TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

KHOA TOÁN - CƠ - TIN HỌC

___***___

Báo Cáo Giữa Kỳ
Đề Tài: Nhận dạng trang phục

Học phần: Học Máy

Sinh viên thực hiện:

Lê Văn Nam - 21002158

Phạm Hùng Anh - 20001887

Giảng viên hướng dẫn:

Cao Văn Chung

Mục lục

Danh sách hình vẽ

Danh sách bảng

1 Giới thiệu

1.1 Tổng Quan và Bối Cảnh

Nhận diện trang phục từ hình ảnh là một bài toán nền tảng và ngày càng quan trọng trong lĩnh vực thị giác máy tính và trí tuệ nhân tạo. Mục tiêu chính của bài toán này là xây dựng các hệ thống có khả năng tự động phân loại chính xác các loại quần áo, giày dép, và phụ kiện thời trang từ dữ liệu hình ảnh đầu vào. Sự phát triển của thương mại điện tử, mạng xã hội, và ngành công nghiệp thời trang đã thúc đẩy nhu cầu về các giải pháp nhận dạng trang phục hiệu quả. Các ứng dụng thực tiễn rất đa dạng, bao gồm việc cải thiện hệ thống gợi ý sản phẩm cho người dùng trực tuyến, cho phép tìm kiếm sản phẩm tương tự dựa trên hình ảnh, hỗ trợ phân tích xu hướng thời trang từ dữ liệu lớn, và tối ưu hóa quản lý kho hàng trong ngành bán lẻ.

Tuy nhiên, việc xây dựng một hệ thống nhận dạng trang phục mạnh mẽ phải đối mặt với nhiều thách thức đáng kể. Sự đa dạng về kiểu dáng, chất liệu, màu sắc, và hoa văn của các sản phẩm thời trang là rất lớn. Cùng một loại trang phục (ví dụ: áo sơ mi) có thể có vô số biến thể khác nhau. Thêm vào đó, cách sản phẩm được trình bày trong ảnh (góc chụp, ánh sáng, người mẫu, nền ảnh) cũng ảnh hưởng lớn đến quá trình nhận dạng. Các yếu tố như sự che khuất, biến dạng, và chất lượng ảnh thấp cũng góp phần làm tăng độ khó của bài toán.

Để thúc đẩy nghiên cứu và cung cấp một cơ sở dữ liệu chuẩn (benchmark) cho việc đánh giá các thuật toán, tập dữ liệu Fashion MNIST đã được giới thiệu. Tập dữ liệu này bao gồm 70,000 ảnh thang độ xám (60,000 cho huấn luyện và 10,000 cho kiểm tra) thuộc 10 danh mục trang phục khác nhau (áo thun, quần dài, áo len, váy, áo khoác, dép sandal, áo sơ mi, giày thể thao, túi xách, và bốt cổ chân). Với kích thước ảnh nhỏ (28x28 pixel) nhưng độ phức tạp cao hơn đáng kể so với tập dữ liệu MNIST chữ số viết tay cổ điển, Fashion MNIST đã trở thành một lựa chọn phổ biến để thử nghiệm và so sánh các mô hình học máy và học sâu trong các bài toán phân loại ảnh cơ bản.

1.2 Mục Tiêu và Phạm Vi Báo cáo

Báo cáo này tập trung vào việc áp dụng, triển khai và đánh giá hiệu năng của một số phương pháp học máy và học sâu tiêu biểu cho bài toán nhận dạng 10 loại trang phục trên tập dữ liệu Fashion MNIST. Mục tiêu chính là xây dựng các mô hình phân loại có

khả năng dự đoán nhãn trang phục từ ảnh đầu vào và so sánh hiệu quả giữa các cách tiếp cận khác nhau.

Các phương pháp được lựa chọn để khảo sát bao gồm các thuật toán kinh điển, kỹ thuật học sâu hiện đại và phương pháp học không giám sát, nhằm cung cấp cái nhìn toàn diện về cách tiếp cận bài toán từ các góc độ khác nhau:

- **Các thuật toán học máy cổ điển: Naïve Bayes** (với các biến thể Gaussian và Multinomial) và **Support Vector Machine (SVM)** (với kernel tuyến tính và kernel phi tuyến RBF). Các phương pháp này đại diện cho cách tiếp cận dựa trên lý thuyết xác suất và tối ưu hóa biên phân lớp.
- **Mô hình học sâu: Mạng Nơ-ron Tích Chập (Convolutional Neural Network)**, một kiến trúc hiện đại và rất thành công trong các bài toán thị giác máy tính, có khả năng tự động học các đặc trưng phân cấp từ dữ liệu ảnh.
- **Học không giám sát và Giảm chiều: Phân tích thành phần chính (PCA)** để giảm chiều và trực quan hóa dữ liệu, **K-Means** và **DBSCAN** để khám phá cấu trúc cụm trong dữ liệu mà không cần nhãn.

