

ỨNG DỤNG HỌC MÁY VÀ TẦN SUẤT THUẬT NGỮ - NGHỊCH ĐẢO TẦN SUẤT TÀI LIỆU TRONG PHÂN LOẠI HỒ SƠ ỨNG VIÊN NGÀNH CÔNG NGHỆ THÔNG TIN HỖ TRỢ TIỀN SÀNG LỌC TUYỂN DỤNG

Nguyễn Thị Mười Phương^{1*}, Trần Thị Lua¹, Trần Thị Nga¹

Lý Cao Tú¹, Vũ Thị Duyên¹

*Tác giả liên hệ, Email: phuong.ntm@phuongdong.edu.vn; ORCID: 0009-0009-0157-9281

Ngày tòa soạn nhận được bài báo: 15/06/2026

Ngày phản biện đánh giá: 22/06/2026

Ngày bài báo được duyệt đăng: 14/09/2026

DOI: 10.59266/houjs.2026.1425

Tóm tắt: Nghiên cứu xây dựng hệ thống hỗ trợ tiền sàng lọc hồ sơ ứng viên ngành Công nghệ thông tin thông qua phân loại hồ sơ ứng viên (Curriculum Vitae - CV) tiếng Việt theo nhóm nghề. Hệ thống tích hợp mô-đun trích xuất văn bản từ các định dạng đầu vào, tiền xử lý văn bản, biểu diễn đặc trưng bằng TF-IDF và bốn thuật toán học máy gồm Máy véc-tơ hỗ trợ (Support Vector Machine - SVM), bộ phân loại Bayes đơn giản (Naive Bayes - NB), hồi quy logistic (Logistic Regression - LR) và rừng ngẫu nhiên (Random Forest - RF). Bộ dữ liệu nghiên cứu gồm 345 CV được gán vào bốn nhóm Web Developer, Backend Developer, Data Analyst và Tester. Dữ liệu được phân chia có phân tầng thành tập huấn luyện và tập kiểm tra; việc lựa chọn mô hình và tối ưu siêu tham số được thực hiện trên tập huấn luyện, trong khi tập kiểm tra độc lập gồm 69 CV được sử dụng cho đánh giá cuối cùng. Trên tập kiểm tra, SVM tuyến tính đạt độ chính xác (Accuracy) 88,41%, điểm F1 có trọng số (Weighted-F1) 88,22%, điểm F1 trung bình vĩ mô (Macro-F1) 87,10% và độ chính xác cân bằng (Balanced Accuracy) 87,04%. Phân tích lỗi cho thấy sự nhầm lẫn tập trung chủ yếu giữa Web Developer và Backend Developer. Kết quả cho thấy TF-IDF kết hợp SVM có tiềm năng hỗ trợ phân loại CV ở bước tiền sàng lọc; tuy nhiên hệ thống không thay thế đánh giá chuyên môn của nhà tuyển dụng và cần được kiểm định trên dữ liệu độc lập lớn hơn trước khi triển khai thực tế.

Từ khóa: học máy, hồ sơ ứng viên, phân loại văn bản, tiền sàng lọc, tần suất thuật ngữ - nghịch đảo tần suất tài liệu

I. Đặt vấn đề

Sự phát triển của các nền tảng tuyển dụng trực tuyến làm gia tăng đáng kể số lượng hồ sơ mà doanh nghiệp cần tiếp

nhận và xử lý. Đối với lĩnh vực Công nghệ thông tin (CNTT), một CV thường chứa đồng thời thông tin về ngôn ngữ lập trình, framework, cơ sở dữ liệu, công cụ, dự án,

¹Trường Đại học Phương Đông, Hà Nội, Việt Nam

vị trí công việc và kinh nghiệm thực tế. Việc phân loại thủ công một số lượng lớn hồ sơ vì vậy đòi hỏi nhiều thời gian và có thể chịu ảnh hưởng bởi sự khác biệt trong cách đánh giá của người tuyển dụng.

Các kỹ thuật xử lý ngôn ngữ tự nhiên và học máy cho phép biểu diễn nội dung CV dưới dạng đặc trưng số và xây dựng mô hình nhận biết các nhóm hồ sơ. Tuy nhiên, phân loại CV theo nhóm nghề không đồng nghĩa với đánh giá năng lực của ứng viên. Đánh giá năng lực đòi hỏi mô hình năng lực, tiêu chí và thang đo riêng cũng như quá trình kiểm định phù hợp. Vì vậy, nghiên cứu này giới hạn phạm vi ở việc xây dựng hệ thống hỗ trợ tiền sàng lọc bằng cách phân loại CV vào nhóm nghề dự kiến, không đưa ra quyết định tuyển dụng hay điểm năng lực cho ứng viên.

