Tuần 4 - Hệ thống phân loại chủ đề bài báo

Time-Series Team

Ngày 11 tháng 10 năm 2025

Dự án Hệ thống phân loại bài báo gồm các nội dung chính:

- ***Đặt vấn đề***
- ***Mở rộng của nhóm***
- ***Giới thiệu chi tiết mở rộng của nhóm***

Mục lục

1 Đặt vấn đề

Trong kỷ nguyên bùng nổ tri thức, số lượng *publication abstract* tăng nhanh trên các kho học thuật (ví dụ: arXiv). Điều này đặt ra thách thức lớn cho việc **tìm kiếm, lập chỉ mục và khai thác tri thức**. Bài toán trọng tâm là: *với một abstract đầu vào, dự đoán chính xác topic phù hợp (ví dụ: Vật lý, Toán học, Khoa học máy tính, …)*. Bên cạnh yêu cầu về độ chính xác, hệ thống hiện đại cần khả năng **giải thích dự đoán**, giúp người dùng hiểu được tại sao mô hình lại đưa ra kết quả đó.

Mục tiêu

- Xây dựng một chương trình phân loại topic cho abstract bằng Python, sử dụng các thư viện học máy phổ biến như `scikit-learn`, `numpy`, …
- So sánh hiệu quả của các phương pháp mã hoá văn bản: **Bag-of-Words (BoW)**, **TF–IDF**, và **Sentence Embeddings**.
- Huấn luyện và đánh giá nhiều mô hình phân loại khác nhau, bao gồm **Naive Bayes**, **KNN**, và **Decision Tree**.

2 Mở rộng của nhóm

Ngoài các yêu cầu cơ bản, nhóm tiến hành một số mở rộng để tăng tính thực tiễn và giá trị học thuật của dự án. Các mở rộng được lựa chọn nhằm giải quyết **những hạn chế quan sát được trong thực nghiệm cơ bản**, đồng thời bổ sung khả năng **giải thích kết quả dự đoán**. Cụ thể:

(i) Thử nghiệm phân loại theo *primary scores*

Ở bước đầu, nhóm tiến hành kiểm chứng giả thuyết rằng việc tận dụng trực tiếp các *primary scores* có thể phân loại abstract nhanh chóng. Tuy nhiên, kết quả thực nghiệm cho thấy phương pháp này **không mang lại độ chính xác cao** do thiếu tính khái quát và khó nắm bắt đặc trưng ngữ nghĩa. Điều này khẳng định sự cần thiết phải quay về khung phân loại với các nhóm chủ đề chính (`astro-ph`, `cond-mat`, `cs`, `math`, `physics`) và sử dụng các phương pháp mã hoá văn bản hiện đại hơn.

(ii) Đề xuất KNN bỏ phiếu trọng số cải tiến

KNN truyền thống thường gặp hạn chế khi dữ liệu mất cân bằng hoặc khi mức độ tương đồng giữa các điểm lân cận không phản ánh đúng tầm quan trọng thực tế. Để khắc phục, nhóm đề xuất công thức bỏ phiếu trọng số mới:

$$\text{weight} \;=\; (1-\alpha)\cdot \text{similarity}\cdot \text{class\_weights} \;+\; \alpha \cdot \text{saliency}.$$

Trong đó:

- similarity: mức độ tương đồng (cosine similarity) giữa điểm kiểm tra và láng giềng.
- class_weights: trọng số lớp, giảm thiểu ảnh hưởng của mất cân bằng dữ liệu.
- saliency: độ quan trọng của đặc trưng, giúp mô hình chú ý hơn đến các tín hiệu ngôn ngữ có giá trị.

Công thức này giúp KNN **ổn định hơn, chính xác hơn**, đồng thời tăng khả năng thích ứng với dữ liệu đa dạng.

(iii) Giải thích từ vựng quan trọng góp phần phân loại chủ để của bài báo khoa học bằng *saliency scores*

Để mô hình không chỉ "dự đoán đúng" mà còn "giải thích được", nhóm khai thác *saliency scores* để nhận diện những từ/cụm từ có đóng góp lớn vào quyết định phân loại. Kết quả này cho phép:

- Loại bỏ *stopwords* và làm nổi bật các thuật ngữ đặc trưng của từng lĩnh vực (ví dụ: `graph`, `algorithm` cho `cs`; `galaxy`, `spectra` cho `astro-ph`).
- Cung cấp công cụ giải thích trực quan, giúp người dùng hoặc nhà nghiên cứu hiểu rõ *"tại sao abstract này thuộc về topic X"*.

