ĐẠI HỌC QUỐC GIA TPHCM

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

KHOA CÔNG NGHỆ THÔNG TIN

Nghiên cứu và xây dựng các mô hình

Đề tài: Hệ thống tự động gán nhãn bài báo tin tức tiếng Việt

Môn học: Nhập môn học máy

Sinh viên thực hiện:

Tô Hữu Danh (23127336)

Nguyễn Đăng Hưng (23127050)

Lê Phú Cường (23127164)

Nguyễn Bá Đăng Khoa (23127392)

Giáo viên hướng dẫn:

TS. Bùi Tiến Lên

Ngày 15 tháng 12 năm 2025

Mục lục

1 Giới thiệu bài toán **1**

2 Tổng quan về dữ liệu đầu vào **2**

2.1 Dữ liệu huấn luyện 2

2.2 Tiền xử lý 2

3 Lựa chọn mô hình và kiến trúc **4**

3.1 Mô hình sử dụng 4

3.2 Lý do lựa chọn 4

3.3 Kiến trúc chi tiết (Đối với Deep Learning) 5

3.3.1 FastText 5

3.3.2 PhoBERT (Transformer) 7

4 Cấu hình huấn luyện **9**

4.1 FastText 9

4.2 PhoBERT [7] 10

4.3 TF-IDF + SVM 11

5 Kết quả thực nghiệm **14**

5.1 TF-IDF + Linear SVM 14

5.1.1 Biểu đồ quá trình học (Learning Curves) 14

5.1.2 Đánh giá trên tập kiểm thử (Test set) 16

5.1.3 Ma trận nhầm lẫn (Confusion Matrix) 17

5.2 FastText 18

5.3 PhoBERT 21

5.3.1 Biểu đồ quá trình học 21

5.3.2 Đánh giá trên tập kiểm thử 22

5.4 Ma trận nhầm lẫn (Confusion Matrix) 23

6 Thảo luận và Phân tích lỗi **25**

6.1 Hiện tượng Overfitting và Underfitting 25

6.2 Các biện pháp khắc phục đã thực hiện 25
6.3 Phân tích các trường hợp sai (Error Analysis) 26
6.4 So sánh hiệu năng các mô hình 26

7 Tài liệu tham khảo **28**

Danh sách bảng

1 Các siêu tham số của mô hình FastText 9
2 Bảng mô tả siêu tham số cho mô hình PhoBERT 10
3 Các siêu tham số và thiết lập huấn luyện cho mô hình TF-IDF + SVM 12
4 Bảng so sánh hiệu năng giữa các mô hình trên tập Test 27

Danh sách hình vẽ

1 Sơ đồ kiến trúc FastText 6
2 Sơ đồ kiến trúc PhoBERT 7
3 Learning Curve cho Loss theo từng epoch (TF-IDF + Linear SVM via SGD) 14
4 Learning Curve cho Accuracy theo từng epoch (TF-IDF + Linear SVM via SGD) . 15
5 Các chỉ số đánh giá tổng thể trên tập test của TF-IDF + SVM 16
6 Confusion matrix của TF-IDF + SVM trên tập Test 17
7 Hình learning curves của FastText 18
8 Các chỉ số đánh giá tổng thể trên tập test của FastText 19
9 Confusion matrix của FastText trên tập Test 20
10 Learning Curve cho Loss theo từng epoch 21
11 Learning Curve cho Accuracy theo từng epoch 22
12 Các chỉ số đánh giá tổng thể trên tập test của PhoBERT 23
13 Ma trận nhầm lẫn cho PhoBERT 24

1 Giới thiệu bài toán

Trong bối cảnh bùng nổ thông tin số, các tòa soạn báo điện tử và hệ thống tổng hợp tin tức tại Việt Nam đang đối mặt với thách thức lớn trong việc quản lý và phân phối nội dung. Các phương pháp phân loại thủ công truyền thống không còn đáp ứng được yêu cầu về tốc độ, khả năng mở rộng và thường dễ xảy ra sai sót do yếu tố chủ quan. Bên cạnh đó, nhu cầu tiếp cận thông tin của độc giả hiện đại đã chuyển dịch sang xu hướng tìm kiếm chủ động theo từ khóa và chủ đề cụ thể thay vì duyệt thụ động theo chuyên mục tĩnh.

Nhóm nghiên cứu xác định bài toán cần giải quyết là **Phân loại văn bản tiếng Việt đa lớp (Multi-class Vietnamese Text Classification)**.

Mục tiêu của bài toán là xây dựng một hệ thống học máy có khả năng:

- Tiếp nhận đầu vào là văn bản thô (raw text) hoặc đường dẫn (URL) của một bài báo tiếng Việt.
- Tự động phân tích ngữ nghĩa và gán nhãn chủ đề chính xác nhất cho bài báo đó dựa trên 7 nhãn chuyên mục đã định nghĩa trước: **Thể thao, Thời sự, Chính trị, Kinh tế, Giáo dục, Sức khoẻ, Thế giới**.
- Giải quyết bài toán dưới góc độ so sánh hiệu năng giữa các phương pháp học máy truyền thống (TF-IDF + SVM) và các mô hình học sâu hiện đại (FastText, PhoBERT) để tìm ra kiến trúc tối ưu nhất cho đặc thù ngôn ngữ tiếng Việt .

2 Tổng quan về dữ liệu đầu vào

Bộ dữ liệu sử dụng cho việc huấn luyện và đánh giá mô hình bao gồm **20,777** bài báo được thu thập từ báo Thanh Niên, sau khi đã qua các bước làm sạch và loại bỏ nhiễu.

2.1 Dữ liệu huấn luyện

Dữ liệu được phân chia thành 3 tập riêng biệt: **Train**, **Validation**, và **Test** với tỷ lệ 80/10/10. Cụ thể:

- **Tập Train (80%):** Dùng để huấn luyện các tham số của mô hình.
- **Tập Validation (10%):** Dùng để tinh chỉnh siêu tham số (hyperparameter tuning) và đánh giá trong quá trình huấn luyện nhằm tránh overfitting.
- **Tập Test (10%):** Dùng để đánh giá khách quan hiệu năng cuối cùng của mô hình.

Lý do lựa chọn tỷ lệ này:

- Với kích thước bộ dữ liệu tương đối lớn (gần 21,000 mẫu), việc dành 80% cho tập huấn luyện giúp mô hình học được tối đa các đặc trưng từ vựng phong phú của tiếng Việt.
- Tỷ lệ 10% cho tập Validation và Test tương ứng với khoảng hơn 2,000 mẫu cho mỗi tập. Đây là con số đủ lớn về mặt thống kê để đảm bảo độ tin cậy của kết quả đánh giá mà không gặp phải phương sai quá lớn.
- Nhóm áp dụng kỹ thuật Lấy mẫu phân tầng (Stratified Sampling) khi chia dữ liệu để đảm bảo tỷ lệ phân bố của 7 nhãn chủ đề là đồng đều trên cả 3 tập, tránh hiện tượng lệch dữ liệu (data shift) gây ảnh hưởng đến kết quả kiểm thử.

2.2 Tiền xử lý

Quy trình tiền xử lý được thiết kế chặt chẽ để chuyển đổi dữ liệu thô thành dạng vector phù hợp cho từng loại kiến trúc mô hình. Các bước chính bao gồm:

Bước 1: Làm sạch dữ liệu (Data Cleaning)

- Loại bỏ các thẻ HTML, CSS, JavaScript rác và các ký tự đặc biệt không mang ngữ nghĩa.
- Xử lý các giá trị thiếu (missing values) và loại bỏ hoàn toàn các bài báo trùng lặp nội dung để tránh rò rỉ dữ liệu (data leakage).
- Chuẩn hóa bảng mã về định dạng Unicode NFC thống nhất cho toàn bộ văn bản.

Bước 2: Xử lý đặc thù theo mô hình (Model-specific Preprocessing) Nhận thấy sự khác biệt về cơ chế hoạt động giữa các mô hình, nhóm chia luồng xử lý thành hai hướng riêng biệt:

- **Đối với mô hình truyền thống (SVM, FastText):**
  - **Chuẩn hóa ký tự thường (Lowercasing):** Đưa toàn bộ văn bản về chữ thường để giảm chiều dữ liệu của bộ từ điển.
  - **Tách từ (Word Segmentation):** Sử dụng thư viện `pyvi` [1] để nhận diện và ghép các từ ghép tiếng Việt (ví dụ: `"học_máy"`, `"trí_tuệ_nhân_tạo"`).
  - **Loại bỏ từ dừng (Stopword Removal):** Lọc bỏ các từ hư từ, từ chức năng ít mang ý nghĩa phân loại dựa trên danh sách từ dừng tiếng Việt chuẩn.
- **Đối với mô hình Transformer (PhoBERT):**
  - **Giữ nguyên định dạng gốc:** Không chuyển về chữ thường và không loại bỏ từ dừng. Điều này nhằm bảo toàn cấu trúc ngữ pháp và ngữ cảnh (context), giúp cơ chế Attention của PhoBERT nắm bắt tốt hơn ý nghĩa của câu.
  - **Tách từ (Word Segmentation):** Vẫn thực hiện tách từ bằng công cụ chuyên dụng (**VnCoreNLP** [2]) để đồng bộ với cách thức mà PhoBERT đã được tiền huấn luyện.