Nghiên cứu tập trung trả lời ba câu hỏi: (1) TF-IDF (Term Frequency - Inverse Document Frequency) kết hợp các thuật toán học máy truyền thống có thể phân loại CV tiếng Việt ngành CNTT với hiệu năng như thế nào; (2) mô hình nào cho kết quả phù hợp nhất trên bộ dữ liệu nghiên cứu; và (3) các nhóm nghề nào có mức độ nhầm lẫn cao và cần lưu ý khi sử dụng hệ thống trong tiền sàng lọc.

Đóng góp của nghiên cứu gồm: xây dựng một bộ dữ liệu CV tiếng Việt theo bốn nhóm nghề CNTT; xây dựng quy trình xử lý từ trích xuất văn bản đến phân loại; và thực nghiệm so sánh các mô hình học máy trên cùng biểu diễn TF-IDF.

II. Cơ sở lý thuyết và tổng quan nghiên cứu

2.1. Phân loại văn bản và biểu diễn TF-IDF

Phân loại văn bản là bài toán gán một hoặc nhiều nhãn đã xác định trước cho tài liệu dựa trên nội dung (Aggarwal

và Zhai, 2012). Trong CV ngành CNTT, các thuật ngữ như Java, Python, React, Node.js, SQL, Selenium hay Power BI có khả năng cung cấp tín hiệu phân biệt giữa các nhóm nghề.

TF-IDF biểu diễn tài liệu dưới dạng vector và điều chỉnh trọng số của từ dựa trên tần suất xuất hiện trong một tài liệu và mức độ phổ biến của từ đó trong toàn bộ tập văn bản. Với cách triển khai `TfidfVectorizer` có `smooth_idf=True`, trọng số IDF được tính:

$$\text{idf}(t) = \log[(1+N)/(1+\text{df}(t))] + 1$$

trong đó N là tổng số tài liệu và $\text{df}(t)$ là số tài liệu chứa từ t ; việc cộng 1 vào tử và mẫu tương ứng với cơ chế làm mượt `smooth_idf=True`.

2.2. Các mô hình phân loại

Nghiên cứu đối sánh bốn mô hình: LR, SVM, NB và RF. SVM tuyến tính đặc biệt phù hợp với dữ liệu văn bản TF-IDF do dữ liệu có số chiều lớn và thưa (Joachims, 1998). NB được sử dụng làm mô hình cơ sở nhờ cấu trúc đơn giản và chi phí tính toán thấp. LR cho phép thiết lập ranh giới tuyến tính và có khả năng diễn giải hệ số. RF được đưa vào nhằm so sánh với một phương pháp học tập hợp dựa trên cây (Breiman, 2001).

2.3. Nghiên cứu liên quan và khoảng trống nghiên cứu

Các nghiên cứu về xử lý hồ sơ tuyển dụng có thể chia thành ba hướng chính: trích xuất thông tin từ CV, phân loại CV theo nhóm nghề và ghép nối hồ sơ ứng viên với yêu cầu công việc. Những nghiên cứu gần đây cũng chuyển dần từ phương pháp dựa trên đặc trưng truyền thống sang mô hình ngôn ngữ tiền huấn luyện và mô hình ngôn ngữ lớn.

Gan và Mori (2022) xây dựng corpus hồ sơ và xem việc nhận diện các thành phần của hồ sơ như một bài toán phân loại văn bản, đồng thời đánh giá các mô hình ngôn ngữ tiền huấn luyện. Heakl và cộng sự (2024) chỉ ra rằng quy mô và chất lượng bộ dữ liệu là yếu tố quan trọng đối với phân loại hồ sơ, đồng thời nhấn mạnh các khó khăn liên quan đến sự không đồng nhất của CV và quyền riêng tư. Trong một hướng khác, Wang và cộng sự (2022) nghiên cứu sự phù hợp người - việc dựa trên hồ sơ ứng viên, thông tin tuyển dụng và lịch sử tuyển dụng; Barrak và cộng sự (2022) nhấn mạnh yêu cầu về khả năng giải thích, truy vết và công bằng trong hệ thống gợi ý CV- vị trí việc làm.

Đối với tiếng Việt, các kỹ thuật TF-IDF và SVM đã được sử dụng trong nhiều bài toán phân loại văn bản (Nguyễn và Phạm, 2018). Bộ công cụ VnCoreNLP của Vu và cộng sự (2018) hỗ trợ các bước xử lý tiếng Việt như tách từ và gán nhãn ngôn ngữ.

Khoảng trống mà nghiên cứu hướng tới không nằm ở việc tạo ra phương pháp học máy mới mà ở việc xây dựng và đánh giá một quy trình phân loại CV tiếng Việt trong bối cảnh một số nhóm nghề CNTT có từ vựng chuyên môn chồng lấn.