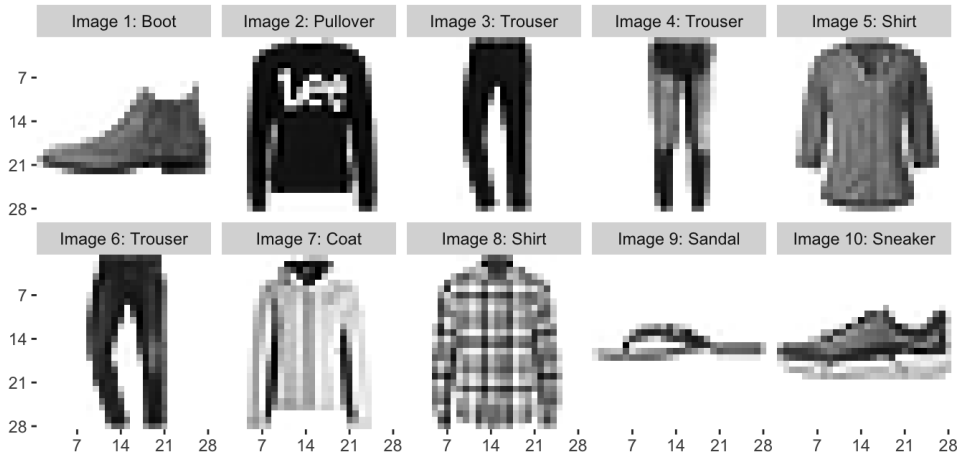
Bài báo cáo sẽ thực hiện huấn luyện các mô hình này trên tập huấn luyện của Fashion MNIST và đánh giá hiệu năng của chúng trên tập kiểm tra độc lập. Việc đánh giá không chỉ dựa trên độ chính xác tổng thể (accuracy) mà còn phân tích chi tiết hơn thông qua ma trận nhầm lẫn (confusion matrix) để hiểu rõ hơn về các lỗi phân loại cụ thể của từng mô hình đối với từng loại trang phục. Các chỉ số khác như precision, recall, và F1-score cũng được xem xét để có cái nhìn toàn diện hơn.

Thông qua việc so sánh kết quả thực nghiệm, báo cáo hướng tới việc đưa ra những nhận định về ưu điểm, nhược điểm và sự phù hợp của từng phương pháp đối với bài toán nhận dạng trang phục cụ thể này. Các công cụ và thư viện chính được sử dụng bao gồm Python, Scikit-learn, TensorFlow/Keras, Matplotlib và Seaborn.

2 Dữ liệu

2.1 Tổng quan về tập dữ liệu Fashion MNIST

Fashion MNIST là một bộ dữ liệu được tạo ra như một sự thay thế phức tạp hơn cho bộ dữ liệu MNIST cổ điển (chứa các chữ số viết tay từ 0-9). Fashion MNIST bao gồm ảnh thang độ xám của 10 loại trang phục khác nhau, thường được sử dụng để huấn luyện và đánh giá các mô hình phân loại ảnh.



Hình 1. Một số ví dụ từ tập dữ liệu Fashion MNIST.

Fashion-MNIST chứa 60.000 hình ảnh đào tạo và 10.000 hình ảnh thử nghiệm về hình ảnh bài viết của Zalando. Bộ dữ liệu bao gồm các hình ảnh thang độ xám có kích thước 28x28 pixel. Mỗi pixel có một giá trị pixel duy nhất liên quan đến nó, biểu thị độ sáng hoặc tối của pixel đó, với số cao hơn có nghĩa là tối hơn. Giá trị pixel này là một số nguyên từ 0 đến 255.

Mỗi ví dụ đào tạo và kiểm tra được gán cho một trong các nhãn sau:

- 0: T-shirt/top (Áo thun/Áo kiểu)
- 1: Trouser (Quần dài)
- 2: Pullover (Áo len chui đầu)

- 3: Dress (Váy liền)
- 4: Coat (Áo khoác)
- 5: Sandal (Dép sandal)
- 6: Shirt (Áo sơ mi)
- 7: Sneaker (Giày thể thao)
- 8: Bag (Túi xách)
- 9: Ankle boot (Bốt cổ chân)

2.2 Cấu trúc khối MNIST

Fashion MNIST có thể được hiểu là một tập hợp các điểm dữ liệu trong một không gian 784 chiều, tương tự như MNIST truyền thống. Mỗi ảnh xám 28×28 pixel được "trải phẳng"(flatten) thành một vector 784 chiều, trong đó:

- Mỗi chiều tương ứng với một pixel cụ thể trong ảnh.
- Giá trị của chiều đó (từ 0 đến 255) đại diện cho độ sáng của pixel (0: đen, 255: trắng, các giá trị giữa: xám).

Ví dụ minh họa: Giả sử ta có 4 ảnh từ Fashion MNIST:

1. Ảnh 1 (Áo thun): Pixel tại vị trí (10, 10) có giá trị 255 (trắng).
2. Ảnh 2 (Quần dài): Cùng pixel (10, 10) có giá trị 128 (xám trung bình).
3. Ảnh 3 (Áo len): Pixel (10, 10) có giá trị 64 (xám đậm).
4. Ảnh 4 (Giày): Pixel (10, 10) có giá trị 0 (đen).

Trong không gian 784 chiều, chiều tương ứng với pixel (10, 10) sẽ có giá trị khác nhau tùy ảnh, phản ánh sự biến đổi về hình dạng và texture của các lớp quần áo.