Bước 3: Mã hóa nhãn (Label Encoding) Các nhãn dạng văn bản (String) được ánh xạ sang dạng số nguyên (Integer) từ 0 đến 6 để phục vụ cho việc tính toán hàm mất mát (Loss function).

3 Lựa chọn mô hình và kiến trúc

3.1 Mô hình sử dụng

Dự án triển khai với ba hướng tiếp cận chính:

1. TF-IDF + Logistic Regression (Baseline): Văn bản được biểu diễn bằng vector TF-IDF, sau đó huấn luyện mô hình Logistic Regression để phân loại.

2. FastText[3]: Sử dụng mô hình học sâu để biểu diễn từ và huấn luyện nhanh, nhằm kiểm chứng hiệu quả trên tiếng Việt.

3. PhoBERT (Transformer): Mô hình PhoBERT của VinAI được fine-tune trên bộ dữ liệu thu thập, kỳ vọng đạt hiệu năng cao nhất.

Các mô hình được huấn luyện bằng Python với các thư viện scikit-learn, PyTorch và transformers [4, 5, 6].

3.2 Lý do lựa chọn

Bài toán của dự án là gắn nhãn tự động bài báo tiếng Việt theo dạng phân loại đơn lớp (single-label classification), trong đó mỗi bài báo chỉ thuộc một trong bảy nhãn (Chính trị, Kinh tế, Giáo dục, ...). Tập dữ liệu gồm khoảng 20,000 bài báo thu thập từ Báo Thanh Niên, với nội dung văn bản tương đối dài, giàu ngữ cảnh và có nhiều chủ đề gần nhau về mặt ngữ nghĩa. Do đó, nhóm lựa chọn các mô hình theo lộ trình từ đơn giản đến phức tạp nhằm vừa đảm bảo khả năng kiểm chứng, vừa tối ưu chất lượng dự đoán.

1. TF-IDF + Logistic Regression

   Mô hình TF-IDF + Logistic Regression được sử dụng làm baseline cho bài toán phân loại đơn lớp vì Logistic Regression là phương pháp kinh điển trong phân loại chủ đề, hoạt động hiệu quả khi mỗi văn bản chỉ thuộc một nhãn. Các chuyên mục báo chí thường có tập từ vựng đặc thù; biểu diễn TF-IDF giúp mô hình phân biệt các nhãn dựa trên sự xuất hiện và mức độ quan trọng của từ khóa. Với quy mô 50k bài báo, mô hình tuyến tính cho phép huấn luyện và tinh chỉnh siêu tham số nhiều lần mà không tốn nhiều tài nguyên. Ngoài ra, trọng số đặc

trưng của từng lớp giúp phân tích đặc điểm ngôn ngữ của mỗi nhãn và phát hiện các trường hợp gán nhãn chưa chính xác.

2. FastText

   FastText được sử dụng như một bước mở rộng từ mô hình tuyến tính sang học sâu cho bài toán phân loại đơn lớp. Mô hình sử dụng embedding kèm cơ chế subword giúp nhận diện các bài viết cùng chủ đề dù khác cách diễn đạt, đặc biệt phù hợp với tiếng Việt. So với Transformer, FastText có kiến trúc nhẹ hơn nhiều, cho phép đánh giá mức cải thiện so với TF-IDF trước khi đầu tư fine-tune mô hình lớn.

3. PhoBERT (Transformer)

   PhoBERT được lựa chọn là mô hình chính nhằm đạt chất lượng dự đoán cao nhất. Cơ chế self-attention cho phép mô hình học quan hệ giữa các từ trong toàn bộ bài báo, đặc biệt quan trọng với các chủ đề có nội dung chồng lấn. PhoBERT đã được huấn luyện trước trên tập dữ liệu tiếng Việt lớn, giúp mô hình có kiến thức ngôn ngữ nền và thích nghi nhanh khi fine-tune. Trong các trường hợp tiêu đề gây nhiễu hoặc cách diễn đạt mới, PhoBERT thường ổn định hơn so với TF-IDF và FastText.

3.3 Kiến trúc chi tiết (Đối với Deep Learning)

Đồ án sử dụng hai mô hình Deep Learning là FastText và PhoBERT. Mô hình TF-IDF + Logistic Regression không được mô tả trong phần này do không thuộc nhóm Deep Learning.

3.3.1 FastText

FastText được sử dụng như một mô hình học sâu nhẹ, đóng vai trò trung gian giữa baseline tuyến tính và Transformer.

Luồng xử lý dữ liệu Quy trình xử lý và suy luận của FastText gồm các bước sau:

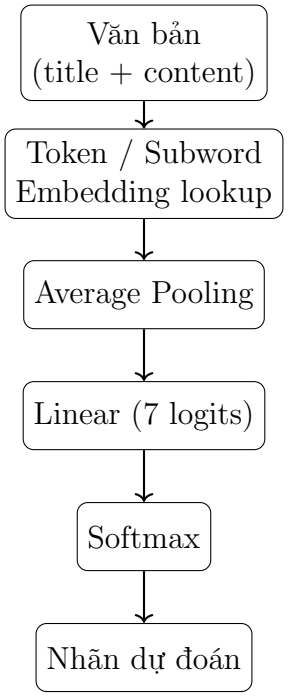
1. Văn bản đầu vào được làm sạch và ghép từ `title` và `content`, sau đó chuyển sang định dạng huấn luyện FastText với tiền tố nhãn `__label__`.

2. Mỗi từ được ánh xạ thành vector embedding thông qua cơ chế subword n-gram.

3. Các embedding của toàn bộ câu được trung bình để tạo biểu diễn vector cố định cho văn bản.

4. Vector này được đưa qua một tầng tuyến tính để sinh logits cho 7 nhãn.

5. Softmax được áp dụng để chọn nhãn có xác suất cao nhất.



Hình 1: Sơ đồ kiến trúc FastText

Sơ đồ kiến trúc

Số lượng tham số FastText không có kiến trúc tầng sâu. Số lượng tham số chủ yếu đến từ:

$$|V_{\text{subword}}| \times d + 7 \times d$$

trong đó d là kích thước embedding (thường từ 100 đến 300). Với từ điển subword quy mô vài trăm nghìn, mô hình có kích thước vài chục triệu tham số. Phiên bản quantized được sử dụng trong dự án giới hạn dung lượng khoảng 20MB.

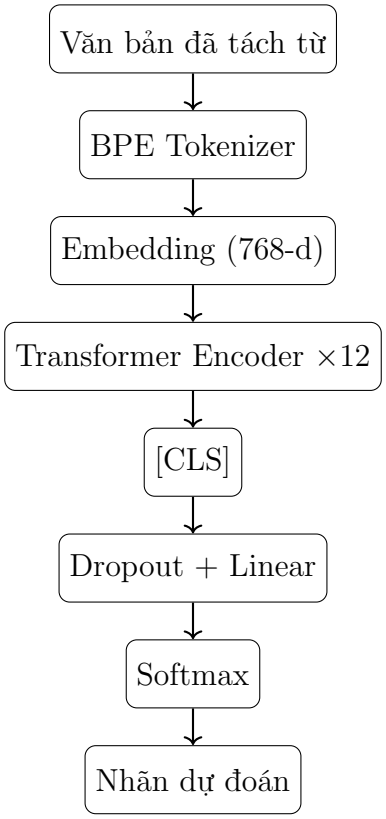
Hàm kích hoạt FastText không sử dụng hàm kích hoạt phi tuyến giữa các tầng; chỉ áp dụng Softmax tại tầng đầu ra.

3.3.2 PhoBERT (Transformer)

PhoBERT là mô hình Deep Learning chính của hệ thống, được fine-tune cho bài toán phân loại bài báo tiếng Việt.

Luồng xử lý dữ liệu Quy trình xử lý của PhoBERT được thực hiện như sau:

1. Văn bản đầu vào được tách từ bằng `PyVi` và token hóa bằng BPE, giới hạn tối đa 256 token.
2. Sinh các tensor `input_ids` và `attention_mask`.
3. Dữ liệu đi qua tầng embedding kích thước 768.
4. Qua 12 tầng Transformer encoder.
5. Vector ẩn của token `[CLS]` được trích xuất làm biểu diễn toàn văn bản.
6. Vector này đi qua dropout và tầng tuyến tính để sinh logits.
7. Softmax được áp dụng để suy ra nhãn cuối cùng.



Hình 2: Sơ đồ kiến trúc PhoBERT

Sơ đồ kiến trúc

Số lượng tham số PhoBERT-base có khoảng 135 triệu tham số, gồm 12 tầng encoder, hidden size 768, feed-forward size 3072, 12 attention heads và vocabulary BPE khoảng 64k token. Tầng phân loại bổ sung có ma trận trọng số kích thước 768×7 và bias 7.