III. Phương pháp nghiên cứu

3.1. Bộ dữ liệu và gán nhãn

Bộ dữ liệu ban đầu gồm 345 CV ngành Công nghệ thông tin, được thu thập từ tháng 4 đến tháng 6 năm 2025 từ các nguồn công khai trên môi trường web, thu thập thủ công và một số doanh nghiệp quy mô nhỏ cung cấp cho mục đích nghiên cứu. Trước khi sử dụng, các thông

tin định danh trực tiếp như họ tên, số điện thoại, email và địa chỉ được loại bỏ. Dữ liệu được phân thành bốn nhóm nghề: Tester 95 CV (27,54%), Web Developer 65 CV (18,84%), Backend Developer 94 CV (27,25%) và Data Analyst 91 CV (26,38%). Dữ liệu sử dụng cho mô hình gồm các thông tin chuyên môn như kỹ năng kỹ thuật, công cụ và framework, ngôn ngữ lập trình, kinh nghiệm, dự án, chứng chỉ, kỹ năng mềm và ngoại ngữ; không sử dụng các trường định danh trực tiếp như họ tên, số điện thoại, email và địa chỉ.

CV được lựa chọn khi có đủ thông tin chuyên môn để xác định một trong bốn nhóm nghề. Nhãn được xác định dựa trên vị trí nghề nghiệp, kinh nghiệm và nhóm kỹ năng chiếm ưu thế. Với CV phù hợp nhiều nhóm, ưu tiên vị trí công việc chính hoặc gần nhất; nếu không xác định rõ, nhãn được xác định theo nội dung chuyên môn chiếm ưu thế. Các trường hợp không thuộc phạm vi bốn nhóm nghề được loại khỏi tập thực nghiệm.

Việc gán nhãn được thực hiện bởi hai người dựa trên cùng bộ tiêu chí về vị trí nghề nghiệp, kinh nghiệm và nhóm kỹ năng chiếm ưu thế. Đối với các trường hợp có cách xác định nhãn khác nhau, hai người cùng rà soát và thống nhất nhãn cuối cùng theo quy tắc gán nhãn đã xác định.

Trước khi huấn luyện, dữ liệu được rà soát tính nhất quán của nhãn và kiểm tra trùng lặp trên nội dung văn bản đã chuẩn hóa. Các bản ghi trùng hoàn toàn chỉ giữ một bản; các trường hợp gần trùng được rà soát trước khi loại bỏ. Bộ dữ liệu sau làm sạch được khóa trước khi phân chia và đánh giá mô hình.

3.2. Trích xuất và tiền xử lý văn bản

Hệ thống tiếp nhận văn bản nhập trực tiếp và các tệp PDF, Word hoặc hình ảnh. Với tệp chứa văn bản có thể đọc trực tiếp, nội dung được trích xuất từ cấu trúc tài liệu. OCR (Optical Character Recognition - nhận dạng ký tự quang học) chỉ được sử dụng đối với tệp ảnh hoặc PDF dạng ảnh.

Quy trình tiền xử lý gồm chuẩn hóa văn bản, chuyển chữ về dạng thống nhất, xử lý ký tự không có giá trị phân loại, tách từ và loại bỏ một số từ dừng. VnCoreNLP được sử dụng hỗ trợ xử lý tiếng Việt; bộ công cụ xử lý ngôn ngữ tự nhiên (Natural Language Toolkit - NLTK) được sử dụng cho một số thành phần tiếng Anh.

3.3. Vector hóa dữ liệu bằng TF-IDF

Dữ liệu sau khi làm sạch được biểu diễn dưới dạng véc-tơ số theo phương pháp TF-IDF bằng công cụ TfidfVectorizer thuộc thư viện scikit-learn (Pedregosa và cộng sự, 2011). Kỹ thuật TF-IDF được áp dụng nhằm định lượng tầm quan trọng của từng từ trong ngữ cảnh toàn bộ tập dữ liệu theo công thức:

$$TF - IDF(t,d) = TF(t,d) * IDF(t)$$

Văn bản sau tiền xử lý được biểu diễn bằng TF-IDF sử dụng TfidfVectorizer. Các tham số thực nghiệm gồm max_features=5000, min_df=2, max_df=0,95, ngram_range=(1,2) và sublinear_tf=True; các tham số lowercase=True, norm='l2', use_idf=True và smooth_idf=True sử dụng giá trị mặc định của scikit-learn. Cấu hình này sử dụng unigram và bigram, loại các đặc trưng xuất hiện trong dưới 2 tài liệu hoặc trên 95% số tài liệu và giới hạn tối đa 5.000 đặc trưng.

Quy trình chuyển đổi một CV mới thành vector đặc trưng TF-IDF được mô tả chi tiết thuật toán 1 dưới đây.

