0002$$
$$\text{F1}_{avg} = 0.9994 \pm 0.0002$$

Điều này cho thấy mô hình có độ ổn định rất cao trên các tập chia khác nhau, đặc biệt không có hiện tượng overfitting đáng kể giữa các fold.

5.3.2 2. Kết quả huấn luyện toàn bộ mô hình

Sau Cross-Validation, mô hình được huấn luyện lại trên toàn bộ tập train với **3 epoch**. Log chi tiết được ghi nhận như sau:

- Epoch 1: Training Loss = 0.0004, Validation Loss = 0.002882, Accuracy = 0.999554
- Epoch 2: Training Loss = 0.0105, Validation Loss = 0.007354, Accuracy = 0.998441
- Epoch 3: Training Loss = 0.0000, Validation Loss = 0.002205, Accuracy = 0.999777

Kết quả cho thấy:

- Sai số huấn luyện (training loss) giảm mạnh qua từng epoch.
- Sai số kiểm định giữ mức thấp và ổn định.
- Các chỉ số đánh giá gần như tuyệt đối.

Thông tin từ quá trình huấn luyện:

```
TrainOutput(global_step=7575,
training_loss=0.004263,
train_runtime=916.7s,
train_steps_per_second=8.263)
```

5.3.3 3. Kết quả trên tập kiểm thử (Hold-out Test)

Sau khi mô hình hoàn tất huấn luyện, tập kiểm thử cuối cùng được sử dụng để đánh giá hiệu năng thực tế:

$$\text{Accuracy} = 0.9998$$

$$\text{Precision} = 0.9996$$

$$\text{Recall} = 1.0000$$

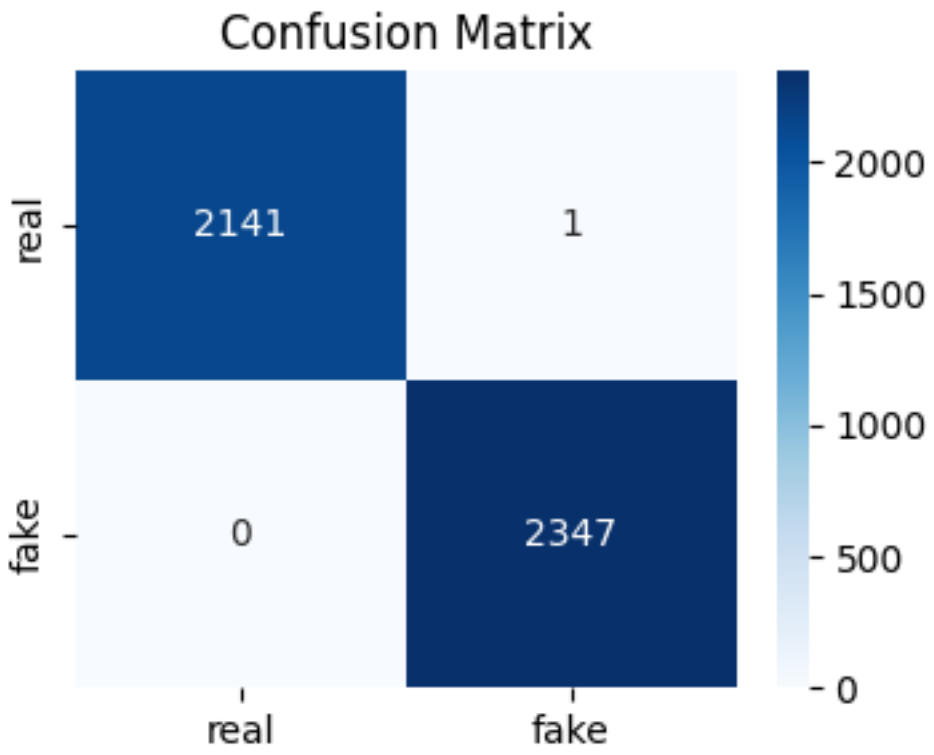
$$\text{F1} = 0.9998$$

Các chỉ số đều đạt gần mức tuyệt đối, cho thấy DistilBERT đã học được đặc trưng rất tốt từ dữ liệu.