Hàm kích hoạt PhoBERT sử dụng GELU trong các khối feed-forward của Transformer encoder và Softmax tại tầng đầu ra.

4 Cấu hình huấn luyện

4.1 FastText

Hệ thống sử dụng thư viện `fastText` cho bài toán phân loại đơn nhãn với 7 lớp. Dữ liệu được trích xuất và chuẩn hóa từ cơ sở dữ liệu SQLite (`dataset/articles.db`), trong đó mỗi mẫu được ghép từ trường `title` và `content`. Văn bản được chuyển về chữ thường, giữ lại các dấu câu cơ bản, sau đó gắn tiền tố nhãn `__label__` theo định dạng yêu cầu của fastText. Tập dữ liệu được chia theo tỷ lệ 70/15/15 cho các tập Train, Validation và Test (`train_fasttext.py`).

Tham số	Giá trị	Giải thích
Epochs	{5, 10, 15, 20, 25}	Huấn luyện với nhiều mốc epoch để quan sát quá trình hội tụ thông qua learning curves và chọn mô hình tốt nhất theo F1 trên Validation.
Learning Rate	0.3	Tốc độ học mặc định của fastText, phù hợp với cơ chế tối ưu SGD nhẹ.
Embedding dim	150	Kích thước vector biểu diễn từ và subword.
wordNgrams	2	Bổ sung thông tin bi-gram nhằm giữ lại ngữ cảnh cục bộ giữa các từ.
Subword	minn=2, maxn=5	Sử dụng ký tự n-gram để giảm hiện tượng từ ngoài từ điển (OOV) và cải thiện khả năng tổng quát hóa.
Loss	`softmax`	Phù hợp cho bài toán phân loại đơn lớp; LogLoss được sử dụng để theo dõi quá trình huấn luyện.

Bảng 1: Các siêu tham số của mô hình FastText

Chiến lược huấn luyện và đánh giá được thực hiện như sau:

- Huấn luyện mô hình với từng cấu hình số epoch, ghi lại Accuracy và LogLoss trên tập Train và Validation. Mô hình có F1 cao nhất trên Validation được lưu lại.
- Đánh giá trên tập Test với các chỉ số Accuracy, Precision, Recall và F1 (macro), đồng thời xuất classification report để phân tích chi tiết từng lớp. Confusion matrix được sử dụng để quan sát các nhãn dễ nhầm lẫn.
- Tiêu chí lựa chọn mô hình dựa trên F1 cao nhất trên Validation, kết hợp theo dõi LogLoss để kiểm tra mức độ hội tụ và phát hiện hiện tượng overfitting hoặc lệch pha giữa Train và

Validation.

4.2 PhoBERT [7]

Hệ thống được xây dựng dựa trên các thư viện mã nguồn mở tiêu chuẩn trong nghiên cứu Xử lý Ngôn ngữ Tự nhiên (NLP):

- **Framework:** Sử dụng PyTorch kết hợp với Hugging Face Transformers (API Trainer) để quản lý quy trình huấn luyện và đánh giá.
- **Xử lý dữ liệu:** Dữ liệu đầu vào được yêu cầu đã qua bước tách từ (segmentation) trước khi nạp vào mô hình, sử dụng trường thông tin `content_segmented` để đảm bảo tương thích với bộ từ điển của PhoBERT.

Quá trình huấn luyện được thực hiện với bộ tham số được tối ưu hoá như sau:

Tham số	Giá trị	Giải thích
Kiến trúc mô hình	`vinai/phobert-base`	Sử dụng phiên bản Base của PhoBERT, phù hợp với tài nguyên tính toán và quy mô dữ liệu.
Epochs	4	Số lần mô hình duyệt qua toàn bộ tập dữ liệu huấn luyện.
Batch size	16	Thiết lập cho cả tập huấn luyện và kiểm thử để phù hợp với bộ nhớ GPU.
Learning Rate	2×10^{-5}	Tốc độ học nhỏ giúp mô hình hội tụ ổn định, tránh dao động mạnh quanh điểm cực trị.
Độ dài chuỗi tối đa	256 tokens	Giới hạn độ dài văn bản đầu vào để cân bằng giữa ngữ cảnh và hiệu năng.

Bảng 2: Bảng mô tả siêu tham số cho mô hình PhoBERT

Bên cạnh đó, chiến lược tối ưu hoá (Optimization Strategy) cũng được áp dụng, cụ thể là lớp `TrainingArguments`:

- **Cơ chế Warm-up:** Áp dụng `warmup_steps=500`. Mô hình khởi động với tốc độ học thấp và tăng dần trong 500 bước đầu tiên để ổn định trọng số trước khi học với tốc độ tiêu chuẩn.
- **Regularization:** Sử dụng trọng số suy giảm (`weight_decay=0.01`) để hạn chế hiện tượng quá khớp (overfitting) bằng cách phạt các trọng số có giá trị quá lớn.

- **Chiến lược lưu trữ:** Sử dụng chiến lược `save_strategy="epoch"` và `load_best_model_at_end=True`, đảm bảo mô hình cuối cùng được chọn là mô hình có chỉ số F1 tốt nhất trên tập Validation chứ không phải mô hình tại epoch cuối cùng.

4.3 TF-IDF + SVM

Bên cạnh FastText và PhoBERT, hệ thống triển khai một mô hình học máy cổ điển (traditional ML) dựa trên đặc trưng TF-IDF kết hợp bộ phân loại Linear SVM (LinearSVC) cho bài toán phân loại đơn nhãn với 7 lớp. Dữ liệu được nạp từ các tệp CSV đã được tiền xử lý và chia sẵn theo tỷ lệ 70/15/15 cho Train/Validation/Test (`train/X_train_basic.csv`, `val/X_val_basic.csv`, `test/X_test_basic.csv`).

Mỗi mẫu văn bản sử dụng cột `content_final` (đã qua chuẩn hóa ở pipeline tiền xử lý). Nhãn tương ứng được lấy từ cột `label_encoded`. Các mẫu có nhãn bị thiếu (NaN) được loại bỏ trước khi huấn luyện để đảm bảo tính nhất quán dữ liệu.

Tham số	Giá trị	Giải thích
Biểu diễn văn bản	TF-IDF	Chuyển văn bản sang vector trọng số theo tần suất (TF) và độ hiếm (IDF), phù hợp với các mô hình tuyến tính.
ngram_range	(1,1) hoặc (1,2)	So sánh giữa unigram và unigram+bigram nhằm tăng khả năng nắm bắt ngữ cảnh cục bộ.
min_df	2	Loại bỏ các từ xuất hiện quá ít (nhiễu), giúp giảm kích thước không gian đặc trưng.
max_df	{0.8, 0.9, 1.0}	Loại bỏ từ quá phổ biến (ít phân biệt), được tinh chỉnh bằng Grid Search.
sublinear_tf	True	Áp dụng chuẩn hóa log cho TF để giảm ảnh hưởng của các từ lặp quá nhiều.
Bộ phân loại	LinearSVC	SVM tuyến tính tối ưu biên phân tách, phù hợp cho dữ liệu chiều cao và thưa như TF-IDF.
class_weight	balanced	Tự động cân bằng trọng số theo phân phối lớp để giảm thiên lệch về lớp phổ biến.
C (regularization)	{0.1, 0.5, 1.0, 2.0, 5.0}	Hệ số phạt điều khiển mức độ regularization: C lớn → ít phạt hơn, mô hình fit mạnh hơn.
Tối ưu siêu tham số	GridSearchCV (cv=3)	Dò lưới siêu tham số trên tập Train với 3-fold CV nội bộ, chọn theo `f1_macro`.
Metric lựa chọn	F1-macro	Phù hợp cho đa lớp và coi trọng đồng đều các lớp, hạn chế ảnh hưởng mất cân bằng.

Bảng 3: Các siêu tham số và thiết lập huấn luyện cho mô hình TF-IDF + SVM

Chiến lược huấn luyện và đánh giá được thực hiện như sau:

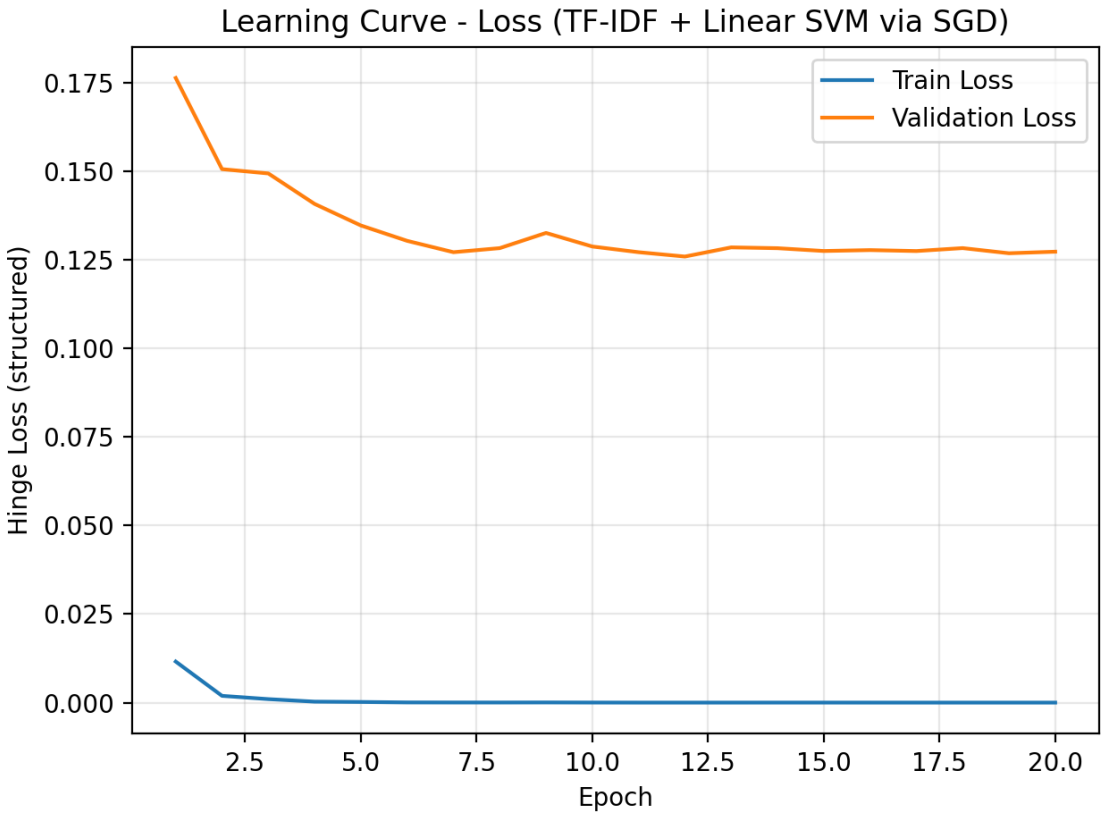
- **Huấn luyện & tinh chỉnh:** Xây dựng pipeline `TF-IDF` → `LinearSVC`. Toàn bộ siêu tham số được tinh chỉnh bằng `GridSearchCV` trên tập Train với 3-fold CV nội bộ, tiêu chí chọn là **F1-macro** nhằm tối ưu hiệu năng đồng đều giữa các lớp.
- **Đánh giá Validation:** Sau khi tìm được cấu hình tốt nhất từ CV, mô hình được đánh giá trên tập Validation bằng các chỉ số Accuracy, Precision, Recall, F1-macro và `classification_report` để phân tích chi tiết theo từng lớp.
- **Huấn luyện lại và kiểm thử:** Mô hình tốt nhất (`best_estimator_`) được train lại trên Train+Validation để tận dụng tối đa dữ liệu trước khi đánh giá cuối cùng trên tập Test.
- **Lưu mô hình:** Pipeline tốt nhất được lưu bằng `joblib` (`tfidf_svm.joblib`); đồng thời lưu

metadata ánh xạ nhãn (`tfidf_svm_labels.json`) phục vụ suy luận và hiển thị nhãn dễ đọc trong ứng dụng.

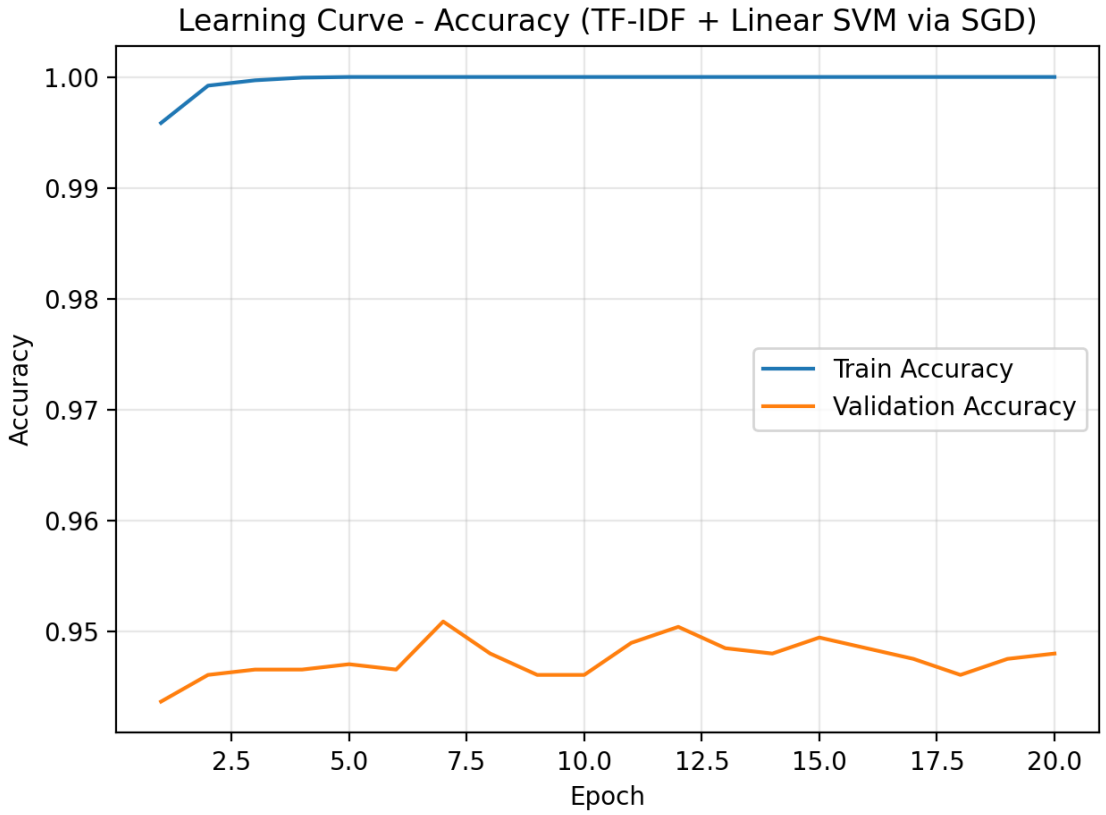
5 Kết quả thực nghiệm

5.1 TF-IDF + Linear SVM

5.1.1 Biểu đồ quá trình học (Learning Curves)



Hình 3: Learning Curve cho Loss theo từng epoch (TF-IDF + Linear SVM via SGD)



Hình 4: Learning Curve cho Accuracy theo từng epoch (TF-IDF + Linear SVM via SGD)

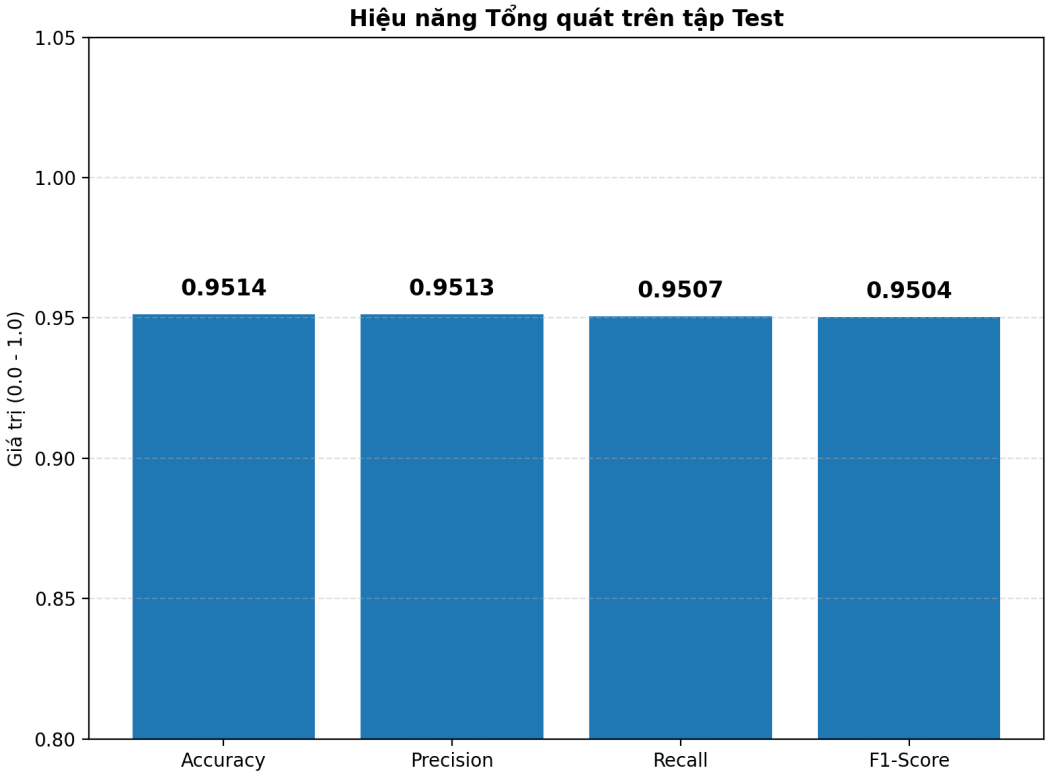
Nhận xét

- **Hội tụ và ổn định:** Validation Loss giảm mạnh trong vài epoch đầu (epoch 1–7) và sau đó gần như ổn định quanh mức $\sim 0.126 - 0.13$. Đồng thời Validation Accuracy tăng dần và dao động nhẹ quanh $\sim 0.946 - 0.951$, cho thấy mô hình đạt trạng thái hội tụ tương đối sớm và quá trình huấn luyện ổn định (không có dao động mạnh).