2.3 Cấu trúc tập dữ liệu được sử dụng

Nhóm sử dụng tập dữ liệu gốc gồm 4 tệp có định dạng `.gz`. Sau khi thực hiện quá trình đọc dữ liệu, ta nhận được dữ liệu có cấu trúc như sau:

- **Tập huấn luyện images (X_train):** Mảng NumPy (hoặc tương đương) kích thước (60000, 784). Mỗi hàng là một ảnh đã được trải phẳng. Giá trị pixel: Số nguyên từ 0 đến 255 (thường là kiểu `uint8`).
- **Tập huấn luyện labels (y_train):** Mảng NumPy kích thước (60000,). Chứa nhãn số nguyên từ 0 đến 9 tương ứng với `X_train`. (thường là kiểu `int64`).
- **Tập kiểm tra images (X_test):** Mảng NumPy kích thước (10000, 784). Cấu trúc tương tự `X_train`.
- **Tập kiểm tra labels (y_test):** Mảng NumPy kích thước (10000,). Chứa nhãn tương ứng với `X_test`.

Tiền xử lý thông thường: Mặc dù không được nêu rõ trong báo cáo gốc, các bước tiền xử lý chuẩn cho dữ liệu ảnh như Fashion MNIST thường bao gồm:

- Chuyển đổi kiểu dữ liệu sang số thực (ví dụ: `float32`).
- Chuẩn hóa giá trị pixel về khoảng [0, 1] bằng cách chia tất cả các giá trị cho 255. Bước này rất quan trọng đối với hiệu suất của nhiều thuật toán học máy (như K-Means, SVM, CNN) và PCA.
- Đối với CNN, dữ liệu ảnh thường được giữ ở dạng 2D (28x28) hoặc thêm một chiều kênh (28x28x1) thay vì trải phẳng.

Trong các phân tích sau, chúng tôi giả định rằng dữ liệu đã được chuẩn hóa phù hợp với yêu cầu của từng thuật toán, đặc biệt là việc chia tỷ lệ giá trị pixel.

3 Các phương pháp và mô hình

3.1 Giảm chiều dữ liệu

3.1.1 PCA - Phân tích thành phần chính

Principal Component Analysis (PCA) là phương pháp giảm chiều dữ liệu tuyến tính. Nó tìm một hệ cơ sở trực chuẩn mới sao cho phương sai (thông tin) của dữ liệu tập trung chủ yếu ở một vài tọa độ (thành phần chính) đầu tiên. Mục tiêu là biểu diễn dữ liệu trong không gian ít chiều hơn mà vẫn giữ lại được nhiều thông tin nhất có thể.

Các bước trong PCA:

1. **Tính vector kỳ vọng (trung bình)** $\bar{\mathbf{x}}$ của toàn bộ dữ liệu $\mathbf{X} = \{\mathbf{x}_1, ..., \mathbf{x}_N\}$:

$$\bar{\mathbf{x}} = \frac{1}{N} \sum_{n=1}^{N} \mathbf{x}_n$$

2. **Chuẩn hóa dữ liệu** (trừ vector trung bình): Tạo dữ liệu có tâm tại gốc tọa độ.

$$\tilde{\mathbf{x}}_n = \mathbf{x}_n - \bar{\mathbf{x}}$$

   Gọi $\tilde{\mathbf{X}}$ là ma trận dữ liệu đã chuẩn hóa (mỗi hàng là một $\tilde{\mathbf{x}}_n^T$).

3. **Tính ma trận hiệp phương sai S** (covariance matrix):

$$\mathbf{S} = \frac{1}{N-1} \sum_{n=1}^{N} \tilde{\mathbf{x}}_n \tilde{\mathbf{x}}_n^T = \frac{1}{N-1} \tilde{\mathbf{X}}^T \tilde{\mathbf{X}}$$

   (Lưu ý: Đôi khi người ta dùng $1/N$ thay vì $1/(N-1)$, đặc biệt khi N lớn. Kích thước của $\mathbf{S}$ là $d \times d$, với d là số chiều ban đầu).

4. **Phân tích trị riêng (Eigen-decomposition):** Tính các trị riêng λ_i và vector riêng $\mathbf{u}_i$ tương ứng của ma trận hiệp phương sai $\mathbf{S}$:

$$\mathbf{S}\mathbf{u}_i = \lambda_i \mathbf{u}_i$$

   Các vector riêng $\mathbf{u}_i$ (gọi là các thành phần chính) tạo thành một cơ sở trực chuẩn mới. Các trị riêng λ_i cho biết phương sai của dữ liệu dọc theo hướng vector riêng tương ứng.

5. **Sắp xếp và Chọn thành phần chính:** Sắp xếp các vector riêng theo thứ tự trị riêng giảm dần: $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_d \geq 0$. Chọn k vector riêng đầu tiên $(\mathbf{u}_1, ..., \mathbf{u}_k)$ ứng với k trị riêng lớn nhất để tạo ma trận chiếu $\mathbf{U}_k = [\mathbf{u}_1 \dots \mathbf{u}_k]$ (kích thước $d \times k$). Việc chọn k thường dựa trên tỷ lệ phương sai tích lũy mong muốn giữ lại (ví dụ: 95%).

6. **Chiếu dữ liệu:** Chiếu dữ liệu đã chuẩn hóa $\tilde{\mathbf{x}}_n$ xuống không gian con k chiều:

$$\mathbf{z}_n = \mathbf{U}_k^T \tilde{\mathbf{x}}_n$$

   $\mathbf{z}_n$ (kích thước $k \times 1$) là biểu diễn mới của $\mathbf{x}_n$ trong không gian k chiều.