Thuật toán 1: Kỹ thuật mã hóa văn bản TF-IDF

Input:

D: Tập dữ liệu văn bản huấn luyện gồm N CV thuộc tập huấn luyện.

d_{new} : Văn bản cần được chuyển đổi - CV cần phân loại

Output: Vector đặc trưng TF-IDF đã chuẩn hóa v_{new} của văn bản d_{new}

Begin

// Bước 1: Xây dựng từ điển từ toàn bộ tập dữ liệu huấn luyện

$V \leftarrow$ Tập hợp tất cả các từ duy nhất được trích xuất từ tập D

// Bước 2: Tính trọng số IDF cho từng từ trong từ điển

for each từ $t \in V$ do

df_t $\leftarrow$ số lượng văn bản trong D có chứa từ t

IDF(t) $\leftarrow \log[(1 + N)/(1 + df_t)] + 1$ // Làm mượt Laplace, tránh chia cho 0

end for

// Bước 3: Tính vector TF-IDF cho văn bản mới d_{new}

$v_{new} \leftarrow$ vector khởi tạo kích thước $|V|$, tất cả giá trị bằng 0

for each từ $t \in d_{new}$ do

if $t \in V$ then

TF(t, d_{new}) $\leftarrow 1 + \log(f(t, d_{new}))$, nếu $f(t, d_{new}) > 0$

$v_{new}[t] \leftarrow TF(t, d_{new}) \times IDF(t)$

end if

end for

```

// Bước 4: Chuẩn hóa L2 để loại bỏ ảnh hưởng của độ dài văn bản
norm ← sqrt(  $\sum v\_new[i]^2$  ) //  $i = 1 \rightarrow |V|$ 
if norm > 0 then
    v_new ← v_new / norm
end if

return v_new
End

```

Thuật toán 1 mô tả cách mã hóa một CV thành vector TF-IDF, trong khi Thuật toán 2 mở rộng quá trình này cho toàn bộ dữ liệu theo chiến lược chỉ khớp mô hình trên tập huấn luyện và sử dụng mô hình đã khớp để biến đổi cả tập huấn luyện và tập kiểm tra. TfidfVectorizer chỉ học từ vựng và IDF trên tập huấn luyện, sau đó sử dụng cùng bộ tham chiếu để chuyển đổi tập kiểm tra, qua đó bảo đảm không gian đặc trưng thống nhất và tránh rò rỉ dữ liệu. Đầu ra là hai ma trận thưa và dùng cho huấn luyện và đánh giá mô hình.

Thuật toán 2: Vector hóa văn bản TF-IDF cho tập huấn luyện và kiểm tra

Input:

X_train: Tập dữ liệu CV dùng để huấn luyện

X_test: Tập dữ liệu CV dùng để kiểm tra

params: Bộ tham số cấu hình cho TfidfVectorizer bao gồm:

max_features = 5000: giới hạn tối đa 5.000 đặc trưng

ngram_range = (1,2): sử dụng unigram và bigram

lowercase = True: chuyển văn bản về chữ thường khi vector hóa

norm = 'l2': chuẩn hóa L2 trên môi trường vector đặc trưng

Output:

(*X_train_vec*, *X_test_vec*): Cặp ma trận thưa đặc trưng TF-IDF tương ứng với tập huấn luyện và tập kiểm tra.

Begin

```

// Bước 1: Khởi tạo bộ vector hóa TF-IDF với các tham số tùy chỉnh
vectorizer ← TfidfVectorizer(params)
// Bước 2: Học từ vựng và tham số IDF chỉ từ tập huấn luyện
// (fit trên X_train để tránh data leakage từ X_test)
fit(vectorizer, X_train)
// Bước 3: Chuyển đổi cả hai tập sang không gian đặc trưng TF-IDF
// dùng cùng bộ từ vựng và tham số IDF đã học ở Bước 2
X_train_vec ← transform(vectorizer, X_train)
X_test_vec ← transform(vectorizer, X_test)
// Bước 4: Kiểm tra tính nhất quán chiều không gian đặc trưng
num_features_train ← count_columns(X_train_vec)
num_features_test ← count_columns(X_test_vec)
if num_features_train = num_features_test then
    // Hai tập có cùng số chiều đặc trưng → dữ liệu hợp lệ
    return (X_train_vec, X_test_vec)

```

```

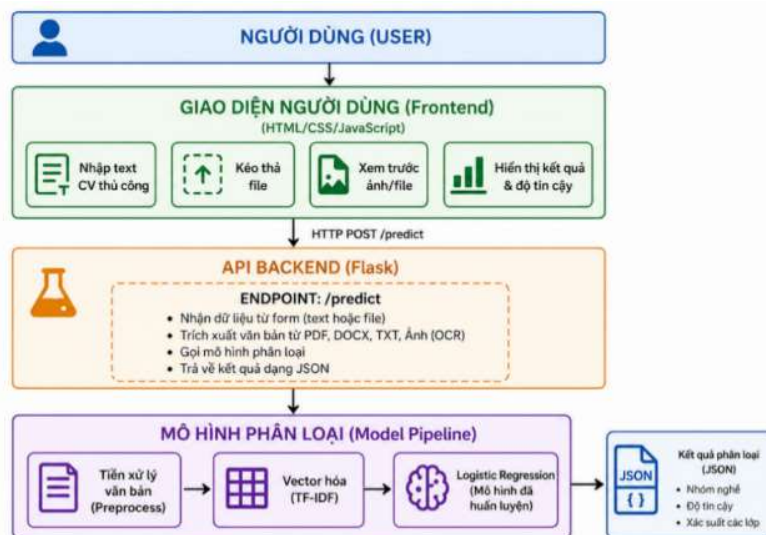
else
    // Không đồng nhất số chiều → báo lỗi, dừng quy trình
    raise Error("Không đồng nhất số chiều giữa tập huấn luyện và tập kiểm tra")
end if
End