- **Khoảng cách Train–Validation:** Train Accuracy nhanh chóng đạt gần 100% từ rất sớm và Train Loss giảm gần về 0. Trong khi đó, Validation Accuracy chỉ tăng nhẹ rồi chững lại. Đây là dấu hiệu overfitting nhẹ (mô hình khớp rất tốt trên tập Train nhưng mức cải thiện trên Validation không tương ứng).

- **Lưu ý về cách ghi nhận loss theo epoch:** LinearSVC không cung cấp lịch sử loss theo từng epoch; do đó learning curves được xây dựng bằng mô hình thay thế `SGDClassifier(loss="hinge")` trên cùng đặc trưng TF-IDF để mô phỏng quá trình tối ưu của SVM tuyến tính và ghi nhận hinge loss theo epoch.

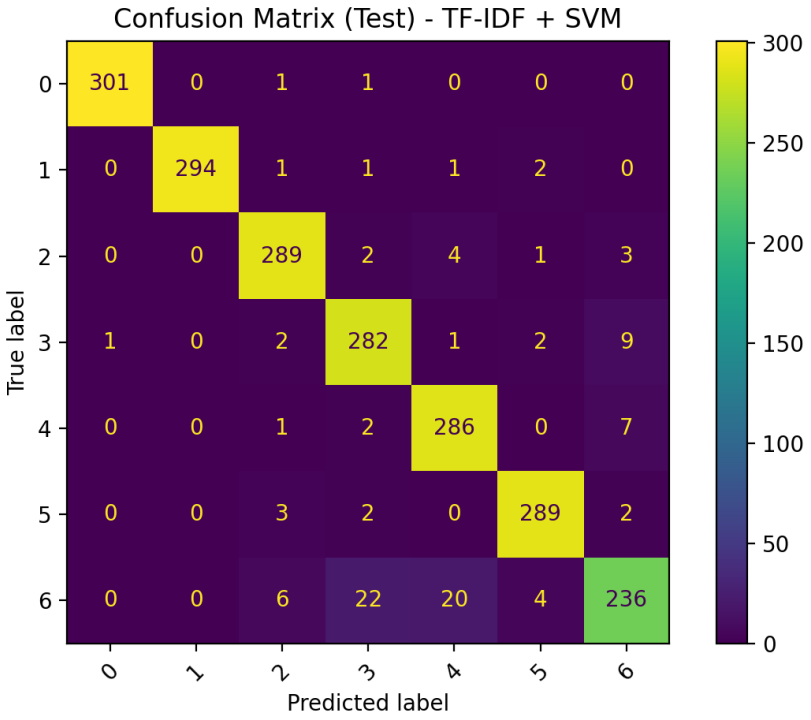
5.1.2 Đánh giá trên tập kiểm thử (Test set)



Hình 5: Các chỉ số đánh giá tổng thể trên tập test của TF-IDF + SVM

Trên tập Test, mô hình đạt Accuracy = 0.9514, Macro Precision = 0.9513, Macro Recall = 0.9507 và Macro F1 = 0.9504. Các chỉ số macro khá gần nhau, cho thấy mô hình hoạt động tương đối cân bằng giữa các lớp, không bị thiên lệch quá nhiều vào một nhãn cụ thể.

5.1.3 Ma trận nhầm lẫn (Confusion Matrix)



Hình 6: Confusion matrix của TF-IDF + SVM trên tập Test

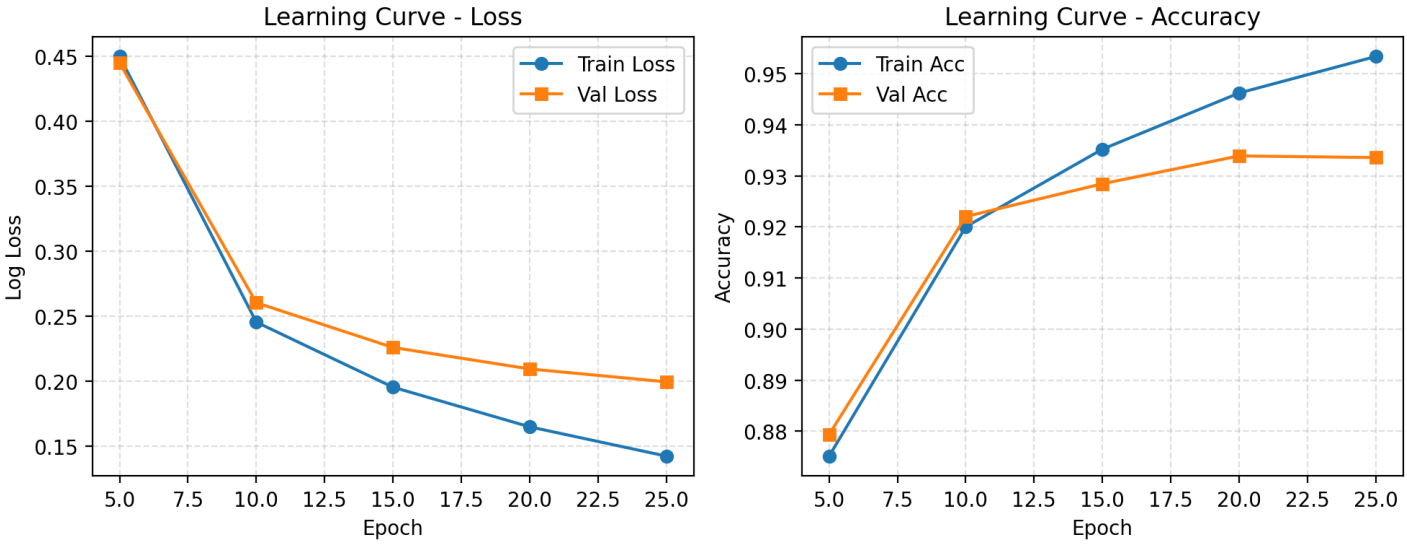
Phân tích

- Đường chéo chính chiếm đa số, phản ánh tỷ lệ dự đoán đúng cao trên hầu hết các lớp.
- **Lớp 0 và 1** có hiệu năng rất tốt (gần như không nhầm lẫn), số lượng dự đoán đúng trên đường chéo rất lớn.
- **Nhóm nhầm lẫn đáng chú ý nhất nằm ở lớp 6:** lớp 6 bị dự đoán nhầm sang lớp 3 và lớp 4 khá nhiều (ví dụ các ô (6,3) và (6,4) có giá trị cao), dẫn đến Recall của lớp 6 thấp hơn đáng kể so với các lớp còn lại (khoảng 0.82 theo classification report).
- Các lớp 2–5 nhìn chung ít nhầm lẫn, tuy nhiên vẫn có một số nhầm chéo nhỏ giữa các lớp có nội dung gần nhau (thể hiện qua các ô ngoài đường chéo có giá trị nhỏ).

Tổng kết lại, TF-IDF + Linear SVM cho hiệu năng tổng thể tốt (Macro F1 $\approx$ 0.9504) và ổn định trên tập Test. Tuy nhiên, sai số vẫn tập trung chủ yếu ở một số lớp có nội dung dễ chồng lấn (đặc biệt lớp 6), đồng thời learning curves cho thấy mô hình có xu hướng khớp rất mạnh trên Train

(overfitting nhẹ). Điều này là đặc trưng thường gặp của mô hình tuyến tính với không gian đặc trưng TF-IDF chiều cao.