7. **(Tùy chọn) Tái tạo dữ liệu gốc:** Dữ liệu gốc có thể được xấp xỉ lại từ dữ liệu đã giảm chiều:

$$\mathbf{x}_n^{approx} = \mathbf{U}_k \mathbf{z}_n + \bar{\mathbf{x}}$$

3.1.2 t-SNE - Nhúng lân cận phân phối t

t-Distributed Stochastic Neighbor Embedding (t-SNE) là một phương pháp giảm chiều phi tuyến, thường dùng để trực quan hóa dữ liệu có chiều cao trong không gian 2D hoặc 3D. Khác với PCA, t-SNE tập trung vào việc bảo toàn cấu trúc lân cận cục bộ (local structure) thay vì cấu trúc toàn cục. Ý tưởng chính của t-SNE là ánh xạ các điểm dữ liệu sao cho các điểm gần nhau trong không gian gốc vẫn gần nhau trong không gian ánh xạ.

Các bước chính của t-SNE:

1. **Tính phân phối xác suất trong không gian gốc:** Với mỗi cặp điểm $\mathbf{x}_i$ và $\mathbf{x}_j$, định nghĩa xác suất có điều kiện $p_{j|i}$ biểu thị mức độ giống nhau giữa hai điểm dựa trên phân phối Gaussian:

$$p_{j|i} = \frac{\exp\left(-\frac{||\mathbf{x}_i - \mathbf{x}_j||^2}{2\sigma_i^2}\right)}{\sum_{k \neq i} \exp\left(-\frac{||\mathbf{x}_i - \mathbf{x}_k||^2}{2\sigma_i^2}\right)}$$

   Tham số σ_i được điều chỉnh sao cho độ rối (*perplexity*) của phân phối $P_i = \{p_{j|i}\}_j$ đạt giá trị cho trước:

$$\text{Perplexity}(P_i) = 2^{H(P_i)}, \quad H(P_i) = -\sum_j p_{j|i} \log_2 p_{j|i}$$

2. **Chuyển đổi thành phân phối đối xứng:** Xây dựng phân phối xác suất đối xứng p_{ij}:

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2N}$$

với N là số lượng điểm dữ liệu.

3. **Tính phân phối xác suất trong không gian ánh xạ:** Mỗi điểm $\mathbf{x}_i$ được ánh xạ thành điểm $\mathbf{y}_i$ trong không gian chiều thấp. t-SNE sử dụng phân phối Student's t với 1 bậc tự do để đo mức độ tương tự giữa các điểm nhúng:

$$q_{ij} = \frac{(1 + ||\mathbf{y}_i - \mathbf{y}_j||^2)^{-1}}{\sum_{k \neq l}(1 + ||\mathbf{y}_k - \mathbf{y}_l||^2)^{-1}}$$

4. **Tối ưu hóa hàm mất mát:** Mục tiêu là làm cho phân phối $Q = \{q_{ij}\}$ gần với phân phối $P = \{p_{ij}\}$ bằng cách tối thiểu hóa độ đo phân kỳ Kullback-Leibler (KL divergence):

$$KL(P||Q) = \sum_{i \neq j} p_{ij} \log \frac{p_{ij}}{q_{ij}}$$

Tối ưu hóa thường được thực hiện bằng gradient descent với một số kỹ thuật hỗ trợ như: tăng hệ số phóng đại (early exaggeration), giảm tốc độ học dần (learning rate decay), hoặc dùng momentum.

5. **Trực quan hóa kết quả:** Sau khi tối ưu, các điểm nhúng $\mathbf{y}_i$ có thể được vẽ trong mặt phẳng 2D hoặc không gian 3D. Các cụm (clusters) dữ liệu thường trở nên dễ quan sát, hỗ trợ phân tích cấu trúc dữ liệu.

Lưu ý: t-SNE không bảo toàn khoảng cách tuyệt đối hay cấu trúc toàn cục, và không dùng được cho các tác vụ học máy sau giảm chiều. Tuy nhiên, nó rất mạnh trong việc khám phá và trực quan hóa cụm dữ liệu.

3.2 Phân Cụm

Phân cụm là kỹ thuật học không giám sát nhằm nhóm các điểm dữ liệu tương tự nhau vào cùng một cụm.

3.2.1 K-Means

K-Means là thuật toán phân cụm dựa trên việc tối thiểu hóa tổng bình phương khoảng cách nội cụm (within-cluster sum of squares - WCSS), hay còn gọi là Inertia.

Giả sử có N điểm dữ liệu $\mathbf{X} = \{\mathbf{x}_1, ..., \mathbf{x}_N\} \subset \mathbb{R}^d$ và muốn chia thành K cụm. Cần tìm K tâm cụm $\mathbf{M} = \{\mathbf{m}_1, ..., \mathbf{m}_K\}$ và ma trận gán nhãn $\mathbf{Y}$ (với $y_{ij} = 1$ nếu $\mathbf{x}_i$ thuộc cụm j, ngược lại $y_{ij} = 0$).

Hàm mất mát (WCSS):

$$\mathcal{L}(\mathbf{Y}, \mathbf{M}) = \sum_{i=1}^{N} \sum_{j=1}^{K} y_{ij} \|\mathbf{x}_i - \mathbf{m}_j\|^2$$

Bài toán tối ưu: $\min_{\mathbf{Y},\mathbf{M}} \mathcal{L}(\mathbf{Y}, \mathbf{M})$.

Thuật toán K-Means (Lloyd's algorithm):

1. **Khởi tạo (Initialization):** Chọn K điểm làm các tâm cụm ban đầu $\mathbf{m}_1, ..., \mathbf{m}_K$ (ví dụ: chọn ngẫu nhiên K điểm dữ liệu, hoặc dùng K-Means++).