5.3.4 4. Confusion Matrix

Biểu đồ Confusion Matrix (Hình 5.3) cho thấy mô hình dự đoán chính xác gần như tuyệt đối:

	real	fake
real	2141	1
fake	0	2347



Hình 5.3: Confusion Matrix của mô hình trên tập test

Không có mẫu `fake` nào bị dự đoán nhầm thành `real`, chứng tỏ Recall đạt tuyệt đối.

5.3.5 5. Báo cáo phân loại (Classification Report)

	precision	recall	f1-score	support
0 (real)	1.00	1.00	1.00	2142
1 (fake)	1.00	1.00	1.00	2347
accuracy	1.00	1.00	1.00	4489
macro avg	1.00	1.00	1.00	4489
weighted avg	1.00	1.00	1.00	4489

Báo cáo phân loại cho thấy mọi chỉ số đều đạt mức hoàn hảo.

5.3.6 6. Nhận xét chung

- DistilBERT cho hiệu suất vượt trội trên bài toán phát hiện tin giả.
- Các chỉ số Precision/Recall đều tiệm cận 1.0, đặc biệt Recall cho lớp `fake` đạt tuyệt đối.
- Mô hình hoạt động ổn định trên K-fold và trên tập test cuối.
- Kiến trúc DistilBERT với năng lực hiểu ngữ cảnh sâu giúp phân biệt tin giả – thật một cách hiệu quả hơn nhiều so với các mô hình truyền thống.

Chương 6

Triển khai mô hình phát hiện tin giả

6.1 Giới thiệu

Sau khi hoàn tất quá trình huấn luyện và đánh giá mô hình DistilBERT, bước tiếp theo là triển khai mô hình dưới dạng một ứng dụng trực quan để người dùng có thể kiểm tra tin tức và xem giải thích cho dự đoán của mô hình.

Trong đề tài này, mô hình được triển khai bằng **Streamlit**, hoạt động hoàn toàn trên CPU và tích hợp **Integrated Gradients (IG)** thông qua thư viện **Captum** nhằm giải thích mức độ đóng góp của từng token trong văn bản.

6.2 Kiến trúc triển khai

Ứng dụng triển khai có ba thành phần chính:

- **Frontend**: xây dựng bằng Streamlit, cung cấp giao diện nhập văn bản, nút dự đoán và hiển thị giải thích.
- **Backend**: mô hình DistilBERT đã fine-tune, được load bằng `AutoModelForSequenceClassification`.
- **Explainability Module**: sử dụng kỹ thuật *Integrated Gradients* từ Captum để tính mức độ ảnh hưởng của từng token.

Toàn bộ tiến trình được thiết kế sao cho mô hình chạy trực tiếp trên CPU mà không cần GPU, phù hợp với môi trường triển khai nhẹ.

6.3 Nạp mô hình và Tokenizer

Ứng dụng sử dụng decorator:

```
@st.cache_resource
```

để lưu trữ mô hình và tokenizer trong bộ nhớ nhằm giảm thời gian load:

- Mô hình: `DistilBertForSequenceClassification`
- Tokenizer: `AutoTokenizer`

- Device: CPU

$$\theta = \text{DistilBERT\_fine\_tuned}$$

Sau khi được nạp, mô hình được thiết lập ở chế độ **eval** để tắt dropout và các thành phần liên quan đến huấn luyện.

6.4 Quy trình dự đoán

Khi người dùng nhập văn bản, hệ thống thực hiện:

1. Tokenization với:

$$\text{max\_length} = 128, \quad \text{padding=“max\_length”}$$

2. Mã hoá thành tensor PyTorch.

3. Forward qua mô hình để lấy logits:

$$\hat{y} = f_\theta(x)$$

4. Tính xác suất bằng softmax:

$$P = \text{softmax}(\hat{y})$$

5. Label:

$$0 = \text{Real}, \quad 1 = \text{Fake}$$

Kết quả được hiển thị dưới dạng phần trăm theo từng lớp.