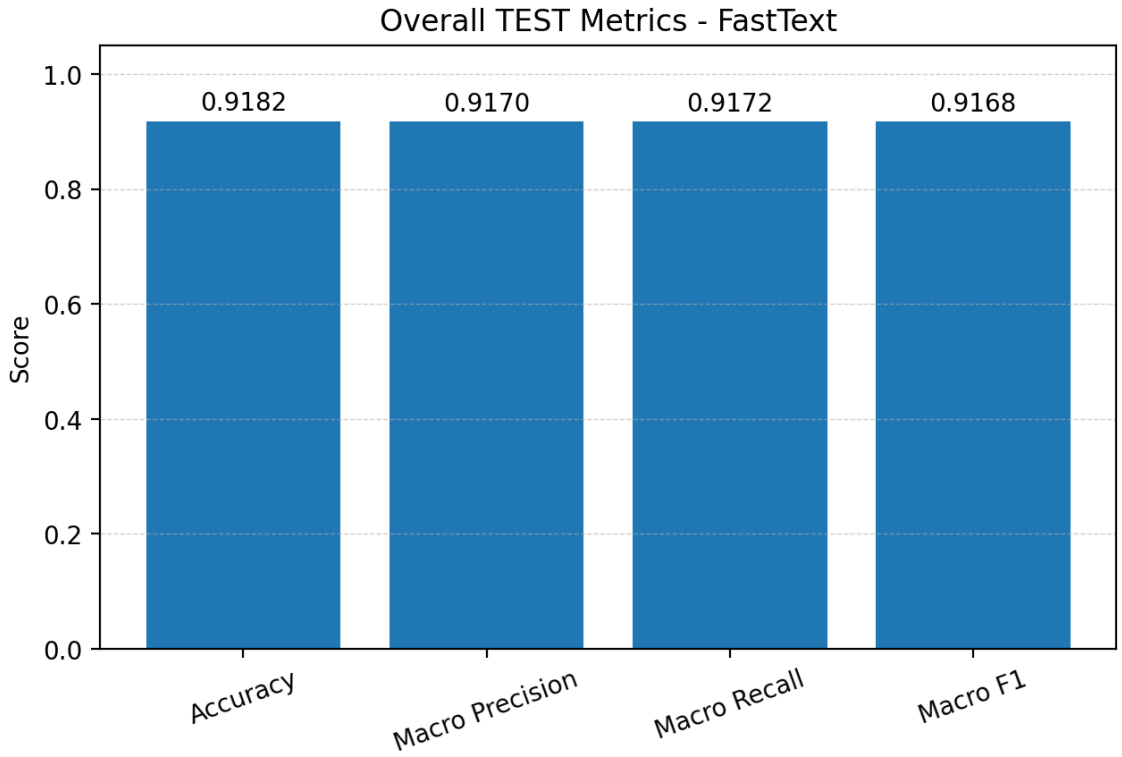
5.2 FastText



Hình 7: Hình learning curves của FastText

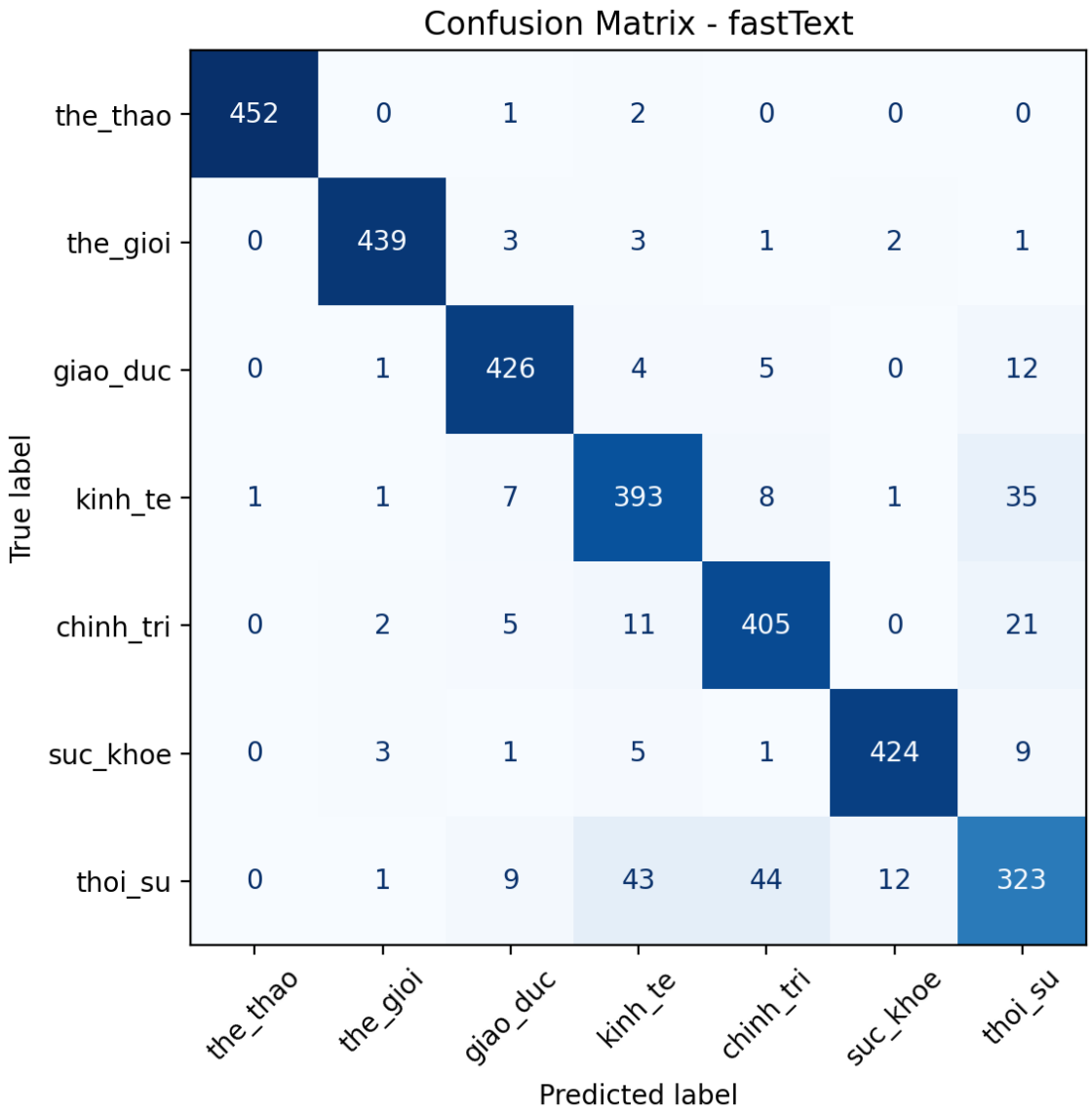
Nhận xét

- LogLoss trên cả Train và Validation giảm đều theo epoch (đặc biệt giảm mạnh từ 5 đến 10 epoch), cho thấy mô hình học được quy luật và có xu hướng hội tụ. Accuracy tăng dần và ổn định theo thời gian, phù hợp với xu hướng giảm của loss.
- Các đường cong thay đổi mượt, không xuất hiện dao động mạnh giữa các mốc epoch. Điều này cho thấy quá trình huấn luyện tương đối ổn định.
- Về khoảng cách Train–Validation: từ sau khoảng 10 - 15 epoch, Train Loss tiếp tục giảm và Train Accuracy tiếp tục tăng, trong khi Validation Accuracy tăng chậm và gần như chững lại ở khoảng 20–25 epoch; đồng thời Validation Loss giảm chậm hơn so với Train Loss. Đây là dấu hiệu mô hình bắt đầu có xu hướng overfitting khi tăng epoch quá cao.



Hình 8: Các chỉ số đánh giá tổng thể trên tập test của FastText

Trên tập Test, mô hình đạt Accuracy = 0.9182, Macro Precision = 0.9170, Macro Recall = 0.9172 và Macro F1 = 0.9168. Các giá trị Precision/Recall/F1 (macro) khá gần nhau, cho thấy hiệu năng giữa các lớp tương đối đồng đều, không bị lệch quá mạnh về một vài lớp.



Hình 9: Confusion matrix của FastText trên tập Test

Nhìn chung, các phần tử trên đường chéo chính chiếm đa số, cho thấy mô hình phân biệt tốt phần lớn các chuyên mục. Một số cặp lớp có mức nhầm lẫn đáng chú ý:

- Lớp thời_sự bị nhầm nhiều sang chính_trị (44 mẫu) và kinh_tế (43 mẫu). Đây là nhóm nhãn có nội dung dễ giao thoa do cùng phản ánh các sự kiện thời sự, chính sách và tình hình kinh tế–xã hội.
- Lớp kinh_tế cũng có xu hướng bị dự đoán sang thời_sự (35 mẫu), phù hợp với việc nhiều bài kinh tế được viết theo dạng tin tức cập nhật.

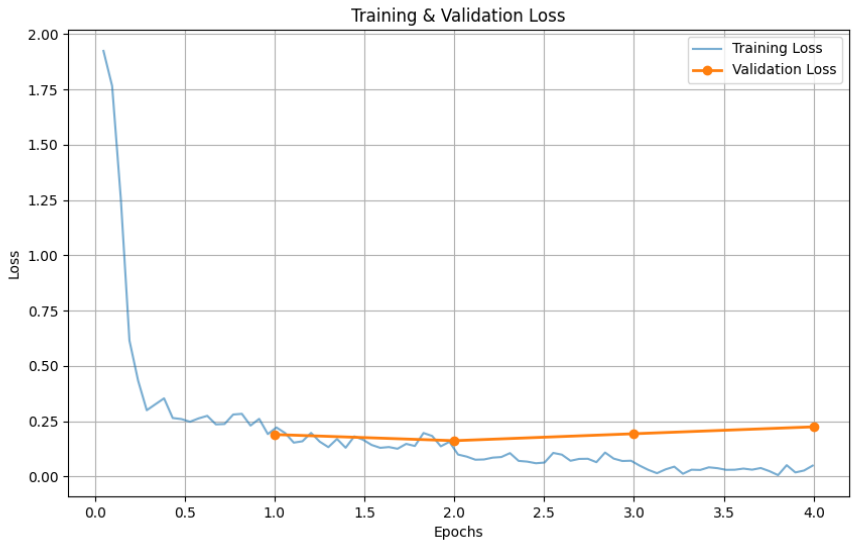
- Lớp chính_trị đôi khi bị nhầm sang thời_sự (21 mẫu) và kinh_tế (11 mẫu), cho thấy ranh giới giữa các nhóm chủ đề này không hoàn toàn tách bạch khi chỉ dựa trên đặc trưng từ/ngữ.
- Các lớp như thể_thao và thế_giới có tỷ lệ nhầm lẫn thấp hơn, nhiều khả năng do có từ khóa đặc trưng và bối cảnh ít chồng lấn hơn.