Ý nghĩa của các mở rộng

Việc thực hiện ba mở rộng trên nhằm đạt được các mục tiêu sau:

1. **Khắc phục hạn chế** của phương pháp cơ bản (primary scores không hiệu quả, KNN truyền thống thiếu ổn định).
2. **Tăng tính thực tiễn** bằng cách điều chỉnh trọng số để giải quyết dữ liệu mất cân bằng.
3. **Nâng cao tính minh bạch** thông qua phân tích *saliency*, phù hợp với xu hướng *Explainable AI*.

Đóng góp chính

- Xây dựng khung phân loại chủ đề *chu trình từ phân loại đến giải thích từ khóa quan trọng* kết hợp nhiều phương pháp mã hoá và mô hình.
- Đề xuất biến thể **KNN bỏ phiếu trọng số** kết hợp đồng thời *similarity*, *class weights*, và *saliency*, nhằm cải thiện hiệu năng trên dữ liệu mất cân bằng.
- Phát triển mô-đun **giải thích dự đoán** ở mức từ/ngữ, giúp hệ thống minh bạch hơn và hỗ trợ kiểm thử dữ liệu.

3 Giới thiệu chi tiết Mở rộng trong thuật toán của nhóm

3.1 Giải thích cách tính *Saliency Score*

Nhận thấy rằng toàn bộ hệ thống phân loại sử dụng mô hình embedding **E5** kết hợp với cơ sở dữ liệu **FAISS** để truy vấn và tìm kiếm `k` láng giềng gần nhất là một mô hình dạng "hộp đen" (black-box). Vì vậy, nhóm đặt mục tiêu **tăng tính giải thích của mô hình** bằng cách chỉ ra cụ thể những *token* nào trong câu đầu vào thực sự ảnh hưởng đến embedding của câu, từ đó dẫn đến quyết định phân loại. Ý tưởng chính là trực quan hoá mức độ đóng góp của từng token bằng *bản đồ nhiệt* (heatmap) — token nào càng có ảnh hưởng lớn thì sẽ được tô màu đậm hơn.

Trong bối cảnh ban đầu nhóm tập trung vào bài toán phân loại và giải thích các tin nhắn **spam**, quá trình tính toán saliency heatmap được triển khai cho các token góp phần vào embedding của câu được phân loại là *spam*. Nếu có thêm thời gian, phương pháp này hoàn toàn có thể mở rộng để giải thích cho cả các câu được phân loại là *ham* với cùng cơ chế.

Ý tưởng thuật toán.

- **Bước 1.** Tính `class_scores` ban đầu: tổng điểm tương đồng giữa embedding của câu đầu vào với các láng giềng có nhãn trùng với lớp mục tiêu (ví dụ: "spam").
- **Bước 2.** Với mỗi token tại vị trí i, thay token đó bằng `[PAD]` (hoặc `[MASK]`) để mô hình embedding xem như vị trí này bị che.
- **Bước 3.** Sinh embedding mới từ câu bị che token, thực hiện truy vấn FAISS, và tính lại tổng điểm `class_scores` cho lớp mục tiêu.
- **Bước 4.** Độ giảm saliency của token i được xác định bằng hiệu số:

$$\text{Saliency}(t_i) \;=\; \texttt{class\_scores}_{\text{original}} \;-\; \texttt{class\_scores}_{\text{masked}(i)}.$$

  Nếu giá trị này càng lớn thì token t_i càng quan trọng đối với quyết định phân loại.
- **Bước 5.** Lặp lại cho tất cả token trong câu để thu được một danh sách giá trị saliency tương ứng với từng token.
- **Bước 6.** Chuẩn hoá các giá trị saliency về khoảng $[0, 1]$, sau đó đưa vào hàm `render_heatmap` để trực quan hoá ảnh hưởng của từng token bằng màu sắc.

Ý nghĩa. Phương pháp này giúp hệ thống không chỉ *dự đoán đúng* mà còn *giải thích được*, nhờ đó người dùng hoặc nhà nghiên cứu có thể hiểu rõ "vì sao" một abstract hay tin nhắn được gán vào một chủ đề cụ thể.