2. **Gán nhãn (Assignment Step):** Gán mỗi điểm dữ liệu $\mathbf{x}_i$ vào cụm có tâm gần nó nhất:
$$y_{ik} = \begin{cases} 1 & \text{if } k = \arg\min_j \|\mathbf{x}_i - \mathbf{m}_j\|^2 \\ 0 & \text{otherwise} \end{cases}$$

3. **Cập nhật tâm (Update Step):** Tính lại vị trí tâm $\mathbf{m}_j$ cho mỗi cụm j bằng trung bình cộng của tất cả các điểm được gán vào cụm đó:
$$\mathbf{m}_j = \frac{\sum_{i=1}^{N} y_{ij} \mathbf{x}_i}{\sum_{i=1}^{N} y_{ij}}$$
(Nếu một cụm không có điểm nào, cần xử lý riêng, ví dụ: giữ nguyên tâm cũ hoặc chọn lại tâm).

4. **Lặp lại:** Quay lại Bước 2 và 3 cho đến khi các tâm cụm không thay đổi đáng kể (hội tụ) hoặc số vòng lặp tối đa đạt được.

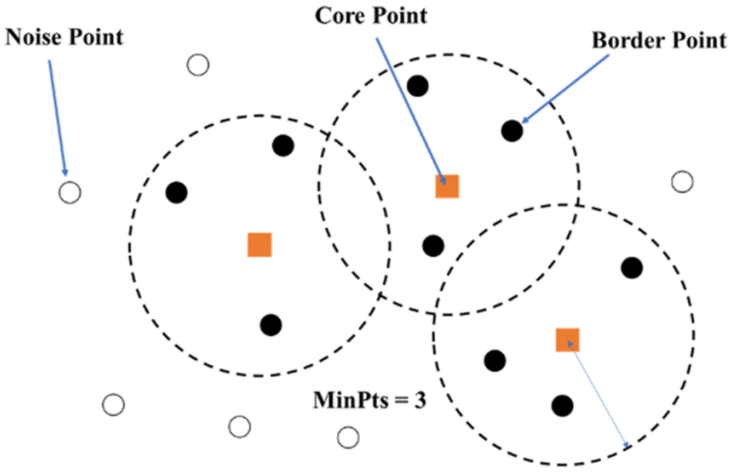
Việc chọn K tối ưu thường được thực hiện bằng phương pháp Elbow (quan sát sự thay đổi của WCSS khi tăng K) hoặc chỉ số Silhouette.

3.2.2 DBSCAN

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) là thuật toán phân cụm dựa trên mật độ. Nó nhóm các điểm gần nhau thành cụm và đánh dấu các điểm ở vùng mật độ thấp là nhiễu (noise). DBSCAN không yêu cầu xác định trước số cụm K.

Các khái niệm chính:

- **ϵ-neighborhood:** Tập hợp các điểm nằm trong bán kính ϵ của một điểm p (bao gồm cả p). Ký hiệu $N_\epsilon(p)$.
- **Core point:** Một điểm p là core point nếu $|N_\epsilon(p)| \geq MinPts$, tức là vùng lân cận ϵ của nó chứa ít nhất $MinPts$ điểm (tham số do người dùng xác định).
- **Border point:** Một điểm q là border point nếu nó không phải là core point, nhưng nằm trong ϵ-neighborhood của một core point p.
- **Noise point (Outlier):** Một điểm không phải là core point hay border point.
- **Density-reachable:** Diểm q là density-reachable từ điểm p nếu có một chuỗi các core points $p_1, ..., p_n$ với $p_1 = p$ và $p_n = q$ sao cho $p_{i+1} \in N_\epsilon(p_i)$.
- **Density-connected:** Hai điểm p và q là density-connected nếu có một core point o sao cho cả p và q đều density-reachable từ o.



Hình 2. Minh họa các loại điểm trong DBSCAN với MinPts=3.

Thuật toán DBSCAN:

1. Chọn một điểm p chưa được xét.
2. Tìm tất cả các điểm density-reachable từ p sử dụng ϵ và $MinPts$.
3. Nếu p là core point, một cụm mới được hình thành. Mở rộng cụm này bằng cách thêm vào tất cả các điểm density-reachable từ các core points trong cụm.
4. Nếu p là border point và chưa thuộc cụm nào, tạm thời đánh dấu là noise. Nó có thể được thêm vào một cụm sau nếu nó density-reachable từ một core point khác.

5. Nếu p là noise point, bỏ qua và xét điểm khác.

6. Lặp lại quá trình cho đến khi tất cả các điểm đã được xét.

Các tham số quan trọng là ϵ và *MinPts*, việc lựa chọn chúng ảnh hưởng lớn đến kết quả phân cụm.

3.3 Phân loại dữ liệu

Phân loại (classification) là một trong những nhiệm vụ cơ bản và quan trọng trong học máy, với mục tiêu là dự đoán nhãn (label) của một mẫu dữ liệu dựa trên đặc trưng (features) của nó. Có nhiều phương pháp phân loại khác nhau, mỗi phương pháp có giả định, ưu điểm và nhược điểm riêng phù hợp với từng loại dữ liệu và bài toán cụ thể.