Nhìn từ confusion matrix, các lỗi chủ yếu tập trung ở nhóm nhãn có nội dung gần nhau, đặc biệt là thời_sự, chính_trị và kinh_tế. Đây cũng là hạn chế của mô hình dựa trên bag-of-words/subwords như FastText, và là lý do các mô hình ngôn ngữ theo ngữ cảnh như PhoBERT được kỳ vọng sẽ cải thiện tốt hơn trong các trường hợp này.

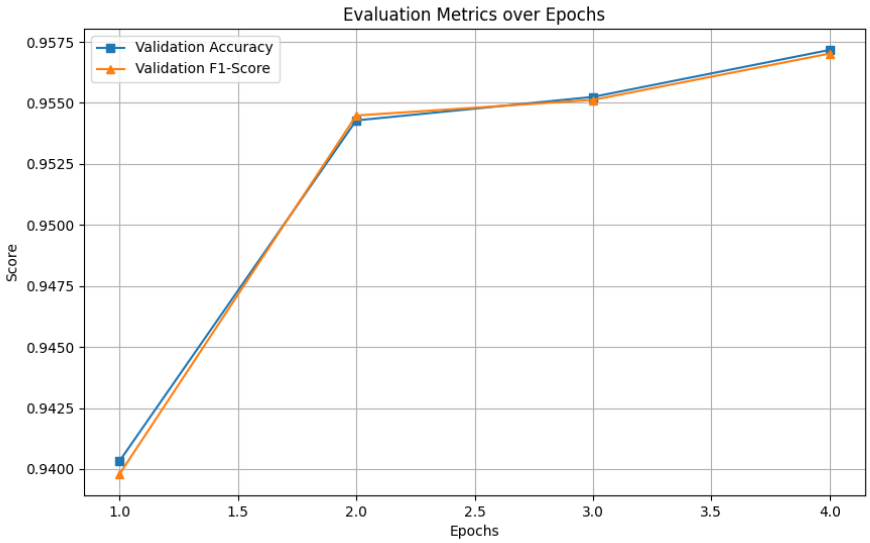
5.3 PhoBERT

5.3.1 Biểu đồ quá trình học

Quá trình huấn luyện được thực hiện trong 4 chu kỳ (epochs) với kích thước lô (batch size) là 16. Diễn biến của hàm mất mát (Loss) và độ chính xác (Accuracy) trên tập huấn luyện (Train) và tập kiểm định (Validation) được thể hiện qua các biểu đồ dưới đây:



Hình 10: Learning Curve cho Loss theo từng epoch



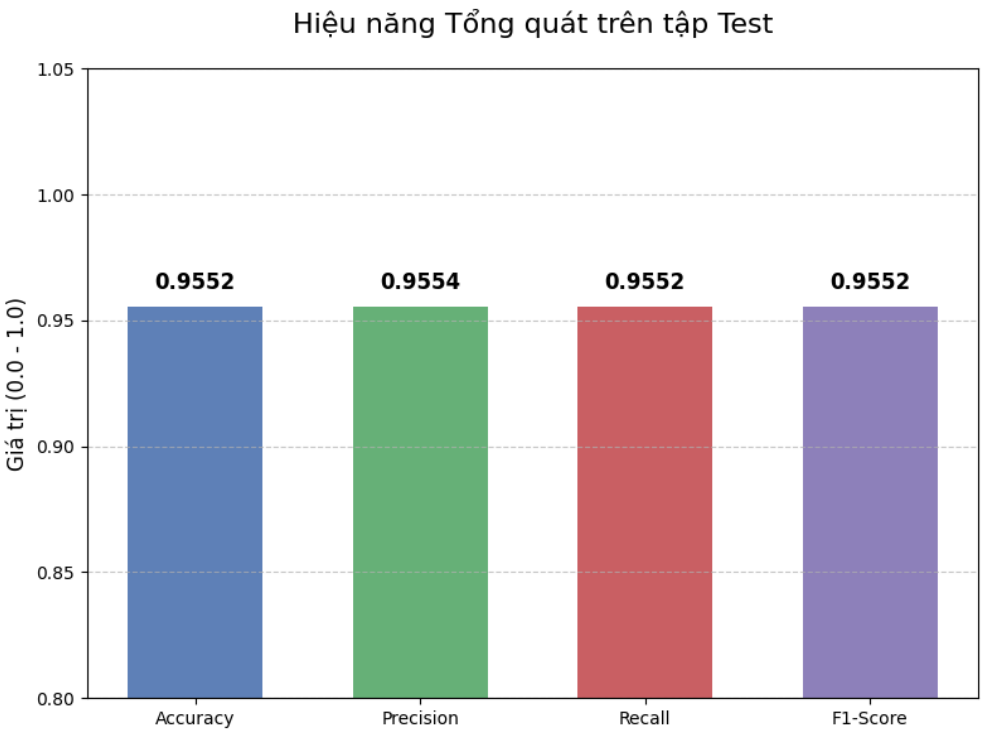
Hình 11: Learning Curve cho Accuracy theo từng epoch

Nhận xét:

- **Hội tụ:** Biểu đồ Loss cho thấy hàm mất mát giảm đều đặn trên cả tập Train và Validation, chứng tỏ mô hình đang học tốt các đặc trưng dữ liệu.
- **Độ ổn định:** Khoảng cách giữa đường Loss của Train và Validation rất nhỏ, đồng thời Accuracy trên tập Validation tăng trưởng đồng biến với tập Train. Điều này cho thấy không xuất hiện hiện tượng dao động mạnh (oscillation) hoặc quá khớp (overfitting) nghiêm trọng trong giai đoạn huấn luyện sớm.

5.3.2 Đánh giá trên tập kiểm thử

Việc đánh giá định lượng được thực hiện trên tập dữ liệu kiểm thử độc lập gồm 2,078 mẫu , đại diện cho 7 chuyên mục tin tức. Kết quả được phân tích dựa trên các chỉ số hiệu năng tổng quát và chi tiết từng lớp. Biểu đồ dưới đây tóm tắt 4 độ đo quan trọng nhất để đánh giá chất lượng mô hình phân loại: Accuracy, Precision, Recall và F1-Score.



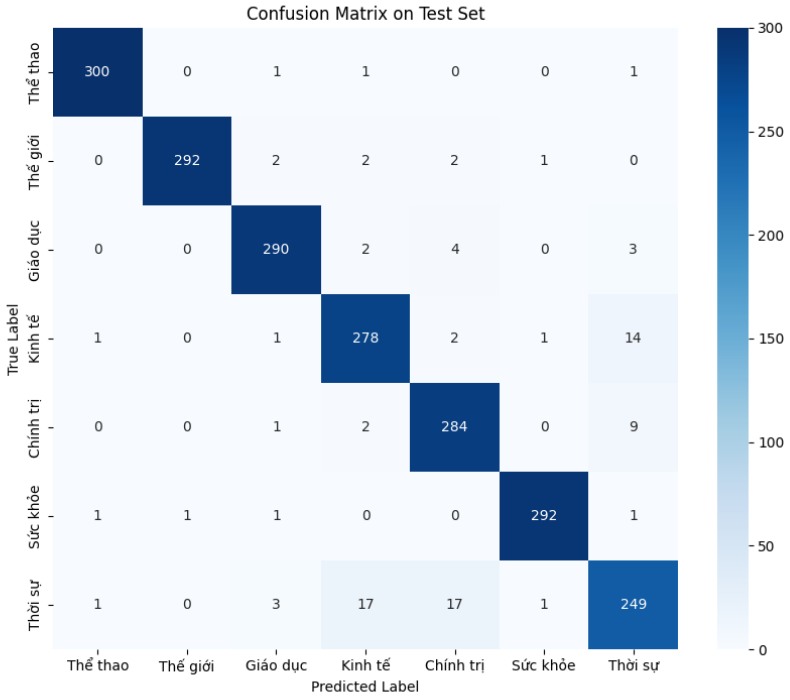
Hình 12: Các chỉ số đánh giá tổng thể trên tập test của PhoBERT

Phân tích kết quả:

- **Độ chính xác (Accuracy) đạt** 95.52%**:** Đây là tỷ lệ dự đoán đúng trên tổng số mẫu, cho thấy mô hình đã học tốt các đặc trưng ngữ nghĩa của văn bản báo chí tiếng Việt.
- **Sự đồng nhất giữa các chỉ số:** Quan sát biểu đồ cột, ta thấy sự chênh lệch giữa Accuracy và F1-Score (95.54%) là không đáng kể (chỉ khoảng 0.02%). Điều này chứng minh mô hình hoạt động ổn định, không bị thiên lệch (bias) về phía các lớp đa số và xử lý tốt sự mất cân bằng nhẹ giữa các lớp dữ liệu.
- **Precision** (95.52%) **và Recall** (95.52%) **cân bằng:** Cho thấy mô hình cân bằng tốt giữa khả năng dự đoán chính xác (không gán nhãn sai) và khả năng bao phủ (không bỏ sót bài báo thuộc nhãn đó).