3.2 Đề xuất KNN bỏ phiếu trọng số cải tiến

Hạn chế của KNN truyền thống. Thuật toán KNN (K-Nearest Neighbors) trong các biến thể *majority voting* và *weighted voting* thường gặp các vấn đề sau:

- **Mất cân bằng dữ liệu:** các lớp có số mẫu ít thường bị lấn át bởi các lớp lớn.
- **Độ tương đồng không đủ mạnh:** sử dụng khoảng cách hoặc cosine similarity thuần túy chưa phản ánh đúng mức độ quan trọng của các đặc trưng.

Công thức cải tiến. Để khắc phục, nhóm đề xuất công thức bỏ phiếu trọng số mới:

$$\text{weight} \;=\; (1-\alpha)\cdot \text{similarity}\cdot \text{class\_weights} \;+\; \alpha \cdot \text{saliency}.$$

Trong đó:

- similarity: mức độ tương đồng (cosine similarity) giữa điểm kiểm tra và láng giềng.
- class_weights: trọng số lớp (nghịch tần suất), giúp giảm thiểu tác động của dữ liệu mất cân bằng.
- saliency: độ quan trọng của đặc trưng, được tính toán từ biến thiên embedding và khả năng đóng góp vào quyết định phân loại.
- $\alpha \in [0, 1]$: hệ số điều chỉnh mức độ ảnh hưởng giữa *similarity* và *saliency*.

Ý nghĩa. Công thức này mang lại ba lợi ích chính:

1. Tăng tính **ổn định** của KNN khi gặp dữ liệu đa dạng và mất cân bằng.
2. Cho phép mô hình **chú ý nhiều hơn** đến các đặc trưng quan trọng thay vì chỉ dựa trên khoảng cách.
3. Linh hoạt trong việc điều chỉnh bằng tham số α, giúp cân bằng giữa độ tương đồng và mức độ quan trọng ngữ nghĩa.

Kết quả thực nghiệm. Nhóm đã tiến hành so sánh giữa KNN truyền thống và KNN cải tiến trên ba phương pháp mã hoá văn bản: Bag-of-Words, TF–IDF và Embeddings. Bảng dưới đây tóm tắt độ chính xác (*accuracy*) thu được:

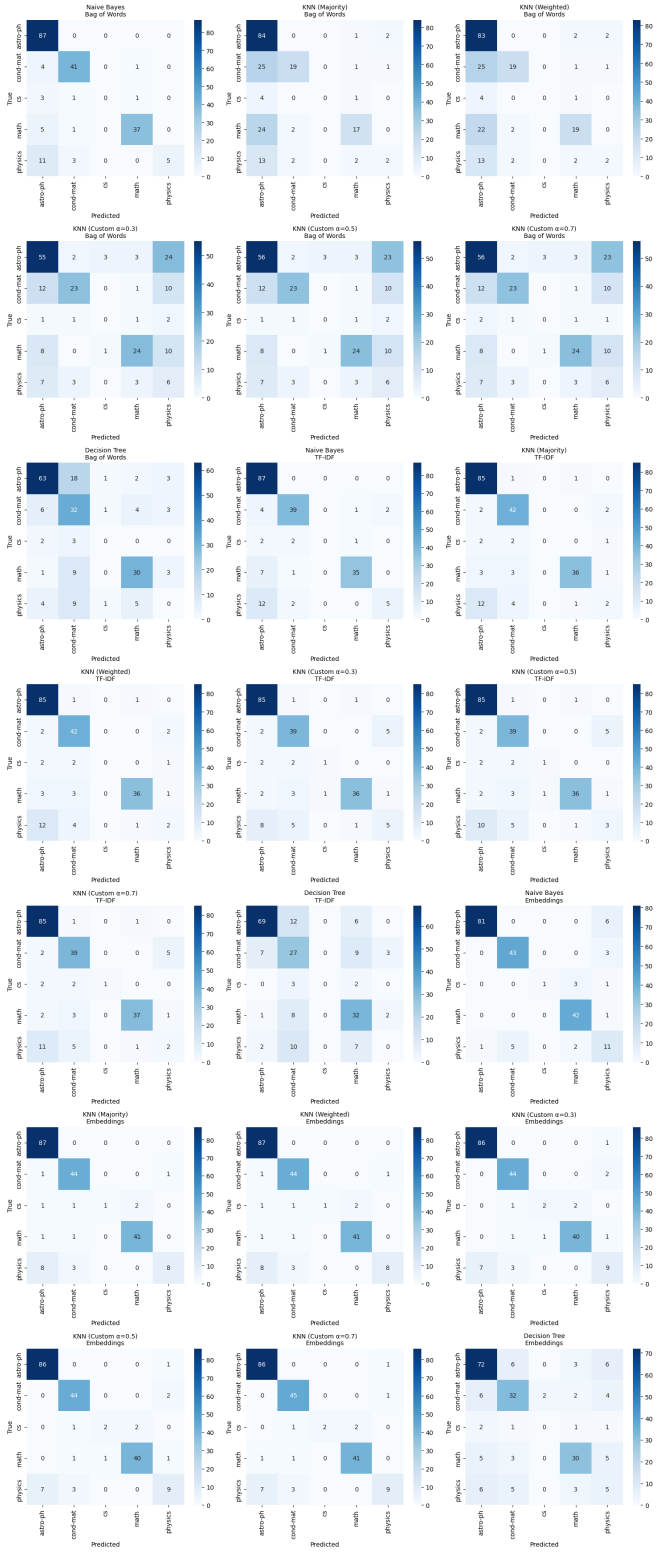
Mô hình	**Bag-of-Words**	**TF–IDF**	**Embeddings**
Naive Bayes	0.8500	0.8300	0.8900
KNN (Majority)	0.6100	0.8250	0.9050
KNN (Weighted)	0.6150	0.8250	0.9050
KNN (Custom $\alpha = 0.3$)	0.5400	0.8300	0.9050
KNN (Custom $\alpha = 0.5$)	0.5450	0.8200	0.9050
KNN (Custom $\alpha = 0.7$)	0.5450	0.8200	**0.9150**
Decision Tree	0.6250	0.6400	0.6950

Phân tích.

- Với **Bag-of-Words**, KNN cải tiến chưa thể hiện ưu thế rõ rệt do biểu diễn thưa thớt và ít thông tin ngữ nghĩa.
- Với **TF–IDF**, KNN cải tiến ($\alpha = 0.3$) đạt kết quả ngang bằng hoặc nhỉnh hơn so với KNN truyền thống.
- Với **Embeddings**, KNN cải tiến với $\alpha = 0.7$ đạt độ chính xác cao nhất **91.5%**, vượt trội so với Naive Bayes (89.0%) và Decision Tree (69.5%).

Phân tích qua Confusion Matrix. Để đánh giá chi tiết hơn ngoài chỉ số *accuracy*, nhóm trình bày các *confusion matrix* cho từng mô hình và phương pháp mã hoá. Kết quả cho thấy:

- Với **Bag-of-Words**, cả KNN truyền thống lẫn KNN cải tiến đều gặp nhiều lỗi nhầm lẫn giữa *math* và *physics*, cho thấy biểu diễn BoW còn hạn chế trong việc tách biệt các khái niệm.
- Với **TF–IDF**, mô hình cải tiến (KNN Custom, $\alpha = 0.3$) có sự cân bằng tốt hơn giữa các lớp, giảm nhầm lẫn so với Weighted KNN.
- Với **Embeddings**, KNN Custom $\alpha = 0.7$ thể hiện vượt trội: các lớp *astro-ph*, *cond-mat*, *cs*, *math*, *physics* đều đạt phân tách rõ rệt, sai số giảm đáng kể so với các baseline.



Hình 1: Confusion Matrix của tất cả mô hình với 3 phương pháp mã hoá (BoW, TF–IDF, Embeddings). Trường hợp tốt nhất: KNN Custom ($\alpha = 0.7$) + Embeddings đạt accuracy 91.5%.

Kết luận. Biến thể KNN bỏ phiếu trọng số tích hợp *similarity*, *class weights* và *saliency* cho thấy hiệu quả vượt trội khi kết hợp với phương pháp mã hoá **Sentence Embeddings**, trở thành lựa chọn

tốt nhất cho bài toán phân loại chủ đề abstract khoa học.