6.5 Giải thích mô hình bằng Integrated Gradients

6.5.1 Cơ chế hoạt động

IG là kỹ thuật dựa trên gradient để đo mức độ ảnh hưởng của từng đầu vào lên đầu ra. Với văn bản, mỗi token có embedding E_i, IG tính:

$$IG_i = (E_i - E_i') \int_{\alpha=0}^{1} \frac{\partial F(E' + \alpha(E - E'))}{\partial E_i} d\alpha$$

Trong đó:

- E: embedding thực
- E': baseline (ở đây là chuỗi toàn PAD token)
- F: xác suất lớp Fake News do mô hình dự đoán

6.5.2 Lý do chọn Integrated Gradients

IG được sử dụng vì:

- Phù hợp với mô hình Transformer.
- Không yêu cầu chỉnh sửa mô hình.
- Giải thích mượt, ít nhiễu hơn so với gradient thông thường.
- Khả năng phân bổ mức độ quan trọng cho **từng token**, giúp người dùng hiểu mô hình đang nhìn vào đâu.

6.5.3 Quy trình tính IG

1. Tạo baseline vector gồm toàn token PAD.
2. Gọi `LayerIntegratedGradients` lên embedding layer của DistilBERT.
3. Tính attributions với:
$$n\_steps = 50$$
4. Gộp attribution theo trục embedding:
$$A_i = \sum_{d=1}^{768} IG_{i,d}$$
5. Chuẩn hóa và tô màu token:
    - Đỏ: đẩy về Fake News
    - Xanh: đẩy về Real News

Kết quả được hiển thị bằng HTML tùy chỉnh trong Streamlit.

6.6 Giao diện người dùng

Hình 6.1: Giao diện người dùng.

Giao diện gồm ba phần chính:

1. Nhập văn bản

Người dùng dán nội dung bài báo cần kiểm tra.

2. Dự đoán

Ứng dụng trả về:

- Xác suất Real News
- Xác suất Fake News
- Nhãn dự đoán cuối cùng

3. Giải thích dự đoán

Ứng dụng hiển thị dòng văn bản với token được tô màu theo mức độ ảnh hưởng:

Màu đỏ → tăng xác suất Fake

Màu xanh → tăng xác suất Real

Người dùng có thể xem trực quan tại sao mô hình đưa ra kết luận.

6.7 Cấu trúc tệp triển khai

Toàn bộ code triển khai được đặt trong tệp:

`app.py`

Các thành phần chính:

- Load model + tokenizer
- Predict function
- Integrated Gradients function
- HTML visualization
- Streamlit UI

6.8 Kết luận

Chương này trình bày toàn bộ quá trình triển khai ứng dụng phát hiện tin giả dựa trên mô hình DistilBERT đã fine-tuning. Tích hợp kỹ thuật giải thích IG giúp hệ thống không chỉ dự đoán mà còn cung cấp khả năng giải thích rõ ràng, trực quan. Ứng dụng đáp ứng tốt yêu cầu thực tế khi triển khai mô hình NLP trong các hệ thống hỗ trợ quyết định.

Chương 7

Kết luận

Tóm tắt kết quả

Trong báo cáo này, mô hình **DistilBERT** đã được nghiên cứu, huấn luyện và triển khai cho bài toán **phát hiện tin giả (Fake News Detection)** dựa trên bộ dữ liệu 44 898 mẫu lấy từ Kaggle. Mô hình được tinh chỉnh (fine-tuning) với quy trình đầy đủ gồm: tiền xử lý dữ liệu, phân tích thống kê, tokenization, xây dựng pipeline huấn luyện, K-fold Cross Validation và đánh giá cuối cùng trên tập Hold-out.

Kết quả thực nghiệm cho thấy mô hình đạt hiệu năng xuất sắc:

- **Accuracy (K-fold avg):** 0.9994 ± 0.0002
- **Precision (K-fold avg):** 0.9991 ± 0.0004
- **Recall (K-fold avg):** 0.9998 ± 0.0002
- **F1-score (K-fold avg):** 0.9994 ± 0.0002

Trên tập kiểm tra cuối cùng (Hold-out):

- **Accuracy:** 0.9998
- **Precision:** 0.9996
- **Recall:** 1.0000
- **F1-score:** 0.9998

Ma trận nhầm lẫn cho thấy mô hình hầu như không mắc lỗi, thể hiện khả năng tổng quát hóa rất tốt.