3.3.1 Mạng Nơron Tích Chập (CNN)

Mạng nơron tích chập (Convolutional Neural Networks - CNN) là một loại mạng nơron sâu, đặc biệt hiệu quả trong việc xử lý dữ liệu dạng hình ảnh hoặc có cấu trúc dạng lưới. CNN có khả năng tự động học các đặc trưng từ dữ liệu thông qua các tầng tích chập, gộp và kết nối đầy đủ. Cấu trúc tổng quát của CNN bao gồm các thành phần sau:

- **Tầng tích chập (Convolutional Layer):** Thực hiện phép tích chập giữa ảnh đầu vào và các bộ lọc (filter) để trích xuất đặc trưng cục bộ như cạnh, góc hoặc mẫu hình lặp. Kết quả của tích chập được tính theo công thức:

$$W' = \frac{W - f + 2p}{s} + 1, \quad H' = \frac{H - f + 2p}{s} + 1$$

  Trong đó W, H là kích thước ảnh đầu vào, f là kích thước filter, p là padding và s là bước nhảy (stride). Sau khi tích chập, một hàm kích hoạt phi tuyến như ReLU thường được áp dụng:

$$\text{ReLU}(x) = \max(0, x)$$

- **Tầng gộp (Pooling Layer):** Dùng để giảm kích thước không gian của đầu ra từ tầng tích chập, giảm số lượng tham số và chi phí tính toán, đồng thời kiểm soát overfitting. Hai kỹ thuật gộp phổ biến là:

  - Max Pooling: lấy giá trị lớn nhất trong vùng lân cận.
  - Average Pooling: lấy giá trị trung bình trong vùng lân cận.

- **Tầng kết nối đầy đủ (Fully-Connected Layer):** Sau khi qua các tầng tích chập và gộp, dữ liệu được chuyển về dạng vector một chiều và đưa vào các tầng kết nối đầy đủ. Tầng cuối thường sử dụng hàm softmax để tính xác suất các lớp:

$$q_i = \frac{e^{z_i}}{\sum_{j=1}^{C} e^{z_j}}$$

- **Tối ưu hóa và Regularization:** Mạng CNN được huấn luyện bằng phương pháp lan truyền ngược để tối thiểu hàm mất mát, thường là cross-entropy:

$$H(\mathbf{p}, \mathbf{q}) = -\sum_{i=1}^{C} p_i \log q_i$$

  Các thuật toán tối ưu phổ biến bao gồm SGD, Adam, RMSProp,... Đồng thời, các kỹ thuật regularization như Dropout, L2, hoặc Data Augmentation giúp cải thiện khả năng tổng quát và tránh quá khớp.

3.3.2 Naïve Bayes

Naïve Bayes là một họ các thuật toán phân loại xác suất dựa trên định lý Bayes, giả định rằng các đặc trưng là độc lập có điều kiện với nhau. Dù giả định này thường không đúng trong thực tế, nhưng Naïve Bayes vẫn hoạt động hiệu quả và có tốc độ huấn luyện rất nhanh.

- **Định lý Bayes:** Với một mẫu dữ liệu đầu vào $\mathbf{X} = (x_1, x_2, ..., x_d)$, xác suất thuộc lớp y được tính theo định lý Bayes:

$$P(y|\mathbf{X}) = \frac{P(\mathbf{X}|y)P(y)}{P(\mathbf{X})}$$

  Vì $P(\mathbf{X})$ là hằng số với mọi y, mục tiêu là cực đại hóa tích $P(\mathbf{X}|y)P(y)$.

- **Giả định độc lập có điều kiện:** Naïve Bayes giả định rằng các đặc trưng x_i là độc lập khi đã biết lớp y:

$$P(\mathbf{X}|y) = \prod_{i=1}^{d} P(x_i|y)$$

- **Các biến thể phổ biến:**
  - **Gaussian Naïve Bayes:** Áp dụng cho đặc trưng liên tục, giả định phân phối chuẩn:
$$P(x_i|y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right)$$
  - **Multinomial Naïve Bayes:** Phù hợp với dữ liệu đếm (như số lần từ xuất hiện trong tài liệu).
  - **Bernoulli Naïve Bayes:** Sử dụng cho đặc trưng nhị phân (0 hoặc 1).

3.3.3 Support Vector Machine (SVM)

Support Vector Machine là một thuật toán học có giám sát, được sử dụng để phân loại và hồi quy. Mục tiêu chính của SVM là tìm một siêu phẳng (hyperplane) phân tách các lớp dữ liệu với khoảng cách lớn nhất.

- **SVM tuyến tính:** Trong trường hợp dữ liệu tuyến tính phân tách được, bài toán tối ưu hóa của SVM được mô tả như sau:

$$\min_{\mathbf{w},b} \quad \frac{1}{2}\|\mathbf{w}\|^2 \quad \text{subject to } y_n(\mathbf{w}^T\mathbf{x}_n + b) \geq 1$$

  Trong đó $\mathbf{w}$ là vector trọng số, b là hệ số dịch chuyển, và $y_n \in \{-1, +1\}$.