3.3 Giải thích từ vựng quan trọng bằng *saliency scores*

Một hạn chế lớn của các mô hình phân loại văn bản hiện nay là tính **hộp đen** (*black-box*): mô hình có thể đưa ra dự đoán chính xác nhưng khó giải thích được cơ chế ra quyết định. Để khắc phục, nhóm áp dụng **saliency scores** nhằm nhận diện các token (từ/cụm từ) có đóng góp lớn nhất vào quá trình phân loại. Cách tiếp cận này giúp mô hình không chỉ "*dự đoán đúng*" mà còn "*giải thích được*".

Ý tưởng chính. Mỗi từ trong `abstract` sẽ được đánh giá về mức độ ảnh hưởng của nó tới quyết định phân loại. Thuật toán được triển khai dựa trên cơ chế **leave-one-out masking**: lần lượt che một token khỏi câu đầu vào, tính toán lại embedding và phân phối xác suất lớp, so sánh với dự đoán ban đầu. Độ giảm điểm số của lớp mục tiêu chính là **saliency** của token đó. Các bước chi tiết:

1. Với câu đầu vào, tính embedding gốc và xác định lớp dự đoán cùng độ tin cậy.
2. Với mỗi token t_i, thay thế bằng `[PAD]` hoặc `[MASK]`, rồi tính embedding và phân phối lớp mới.
3. Tính $\Delta = \texttt{score}_{\text{gốc}} - \texttt{score}_{\text{masked}}$. Nếu Δ lớn, token t_i có vai trò quan trọng.
4. Chuẩn hoá toàn bộ giá trị về khoảng $[0, 1]$ để trực quan hoá bằng bản đồ nhiệt (heatmap).

Kết quả thực nghiệm. Khi áp dụng lên tập dữ liệu `arXiv abstracts`, kết quả thu được các từ vựng đặc trưng nổi bật cho từng lĩnh vực (Hình **??**). Ví dụ:

- **Computer Science (cs)**: "algorithm", "polynomial", "time", "analysis", "framework".
- **Astrophysics (astro-ph)**: "galaxy", "stars", "spectra", "clusters".
- **Mathematics (math)**: "paper", "group", "theorem", "function".
- **Condensed Matter (cond-mat)**: "magnetic", "field", "spin", "temperature".
- **Physics (physics)**: "model", "quantum", "theory", "field".

Những từ/cụm từ này vừa mang tính chuyên ngành cao vừa khẳng định khả năng **loại bỏ stopwords**, chỉ giữ lại tín hiệu ngôn ngữ có giá trị thực sự cho phân loại.

Ý nghĩa. Việc tích hợp **saliency scores** mang lại hai lợi ích chính:

1. **Tăng tính minh bạch**: người dùng có thể thấy vì sao abstract được gán vào một topic cụ thể.
2. **Hỗ trợ nghiên cứu**: các nhà khoa học có thể khai thác trực tiếp danh sách "từ đặc trưng" để phân tích nội dung, đánh giá chất lượng dữ liệu, hoặc phát hiện nhãn sai.

Lợi ích khi kiểm thử abstract mới. Một điểm mạnh của cơ chế **saliency scores** là không chỉ dừng lại ở việc huấn luyện và đánh giá trên tập dữ liệu, mà còn có thể áp dụng trực tiếp cho *abstract mới* nhằm kiểm chứng khả năng giải thích của mô hình. Ví dụ, khi nhập vào một abstract về vật liệu hai chiều, mô hình dự đoán chủ đề **cond-mat** (vật lý chất ngưng tụ) với độ tin cậy 100%.
Kết quả phân tích saliency chỉ ra rằng các token `magnetic`, `using`, `quantum` đóng vai trò then chốt, được đánh dấu nổi bật trong *heatmap* (Hình **??**). Điều này hoàn toàn hợp lý về mặt ngữ nghĩa: các khái niệm "từ trường", "cơ học lượng tử" và "sử dụng phương pháp tính toán" là những đặc trưng chuyên biệt của lĩnh vực condensed matter physics.