5.4 Ma trận nhầm lẫn (Confusion Matrix)

Ma trận nhầm lẫn dưới đây minh họa chi tiết sự phân bố các dự đoán của mô hình so với nhãn thực tế:



Hình 13: Ma trận nhầm lẫn cho PhoBERT

Phân tích:

- Đường chéo chính có giá trị rất lớn, thể hiện số lượng mẫu được dự đoán đúng chiếm đa số tuyệt đối.
- Các ô nằm ngoài đường chéo thể hiện sự nhầm lẫn (confusion), tập trung chủ yếu ở cặp nhãn Thời sự - Chính trị và Thời sự - Kinh tế. Điều này phù hợp với bảng chỉ số F1-Score, nơi nhãn "Thời sự" có hiệu năng thấp nhất.

6 Thảo luận và Phân tích lỗi

Dựa trên kết quả thực nghiệm thu được từ ba mô hình (TF-IDF + SVM, FastText, và PhoBERT), phần này tập trung phân tích trạng thái huấn luyện, đi sâu vào các trường hợp dự đoán sai và so sánh hiệu năng để rút ra kết luận cuối cùng.

6.1 Hiện tượng Overfitting và Underfitting

Việc xác định trạng thái của mô hình dựa trên sự tương quan giữa kết quả trên tập Train và Validation/Test:

- **TF-IDF + Linear SVM:** Mô hình cho thấy dấu hiệu *overfitting nhẹ*. Trên tập Train, độ chính xác đạt gần tuyệt đối (100%) và Loss tiệm cận 0, trong khi Accuracy trên tập Validation chỉ dừng lại ở mức $\approx 95\%$. Điều này phản ánh đặc trưng của các mô hình tuyến tính khi làm việc với không gian vector thưa chiều cao (High-dimensional sparse data), dễ dàng ghi nhớ nhiễu trong dữ liệu huấn luyện.
- **FastText:** Xu hướng *overfitting* xuất hiện rõ rệt sau epoch thứ 15. Trong khi Train Loss tiếp tục giảm sâu, Validation Loss lại có xu hướng chững lại và giảm chậm hơn. Điều này cho thấy mô hình bắt đầu học thuộc lòng các n-grams đặc thù của tập Train mà không đóng góp vào khả năng tổng quát hóa.
- **PhoBERT:** Đây là mô hình đạt trạng thái *Good Fit* tốt nhất. Khoảng cách giữa Loss của tập Train và Validation rất nhỏ, đồng thời độ chính xác trên tập Test (95.52%) tương đương với kết quả trên tập Validation. Kiến trúc Transformer cùng cơ chế Regularization có sẵn đã giúp mô hình tổng quát hóa tốt hơn.

6.2 Các biện pháp khắc phục đã thực hiện

Để kiểm soát hiện tượng Overfitting và tối ưu hóa hiệu năng như đã trình bày, nhóm nghiên cứu đã áp dụng các kỹ thuật sau trong quá trình thực nghiệm:

- **Early Stopping:** Đối với FastText và PhoBERT, thay vì lấy mô hình ở epoch cuối cùng (nơi rủi ro overfitting cao nhất), nhóm thực hiện lưu trữ checkpoint dựa trên chỉ số F1-Score tốt nhất trên tập Validation.

- **Tối ưu hóa hàm mục tiêu:** Với SVM, việc sử dụng hàm Hinge Loss kết hợp với thuật toán tối ưu SGD giúp mô hình tìm được siêu phẳng phân cách tối ưu với biên (margin) rộng nhất, giảm thiểu rủi ro quá khớp so với các phương pháp tối ưu cục bộ.
- **Weight Decay:** Áp dụng kỹ thuật suy giảm trọng số (với hệ số 0.01 cho PhoBERT) để phạt các trọng số quá lớn, giữ cho mô hình đơn giản và mượt mà hơn.

6.3 Phân tích các trường hợp sai (Error Analysis)

Quan sát Ma trận nhầm lẫn (Confusion Matrix) của cả ba mô hình, nhóm nhận thấy một quy luật sai số chung: Lỗi tập trung chủ yếu ở nhãn **"Thời sự"** (Lớp 6).

- **Mô tả lỗi:** Nhãn "Thời sự" thường xuyên bị dự đoán nhầm thành "Chính trị" (Lớp 4) và "Kinh tế" (Lớp 3). Ngược lại, một số bài viết "Kinh tế" cũng bị nhầm sang "Thời sự".
- **Giả thuyết nguyên nhân (Hypothesis):** Nguyên nhân chính xuất phát từ sự *nhập nhằng ngữ nghĩa (Semantic Ambiguity)* và tính chất bao hàm của khái niệm "Thời sự".
  - Một bản tin thời sự nóng hổi thường đề cập đến các quyết sách mới (chứa từ vựng Chính trị) hoặc biến động thị trường (chứa từ vựng Kinh tế).
  - *Ví dụ:* Một bài báo có tiêu đề *"Chính phủ họp phiên thường kỳ về giải ngân vốn đầu tư công"* thực chất là tin thời sự trong ngày, nhưng chứa dày đặc các thực thể như "Chính phủ" (đặc trưng Chính trị) và "Vốn đầu tư" (đặc trưng Kinh tế).
- **So sánh hành vi mô hình:**
  - **FastText** dựa trên n-grams nên rất dễ bị đánh lừa bởi các từ khóa chuyên ngành xuất hiện trong văn bản thời sự (tỷ lệ nhầm lẫn cao nhất).
  - **PhoBERT** nhờ cơ chế Self-Attention có khả năng hiểu ngữ cảnh toàn cục tốt hơn, do đó giảm thiểu đáng kể tỷ lệ nhầm lẫn này so với hai mô hình còn lại, tuy nhiên vẫn chưa thể triệt tiêu hoàn toàn do sự giao thoa nội dung quá lớn.

6.4 So sánh hiệu năng các mô hình

Bảng dưới đây tóm tắt và so sánh hiệu năng của ba hướng tiếp cận trên tập kiểm thử (Test Set):

Bảng 4: Bảng so sánh hiệu năng giữa các mô hình trên tập Test

Mô hình	**Accuracy**	**Macro Precision**	**Macro Recall**	**Macro F1**
TF-IDF + SVM	95.14%	95.13%	95.07%	95.04%
FastText	91.82%	91.70%	91.72%	91.68%
PhoBERT	**95.52%**	**95.54%**	**95.52%**	**95.52%**

Kết luận:

1. **PhoBERT** cho kết quả vượt trội nhất trên mọi chỉ số. Điều này khẳng định sức mạnh của mô hình ngôn ngữ tiền huấn luyện và kiến trúc Transformer trong việc xử lý các đặc điểm ngôn ngữ phức tạp của tiếng Việt.

2. **TF-IDF + SVM** là một baseline cực kỳ mạnh mẽ, đạt hiệu năng xấp xỉ PhoBERT (chỉ kém $\sim 0.4\%$) nhưng với chi phí tính toán thấp hơn rất nhiều. Điều này cho thấy với dữ liệu văn bản báo chí được tiền xử lý tốt, các phương pháp truyền thống vẫn rất hiệu quả.

3. **FastText** tuy có tốc độ huấn luyện nhanh nhất nhưng hiệu năng thấp hơn đáng kể ($\sim 91.8\%$). Hạn chế của việc chỉ dựa vào trung bình hóa vector từ (word averaging) khiến mô hình khó nắm bắt được cấu trúc câu phức tạp so với PhoBERT.

7 Tài liệu tham khảo

[1] Viet Trung Tran. *PyVi: Python Vietnamese CoreNLP*. https://github.com/trungtv/pyvi. 2016.

[2] Thanh Vu et al. "VnCoreNLP: A Vietnamese Natural Language Processing Toolkit". In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*. 2018, pp. 56–60.

[3] Armand Joulin et al. *Bag of Tricks for Efficient Text Classification*. 2016. arXiv: 1607.01759 [cs.CL]. URL: https://arxiv.org/abs/1607.01759.

[4] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[5] Adam Paszke et al. "PyTorch: An Imperative Style, High-Performance Deep Learning Library". In: *Advances in Neural Information Processing Systems 32*. 2019, pp. 8024–8035.

[6] Thomas Wolf et al. "Transformers: State-of-the-Art Natural Language Processing". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Association for Computational Linguistics, Oct. 2020, pp. 38–45.

[7] Dat Quoc Nguyen and Anh Tuan Nguyen. "PhoBERT: Pre-trained language models for Vietnamese". In: *Findings of the Association for Computational Linguistics: EMNLP 2020*. 2020, pp. 1037–1042.