- **SVM phi tuyến và Kernel Trick:** Với dữ liệu không phân tách tuyến tính, SVM sử dụng các hàm kernel để ánh xạ dữ liệu vào không gian đặc trưng cao hơn:

$$k(\mathbf{x}, \mathbf{z}) = \phi(\mathbf{x})^T\phi(\mathbf{z})$$

  Các kernel thường dùng:
  - Linear: $k(\mathbf{x}, \mathbf{z}) = \mathbf{x}^T\mathbf{z}$
  - Polynomial: $k(\mathbf{x}, \mathbf{z}) = (\mathbf{x}^T\mathbf{z} + c)^d$
  - RBF (Gaussian): $k(\mathbf{x}, \mathbf{z}) = \exp(-\gamma\|\mathbf{x} - \mathbf{z}\|^2)$

  - Sigmoid: $k(\mathbf{x}, \mathbf{z}) = \tanh(\alpha \mathbf{x}^T \mathbf{z} + c)$

- **Phân loại đa lớp (Multi-class):** Vì SVM cơ bản là nhị phân, có hai cách mở rộng:

  - One-vs-Rest (OvR): Xây dựng K mô hình, mỗi mô hình phân biệt một lớp với các lớp còn lại.
  - One-vs-One (OvO): Xây dựng $K(K-1)/2$ mô hình cho mỗi cặp lớp.

4 Thực nghiệm

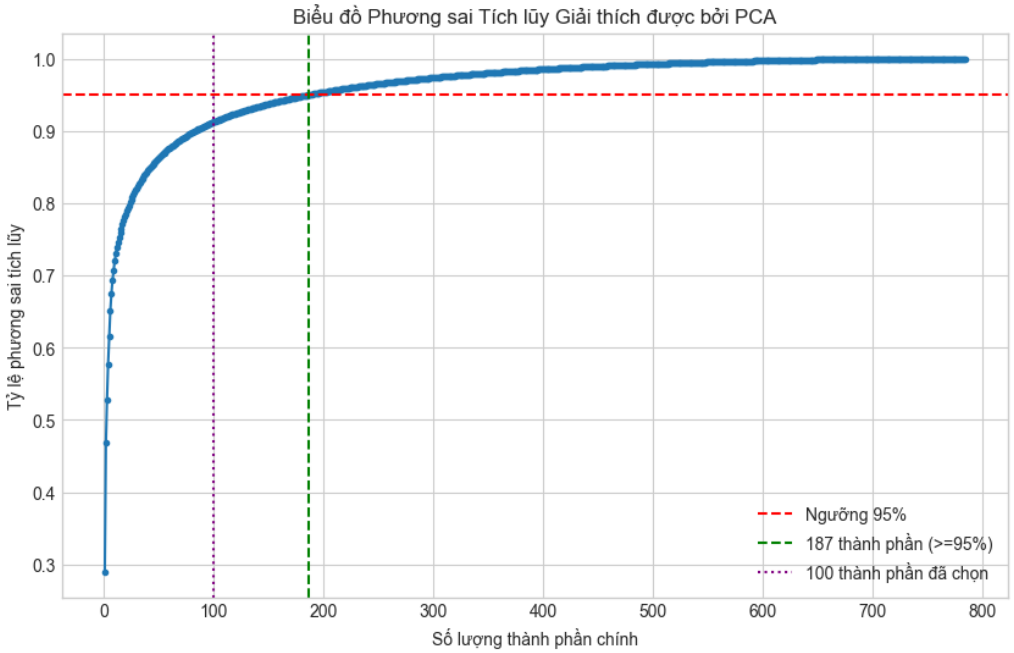
Phần này trình bày chi tiết quá trình thực nghiệm và kết quả thu được khi áp dụng các phương pháp giảm chiều, phân cụm và phân loại đã mô tả ở Phần 3 trên tập dữ liệu Fashion MNIST. Dữ liệu đầu vào bao gồm 60,000 ảnh huấn luyện và 10,000 ảnh kiểm tra, mỗi ảnh có kích thước 28x28 pixel thang độ xám. Trước khi đưa vào các mô hình (ngoại trừ Multinomial Naive Bayes yêu cầu dữ liệu gốc), giá trị pixel của ảnh được chuẩn hóa về khoảng [0, 1] bằng cách chia cho 255.

4.1 Giảm Chiều Dữ Liệu và Trực Quan Hóa

Mục tiêu của bước này là giảm số chiều của dữ liệu từ 784 chiều ban đầu xuống không gian thấp hơn để trực quan hóa cấu trúc dữ liệu và chuẩn bị cho các mô hình phân loại yêu cầu dữ liệu ít chiều hơn hoặc để tăng tốc độ tính toán.

4.1.1 Phân tích Thành phần chính (PCA)

PCA được áp dụng trên toàn bộ 60,000 ảnh huấn luyện đã chuẩn hóa. Quá trình phân tích trị riêng của ma trận hiệp phương sai cho ra 784 thành phần chính. Hình 3 thể hiện tỷ lệ phương sai tích lũy được giải thích bởi các thành phần chính đầu tiên.