Ý nghĩa thực tiễn:

- Người dùng hoặc nhà nghiên cứu có thể hiểu rõ *vì sao* abstract được gán vào lớp **cond-mat**, thay vì chỉ nhìn thấy nhãn dự đoán.
- Cơ chế này đóng vai trò như một "hộp đen có cửa sổ", giúp kiểm thử độ tin cậy, phát hiện dữ liệu nhiễu hoặc sai nhãn (mislabeling).
- Tạo công cụ **hỗ trợ giảng dạy và phân tích**, nơi người dùng có thể xem ngay những từ khoá đặc trưng của từng lĩnh vực học thuật.

4 Giới thiệu chi tiết Mở rộng User-Interface của nhóm bằng Streamlit

4.1 Mục tiêu và giá trị sử dụng

Để giúp giảng viên và người dùng tương tác trực quan với toàn bộ pipeline, nhóm phát triển một bảng điều khiển (*dashboard*) bằng `Streamlit` cho phép:

- **Quan sát hiệu năng** của mọi cặp {mô hình, bộ mã hoá} (BoW/TF–IDF/Embeddings) kèm biểu đồ tương tác.
- **So sánh chi tiết** giữa các biến thể (Naive Bayes, KNN–Majority, KNN–Weighted, KNN–Custom $\alpha \in \{0.3, 0.5, 0.7\}$, Decision Tree) theo độ chính xác, Precision/Recall/F1 macro/weighted.
- **Phân tích lỗi** qua *confusion matrix* tương tác cho từng cấu hình đã chọn.
- **Thử nghiệm suy luận** trên một *abstract* mới và xem ngay *vì sao* được gán nhãn (*explainability* qua saliency/"top words").

4.2 Kiến trúc màn hình và luồng thao tác

Ứng dụng tổ chức thành bốn khối chức năng, người dùng chọn trong `Sidebar`:

1) **chart-increasing Model Performance Overview**: tổng quan mô hình tốt nhất, độ chính xác cao nhất, bộ mã hoá tương ứng, và biểu đồ cột (`plotly`) so sánh hiệu năng.
2) **bullseye Model Comparison**: so sánh nhiều mô hình trên *cùng một* bộ mã hoá, hiển thị biểu đồ *accuracy* và bảng chỉ số chi tiết (Macro/Weighted F1, Precision, Recall).
3) **mag Detailed Analysis**: báo cáo phân loại theo từng lớp (*classification report*) và *confusion matrix* dạng `seaborn heatmap` cho cặp {mô hình, mã hoá} được chọn.
4) **memo Text Classification Demo**: nhập *abstract* hoặc chọn *sample*, hệ thống suy luận bằng **mô hình tốt nhất** tìm được (*best result*) và trả về nhãn dự đoán, mô tả từng chủ đề, văn bản đã tiền xử lý; có thể gắn kèm heatmap saliency ở phần mở rộng.

4.3 Tích hợp mô hình và dữ liệu

Tải và tiền xử lý. Ứng dụng tải tập `UniverseTBD/arxiv-abstracts-large`, lọc 5 chủ đề mục tiêu {`astro-ph, cond-mat, cs, math, physics`} và lấy ≈ 1000 mẫu *single-label*. Chuỗi được chuẩn hoá (xoá ký tự đặc biệt/số, `lowercase`, nén khoảng trắng).

Mã hoá văn bản. Ba đặc trưng song hành:

- **BoW**: `CountVectorizer`.
- **TF–IDF**: `TfidfVectorizer`.
- **Embeddings**: `SentenceTransformer intfloat/multilingual-e5-base` (chuẩn hoá L2).

Tập mô hình. Ứng dụng huấn luyện tự động các mô hình:

$$\text{Naive Bayes},\ \text{KNN(Majority)},\ \text{KNN(Weighted)},\ \text{KNN(Custom } \alpha \in \{0.3, 0.5, 0.7\}),\ \text{Decision Tree}.$$

Mỗi mô hình được chạy trên cả 3 không gian đặc trưng; kết quả (Accuracy, Report, Confusion Matrix) và *model object* được lưu để phục vụ các tab phân tích.