Những đóng góp chính

Báo cáo đã xây dựng một pipeline end-to-end hoàn chỉnh:

- Phân tích và xử lý 44 898 tin tức thật/giả.
- Áp dụng mô hình **DistilBERT**, với ưu điểm nhẹ, nhanh, nhưng độ chính xác cao.

- Huấn luyện mô hình bằng **Transformers Trainer**.
- Đánh giá mô hình bằng các chỉ số chuẩn và **Stratified K-fold**.
- Xây dựng ứng dụng **Streamlit** hoàn chỉnh:
  - Phân loại tin giả theo thời gian thực.
  - Giải thích mô hình bằng **Integrated Gradients (Captum)**.
  - Minh họa mức độ đóng góp của từng token bằng màu sắc.

Ý nghĩa thực tiễn

Mô hình có thể được ứng dụng cho:

- Hệ thống lọc tin tức trong truyền thông.
- Công cụ kiểm tra tin giả cho nhà báo hoặc người dùng phổ thông.
- Tích hợp vào nền tảng mạng xã hội để cảnh báo nội dung độc hại.

Việc kết hợp giữa DistilBERT và kỹ thuật diễn giải IG giúp hệ thống không chỉ mạnh về mặt dự đoán mà còn **minh bạch**, phù hợp các yêu cầu "Explainable AI".

Hạn chế

Mặc dù đạt kết quả rất cao, mô hình vẫn tồn tại một số hạn chế:

- Dữ liệu chủ yếu là tiếng Anh, không bao phủ đa ngôn ngữ.
- Nội dung tin giả trong bộ dữ liệu mang tính báo chí, chưa bao gồm social media.
- Mô hình chỉ phân loại nhị phân, chưa phân nhóm dạng tin giả (giật tít, sai sự thật, xuyên tạc,...).

Hướng phát triển trong tương lai

Trong tương lai, có thể mở rộng theo các hướng:

- Huấn luyện mô hình **đa ngôn ngữ** (mBERT, XLM-R).
- Thu thập thêm dữ liệu từ mạng xã hội để tăng tính thực tế.
- Áp dụng mô hình lớn hơn (RoBERTa, DeBERTa) để so sánh hiệu năng.
- Phân tích chuyên sâu bằng các kỹ thuật XAI khác như LIME, SHAP.
- Phân loại đa nhãn: *"tin bịa đặt", "tin xuyên tạc", "tin lừa đảo"*...

Kết quả đạt được chứng minh rằng **DistilBERT** là một mô hình mạnh mẽ, hiệu quả và phù hợp cho bài toán phát hiện tin giả. Sự kết hợp giữa kiến trúc Transformer, quy trình huấn luyện chuẩn hóa và công cụ giải thích giúp mô hình vừa **chính xác** vừa **dễ hiểu**, đáp ứng tốt yêu cầu thực tiễn trong việc chống lan truyền thông tin sai lệch.

Tài liệu tham khảo

1. Hu, L., Wei, S., Zhao, Z., Wu, B. (2022). Deep learning for fake news detection: A comprehensive survey. *AI Open*.

2. Mishima, K., Yamana, H. (2022). A survey on explainable fake news detection. *IEICE Transactions on Information and Systems*, E105D(7), 1249-1257.

3. Yadav, A., Gaba, S., Khan, H., Budhiraja, I., Singh, A., Singh, K. K. (2022). ETMA: Efficient Transformer-based Multilevel Attention framework for multimodal fake news detection. Preprint arXiv:2206.07331.

4. Yin, S., Gao, C., Wang, Z. (2023). GAMC: An Unsupervised Method for Fake News Detection using Graph Autoencoder with Masking. Preprint arXiv:2312.05739.

5. Alshuwaier, F. A., Alsulaiman, F. A. (2025). Fake News Detection Using Machine Learning and Deep Learning Algorithms: A Comprehensive Review and Future Perspectives. *Computers*, 14(9), 394.