```

3.4. Xây dựng hệ thống hỗ trợ phân loại CV

Bên cạnh việc vector hóa dữ liệu, nghiên cứu xây dựng một luồng xử lý toàn diện nhằm tự động tiếp nhận và phân loại CV. Hệ thống được thiết kế theo kiến trúc ba tầng tách biệt: Giao diện người

dùng (Frontend), Giao diện lập trình ứng dụng phía máy chủ (Backend API) và Mô hình phân loại (Model Pipeline), đảm bảo tính module hóa và khả năng mở rộng. Kiến trúc tổng thể và luồng xử lý dữ liệu được minh họa lần lượt tại hình 1 và hình 2 dưới đây.



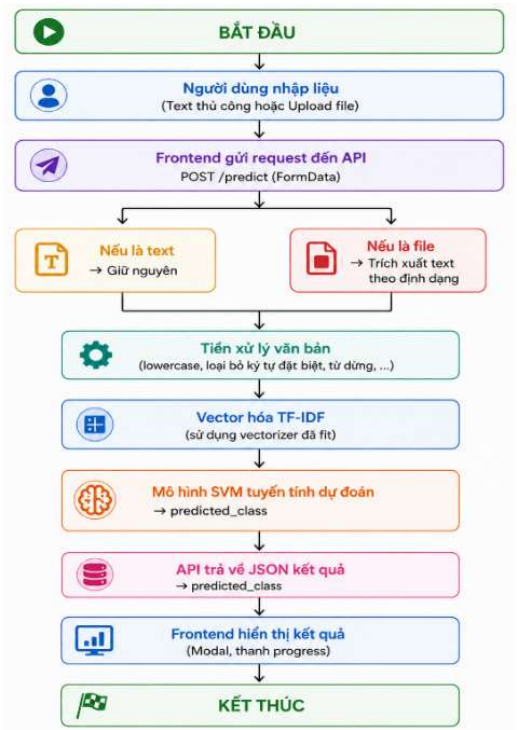
Hình 1. Sơ đồ kiến trúc tổng thể của hệ thống phân loại CV

Tầng 1- Giao diện người dùng: cho phép người dùng nhập văn bản CV hoặc tải lên tệp, xem trước nội dung và nhận kết quả phân loại. Dữ liệu đầu vào được gửi đến hệ thống phía máy chủ thông qua phương thức POST của giao thức truyền siêu văn bản (Hypertext Transfer Protocol - HTTP).

Tầng 2 - Backend API: tiếp nhận dữ liệu từ Frontend và thực hiện trích xuất văn bản theo định dạng tệp đầu vào. Đầu ra là chuỗi văn bản thuần túy, được chuyển vào quy trình phân loại và kết quả được trả về dưới định dạng trao đổi dữ liệu JSON (JavaScript Object Notation).

Tầng 3 - Mô hình phân loại: thực hiện tiền xử lý văn bản, vector hóa TF-IDF và dự đoán nhóm nghề bằng mô hình đã huấn luyện. Đầu ra là nhãn nhóm nghề dự đoán để hỗ trợ người sử dụng xem xét hồ sơ.

Như minh họa tại hình 2, luồng xử lý vận hành tuyến tính từ bước nhập dữ liệu, trích xuất, tiền xử lý, vector hóa, dự đoán đến hiển thị kết quả, đảm bảo tính nhất quán và minh bạch trong toàn bộ quá trình. Kiến trúc giao diện lập trình ứng dụng tạo cơ sở kỹ thuật cho việc thử nghiệm tích hợp với các hệ thống tuyển dụng trong các nghiên cứu tiếp theo.



Hình 2. Sơ đồ luồng xử lý và phân loại CV từ dữ liệu đầu vào

3.5. Thiết lập quy trình huấn luyện và đánh giá

Bộ dữ liệu gồm 345 CV được phân chia theo tỷ lệ 80/20, trong đó 276 mẫu dùng cho tập huấn luyện và 69 mẫu dùng cho tập kiểm tra. Việc tối ưu siêu tham số được thực hiện bằng phương pháp tìm kiếm theo lưới (Grid Search) trên tập huấn luyện.