4.4 KNN tuỳ biến trong ứng dụng

Động cơ. KNN truyền thống dễ *trệch* khi dữ liệu mất cân bằng và khi *similarity* không phản ánh đủ tầm quan trọng ngữ nghĩa. Do đó, biến thể KNN của nhóm dùng quy tắc bỏ phiếu có trọng số:

$$\text{weight} = (1 - \alpha) \cdot \text{similarity} \cdot \text{class\_weights} \ + \ \alpha \cdot \text{saliency},$$

trong đó similarity là *cosine similarity* (trên embeddings), class_weights là trọng số nghịch tần suất lớp, và saliency là độ quan trọng *instance-level* của mẫu kiểm tra. Tham số $\alpha \in [0, 1]$ điều tiết giữa *tín hiệu láng giềng* và *độ nổi bật* của mẫu.

Hiện thực. Lớp `CustomKNN` cài ba kiểu bỏ phiếu (`majority`, `weighted`, `custom`). Nhánh `custom`:

1. Tính similarity từ điểm kiểm tra đến toàn bộ huấn luyện (hoặc lân cận), lấy top-k.
2. Với từng láng giềng, cộng dồn trọng số theo công thức trên.
3. Tiên đoán lớp có tổng trọng số cao nhất.

Trọng số lớp class_weights giúp giảm thiên vị về lớp đông; saliency khiến KNN nhạy hơn với các *tín hiệu chủ đề* thực sự có ích.

Quan sát kết quả trong dashboard. Trên không gian **Embeddings**, `KNN (Custom` $\alpha = 0.7$`)` đạt **0.915** accuracy và là *best overall*.[1] Các tab *Comparison* và *Detailed Analysis* cho thấy ma trận nhầm lẫn cải thiện ở các cặp lớp dễ lẫn.

4.5 Chế độ *Text Classification Demo* và giải thích

Trong phần *Demo*, người dùng nhập *abstract* hoặc chọn mẫu. Hệ thống:

1. Tiền xử lý văn bản theo đúng pipeline huấn luyện.
2. Suy luận bằng mô hình tốt nhất ghi nhận từ phần *Overview*.
3. Hiển thị nhãn dự đoán, thông số mô hình, và văn bản đã chuẩn hoá.
4. (Tuỳ chọn) Gọi mô-đun *saliency* để vẽ *heatmap* tô đậm token/cụm từ đóng góp nhiều nhất vào lớp dự đoán, giúp người dùng *hiểu vì sao* abstract thuộc về chủ đề đó.

[1] Tổng quan số liệu đã liệt kê ở phần kết quả thí nghiệm.

4.6 Trải nghiệm người dùng và khả năng mở rộng

Tương tác/Trực quan. Ứng dụng dùng `Plotly` để:

- So sánh Accuracy theo mô hình, tô màu theo bộ mã hoá.
- Xem chi tiết chỉ số (macro/weighted F1) ở dạng bảng động.
- Hiển thị *confusion matrix* có nhãn trục rõ ràng cho 5 lớp mục tiêu.

Tối ưu vận hành. `@st.cache_data` giúp:

1. Lưu đệm dữ liệu đã tải và kết quả vector hoá để *giảm thời gian chờ*.
2. Tái sử dụng kết quả huấn luyện khi người dùng chuyển tab.

Lộ trình phát triển.

- **Explainability nâng cao**: hiển thị *token-level saliency* cùng biểu đồ *class-score drop* theo thuật toán leave-one-token-out; ghép cùng *topic TF–IDF words* để so khớp thuật ngữ đặc trưng.
- **Model Hub**: lựa chọn thêm bộ mã hoá (SBERT đa ngữ, E5-large) và bộ phân loại (LogReg/SVM/RF) để so sánh.
- **Tối ưu hoá tốc độ**: tiền tính láng giềng bằng FAISS cho không gian *Embeddings* và lưu *index* vào cache.

4.7 Tóm tắt đóng góp của phần `Streamlit`

- **Kết nối đủ sâu với pipeline**: từ tải dữ liệu, vector hoá, huấn luyện, so sánh kết quả đến suy luận thử.
- **Hỗ trợ quyết định mô hình**: biểu đồ tương tác, bảng chỉ số, và ma trận nhầm lẫn giúp chọn cấu hình tối ưu.
- **Giải thích được**: tích hợp *saliency heatmap* để minh hoạ *vì sao* một abstract được xếp vào lớp dự đoán.
- **Dễ mở rộng**: thêm mô hình/bộ mã hoá, FAISS indexing, hoặc các tiện ích giám sát lỗi và làm sạch dữ liệu.