Bảng 1. Kết quả phân loại của mô hình Linear SVM trên tập kiểm tra

Nhóm nghề	Độ chính xác (Precision)	Độ bao phủ (Recall)	Điểm F1	Số mẫu
Lập trình viên phía máy chủ (Backend Developer)	0,7895	0,7895	0,7895	19
Chuyên viên phân tích dữ liệu (Data Analyst)	1,0000	1,0000	1,0000	18
Chuyên viên Kiểm thử (Tester)	0,9500	1,0000	0,9744	19
Lập trình Web (Web Developer)	0,7500	0,6923	0,7200	13
Trung bình vĩ mô (Macro avg)	0,8724	0,8704	0,8710	69
Trung bình có trọng số (Weighted avg)	0,8812	0,8841	0,8822	69

Độ chính xác tổng thể (Accuracy) = 0,8841 (88,41%)

Điểm F1 trung bình vĩ mô (Macro-F1) = 0,8710 (87,10%)

Tập kiểm tra gồm 69 CV được khóa và chỉ sử dụng một lần cho đánh giá cuối cùng.

Môi trường thực nghiệm sử dụng Python 3.10.0, scikit-learn 1.3.0, NLTK 3.9.2 và thư viện NumPy 1.24.3; mô-đun nhận dạng ký tự quang học (OCR) sử dụng pytesseract 0.3.13 và Tesseract 5.5.0.20241111. Dữ liệu được chia theo tỷ lệ 80:20 với random_state=42. SVM tuyến tính được tối ưu bằng Grid Search 5-fold trên tập huấn luyện với $C \in \{0,01;0,05;0,1;0,5;1;2;5;10;20;50\}$, $loss \in \{squared_hinge, hinge\}$ và $max_iter \in \{2000, 3000, 5000\}$; cấu hình cuối cùng sử dụng $C=1,0$.

Hiệu năng của mô hình được đánh giá chủ yếu bằng điểm F1 trung bình vĩ mô (Macro-F1) và độ chính xác cân bằng (Balanced Accuracy) nhằm phản ánh đồng đều hiệu quả phân loại giữa các lớp; độ chính xác (Accuracy) và điểm F1 có trọng số (Weighted-F1) được báo cáo như các chỉ số bổ sung.

IV. Kết quả và thảo luận

4.1. Kết quả trên tập kiểm tra độc lập

Trên tập kiểm tra độc lập gồm 69 CV, SVM tuyến tính đạt kết quả như trình bày tại Bảng 1.

Điểm F1 có trọng số (Weighted-F1) = 0,8822 (88,22%)

Độ chính xác cân bằng (Balanced Accuracy) = 0,8704 (87,04%)

Trên tập kiểm tra độc lập gồm 69 CV, Linear SVM đạt Accuracy 88,41%, Macro-F1 87,10%, Balanced Accuracy 87,04% và Weighted-F1 88,22%. Kết quả theo từng lớp cho thấy Data Analyst đạt F1-score 1,0000 và Tester đạt 0,9744, trong khi Backend Developer đạt 0,7895 và Web Developer thấp nhất với 0,7200. Đặc biệt, recall của Web Developer chỉ đạt 0,6923, cho thấy đây là nhóm khó nhận diện nhất trong tập kiểm tra.

Phân tích ma trận nhầm lẫn cho thấy sai số chủ yếu tập trung giữa Web Developer và Backend Developer: 4 CV Web được dự đoán thành Backend, 3 CV Backend được dự đoán thành Web và 1 CV

Backend được dự đoán thành Tester. Mặc dù độ bao phủ của Tester đạt 1,0000, độ chính xác chỉ đạt 0,9500 do có một trường hợp Backend bị dự đoán thành Tester; vì vậy không nên diễn giải kết quả của lớp Tester là “chính xác tuyệt đối”.

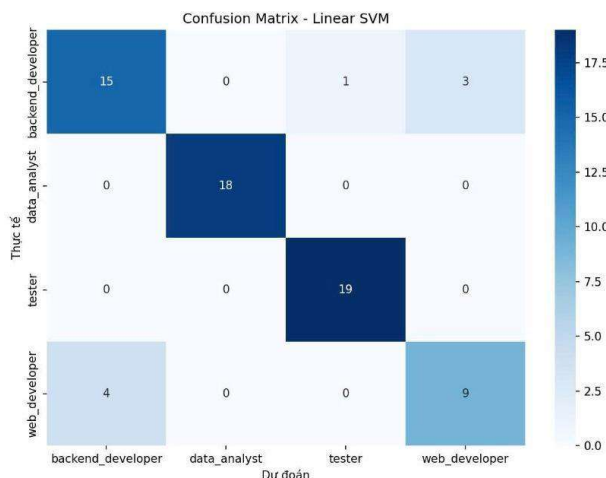
4.2. Kết quả kiểm tra chéo

Để đánh giá khả năng khái quát và hạn chế sự phụ thuộc vào một lần chia dữ liệu, nghiên cứu sử dụng Stratified 10-Fold Cross-Validation. Macro-F1 và Balanced Accuracy được sử dụng làm độ đo chính; Weighted-F1 và Accuracy được báo cáo bổ sung. Kết quả được trình bày dưới dạng trung bình $\pm$ độ lệch chuẩn qua 10 fold.

Bảng 2. Kết quả Stratified 10-Fold Cross-Validation của các mô hình

Mô hình	Điểm F1 trung bình vĩ mô (Macro-F1)	Độ chính xác cân bằng (Balanced Accuracy)	Điểm F1 có trọng số (Weighted-F1)	Độ chính xác tổng thể (Accuracy)	Khoảng tin cậy 95% của Macro-F1 (95% CI Macro-F1)
Máy véc-tơ hỗ trợ (SVM)	0,8433 $\pm 0,0580$	0,8448 $\pm 0,0542$	0,8560 $\pm 0,0538$	0,8603 $\pm 0,0511$	[0,7996; 0,8871]
Bộ phân loại Bayes đơn giản (Naive Bayes)	0,7096 $\pm 0,0865$	0,7356 $\pm 0,0675$	0,7410 $\pm 0,0864$	0,7787 $\pm 0,0742$	[0,6443; 0,7748]
Hồi quy logistic (Logistic Regression)	0,7936 $\pm 0,0898$	0,8002 $\pm 0,0774$	0,8120 $\pm 0,0828$	0,8253 $\pm 0,0738$	[0,7259; 0,8613]
Rừng ngẫu nhiên (Random Forest)	0,8148 $\pm 0,0954$	0,8167 $\pm 0,0850$	0,8303 $\pm 0,0864$	0,8369 $\pm 0,0805$	[0,7429; 0,8868]

Nguồn: Kết quả thực nghiệm của nhóm tác giả



Hình 3. Ma trận nhầm lẫn của mô hình SVM trên tập kiểm tra

4.3. So sánh các mô hình

Kết quả kiểm tra chéo cho thấy SVM đạt Macro-F1 và Balanced Accuracy trung bình cao nhất trong bốn mô hình, đồng thời có độ lệch chuẩn Macro-F1 thấp nhất. Random Forest đứng thứ hai, tiếp theo là Logistic Regression và Naive Bayes. Tuy nhiên, khoảng tin cậy Macro-F1 của một số mô hình còn chồng lấn; do đó nghiên cứu chỉ kết luận SVM đạt hiệu năng trung bình tốt nhất trong thực nghiệm, chưa khẳng định sự khác biệt có ý nghĩa thống kê.

4.4. Hạn chế, khả năng tổng quát hóa và vấn đề đạo đức

Nghiên cứu có một số hạn chế. Thứ nhất, quy mô 345 CV còn nhỏ và chỉ bao phủ bốn nhóm nghề CNTT. Kết quả vì vậy chưa thể khái quát cho toàn bộ thị trường tuyển dụng hoặc các nhóm nghề khác.

Thứ hai, Web Developer và Backend Developer có sự chồng lấn đáng kể về kỹ năng và thuật ngữ, làm tăng nhầm lẫn giữa hai lớp. Trong thực tế, một ứng viên có thể đồng thời phù hợp với nhiều vị trí; mô hình phân loại đơn nhân vì vậy chỉ phản ánh một cách đơn giản hóa bài toán tuyển dụng. Nghiên cứu chưa định lượng mức độ đồng thuận giữa hai người gán nhãn, đây là một hạn chế cần được xem xét khi mở rộng bộ dữ liệu.

Thứ ba, nghiên cứu chưa đánh giá định lượng sai số OCR và ảnh hưởng của sai số này đến kết quả phân loại; do đó chất lượng OCR cần được kiểm định độc lập trong nghiên cứu tiếp theo.

Thứ tư, CV chứa dữ liệu cá nhân và có thể chứa các thuộc tính liên quan đến giới tính, tuổi, trường đào tạo hoặc các đặc điểm khác không cần thiết cho việc phân loại nghề nghiệp. Việc xử lý dữ liệu phải tuân thủ nguyên tắc tối thiểu hóa dữ liệu, ẩn danh và giới hạn quyền truy cập. Bộ dữ

liệu chỉ nên được công bố hoặc chia sẻ ở dạng đã loại bỏ thông tin nhận dạng và khi có cơ sở pháp lý/đồng thuận phù hợp.

Thứ năm, mô hình học máy có thể kế thừa thiên lệch từ dữ liệu lịch sử. Vì vậy đầu ra chỉ nên được xem là gợi ý hỗ trợ tiền sàng lọc. Nhà tuyển dụng phải giữ vai trò quyết định cuối cùng và cần có khả năng xem xét, sửa hoặc bác bỏ dự đoán của hệ thống.

V. Kết luận và kiến nghị

Nghiên cứu đã xây dựng một hệ thống thử nghiệm phân loại CV tiếng Việt ngành CNTT kết hợp trích xuất văn bản, TF-IDF và các mô hình học máy. Trên tập kiểm tra 69 CV, SVM tuyến tính đạt Accuracy 88,41%, macro-F1 khoảng 87,10%, weighted-F1 khoảng 88,22% và balanced accuracy 87,04%. Phân tích ma trận nhầm lẫn cho thấy tám trường hợp phân loại sai, tập trung chủ yếu giữa Web Developer và Backend Developer.

Kết quả cho thấy TF-IDF kết hợp mô hình tuyến tính là một baseline có khả năng hỗ trợ phân nhóm CV trong điều kiện bộ dữ liệu quy mô nhỏ. Tuy nhiên, kết quả nghiên cứu chỉ phản ánh khả năng phân loại CV theo nhóm nghề, không phải đánh giá năng lực hoặc quyết định tuyển dụng.

Các nghiên cứu tiếp theo cần mở rộng dữ liệu về quy mô, nguồn và nhóm nghề; đánh giá độc lập chất lượng OCR; thử nghiệm mô hình ngôn ngữ tiếng Việt; nghiên cứu bài toán đa nhãn và ghép nối ứng viên - vị trí; đánh giá khả năng giải thích và công bằng; đồng thời thực hiện external validation và thử nghiệm vận hành có sự giám sát của con người trước khi xem xét tích hợp vào hệ thống HRM hoặc ATS.

Tài liệu tham khảo

- Aggarwal, C. C., & Zhai, C. (2012). *Mining text data*. Springer. <https://doi.org/10.1007/978-1-4614-3223-4>
- Barrak, A., Adams, B., & Zouaq, A. (2022). Toward a traceable, explainable, and fair JD/resume recommendation system. *arXiv*. <https://doi.org/10.48550/arXiv.2202.08960>
- Breiman, L. (2001). Random forests. *Machine Learning*, 45(1), 5-32. <https://doi.org/10.1023/A:1010933404324>
- Gan, C., & Mori, T. (2022). Construction of English resume corpus and test with pre-trained language models. *arXiv*. <https://doi.org/10.48550/arXiv.2208.03219>
- Heakl, A., Mohamed, Y., Mohamed, N., Elsharkawy, A., & Zaky, A. (2024). ResumeAtlas: Revisiting resume classification with large-scale datasets and large language models. *arXiv*. <https://doi.org/10.48550/arXiv.2406.18125>
- Joachims, T. (1998). Text categorization with support vector machines: Learning with many relevant features. In *Proceedings of the European Conference on Machine Learning* (pp. 137-142). Springer. <https://doi.org/10.1007/BFb0026683>
- Nguyen, L. M. và Pham. Q. N. (2018). Phân loại tài liệu tiếng Việt sử dụng TF-IDF và SVM. *Tạp chí Khoa học và Công nghệ Đại học Đà Nẵng*, (11), 45-50.
- Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12, 2825-2830. <https://jmlr.org/papers/v12/pedregosa11a.html>
- Vu, T., Nguyen, D. Q., Nguyen, D. Q., Dras, M., & Johnson, M. (2018). VnCoreNLP: A Vietnamese natural language processing toolkit. In *Proceedings of NAACL: Demonstrations* (pp. 56-60). Association for Computational Linguistics. <https://doi.org/10.18653/v1/N18-5012>
- Wang, Z., Wei, W., Xu, C., Xu, J., & Mao, X.-L. (2022). Person-job fit estimation from candidate profile and related recruitment history with co-attention neural networks. *arXiv*. <https://doi.org/10.48550/arXiv.2206.09116>

APPLICATION OF MACHINE LEARNING AND TERM FREQUENCY-INVERSE DOCUMENT FREQUENCY (TF-IDF) TO INFORMATION TECHNOLOGY CANDIDATE CV CLASSIFICATION FOR RECRUITMENT PRE-SCREENING

Nguyen Thi Muoi Phuong¹*, Tran Thi Lua¹, Tran Thi Nga¹
Ly Cao Tu¹, Vu Thi Duyen¹

Abstract: *This study develops a system for supporting the pre-screening of information technology candidates by automatically classifying Vietnamese resumes into job categories. The system integrates text extraction from multiple input formats, text preprocessing, TF-IDF feature representation, and four machine-learning algorithms: Support Vector Machine, Naive Bayes, Logistic Regression, and Random Forest. The dataset consists of 345 resumes labeled into four categories: Web Developer, Backend Developer, Data Analyst, and Tester. The data are stratified into training and test sets; model selection and hyperparameter tuning are performed only on the training data, while an independent set of 69 resumes is reserved for final evaluation. On the test set, the linear SVM achieves an accuracy of 88.41%, a weighted F1-score of 88.22%, a macro-F1 of 87.10%, and a balanced accuracy of 87.04%. Errors are concentrated mainly between Web Developer and Backend Developer. The results indicate that TF-IDF combined with linear SVM can support resume classification at the recruitment pre-screening stage. Nevertheless, the system should not replace professional human judgment and requires external validation on larger and more diverse datasets before operational deployment.*

Keywords: *machine learning, pre-screening, resume classification, text classification, TF-IDF*

¹ University of Phuong Dong, Hanoi, Vietnam