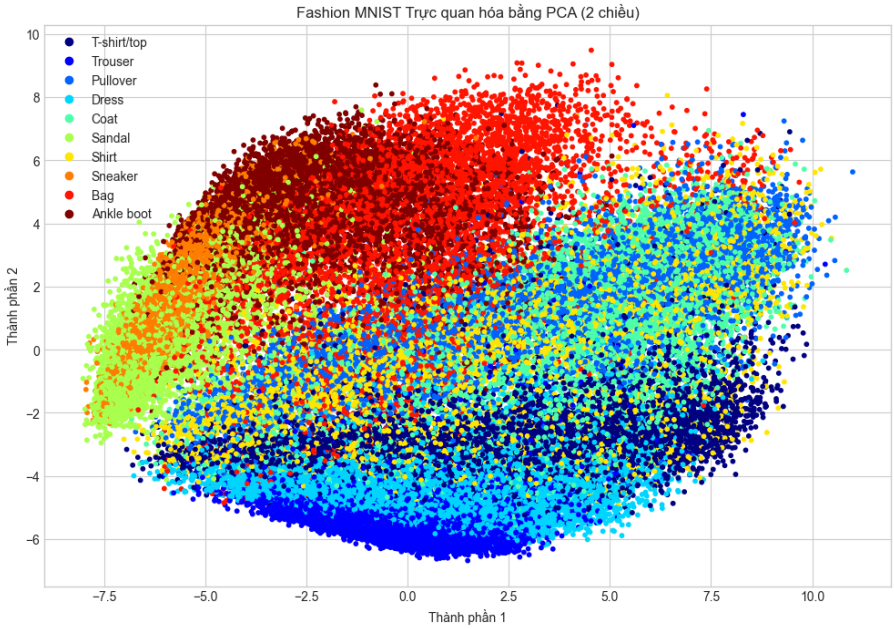
Hình 3. Biểu đồ Phương sai Tích lũy Giải thích được bởi PCA.

Quan sát biểu đồ cho thấy, cần khoảng 187 thành phần chính để giải thích được 95% phương sai của dữ liệu. Để cân bằng giữa việc giữ thông tin và giảm độ phức tạp, $k = 100$ thành phần chính được lựa chọn để sử dụng cho các thí nghiệm phân loại tiếp theo. Với $k = 100$, tỷ lệ phương sai tích lũy giữ lại được là khoảng 91,23%.

4.1.2 Trực quan hóa dữ liệu

Để trực quan hóa sự phân bố của các lớp dữ liệu, PCA và t-SNE được sử dụng để giảm chiều dữ liệu huấn luyện xuống còn 2 chiều.

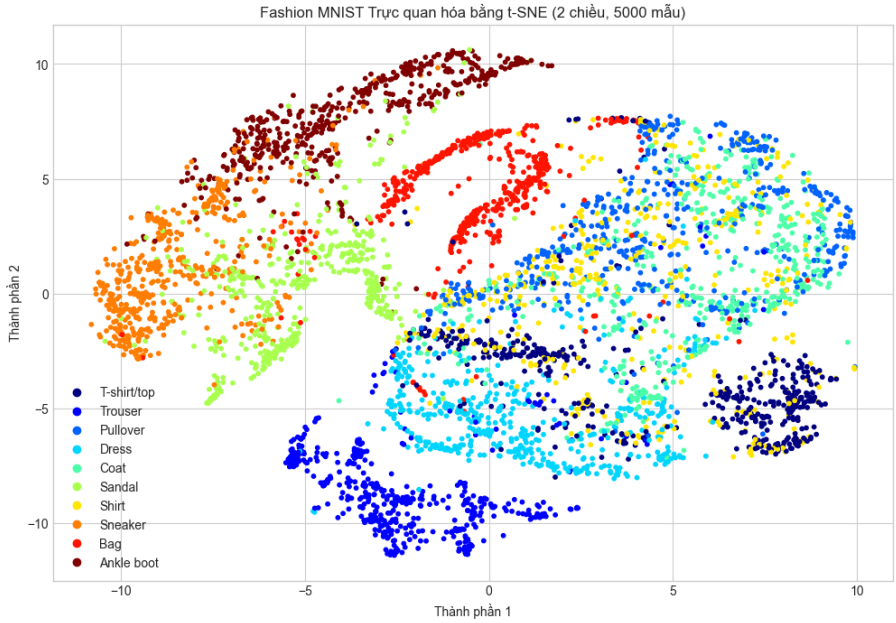
PCA (2 chiều): Hình 4 cho thấy kết quả chiếu dữ liệu lên 2 thành phần chính đầu tiên.



Hình 4. Trực quan hóa dữ liệu Fashion MNIST bằng PCA (2 chiều). Màu sắc thể hiện các lớp khác nhau.

Quan sát Hình 4, có thể thấy một số lớp như 'Ankle boot' (màu X), 'Bag' (màu Y), 'Trouser' (màu Z) có xu hướng tách biệt tương đối. Tuy nhiên, phần lớn các lớp, đặc biệt là các loại áo ('T-shirt/top', 'Pullover', 'Shirt', 'Coat'), bị chồng chéo lên nhau đáng kể, cho thấy việc phân tách tuyến tính trong không gian 2 chiều này là rất khó khăn.

t-SNE (2 chiều): t-SNE được thực hiện trên một tập con gồm 5,000 mẫu huấn luyện (do yêu cầu tính toán cao) với perplexity=30. Kết quả được thể hiện trong Hình 5.



Hình 5. Trực quan hóa dữ liệu Fashion MNIST (5,000 mẫu) bằng t-SNE (2 chiều, perplexity=30).

So với PCA, t-SNE (Hình 5) cho thấy khả năng phân tách các cụm tương ứng với các lớp tốt hơn rõ rệt trong không gian 2D. Các lớp riêng biệt hình thành các cụm khá dày đặc và tách biệt. Điều này nhấn mạnh khả năng của t-SNE trong việc bảo toàn cấu trúc lân cận cục bộ, rất hữu ích cho việc khám phá và trực quan hóa dữ liệu. Tuy nhiên, t-SNE không phù hợp để giảm chiều trước khi huấn luyện các mô hình khác do không có cơ chế biến đổi tường minh cho dữ liệu mới và không bảo toàn cấu trúc toàn cục.

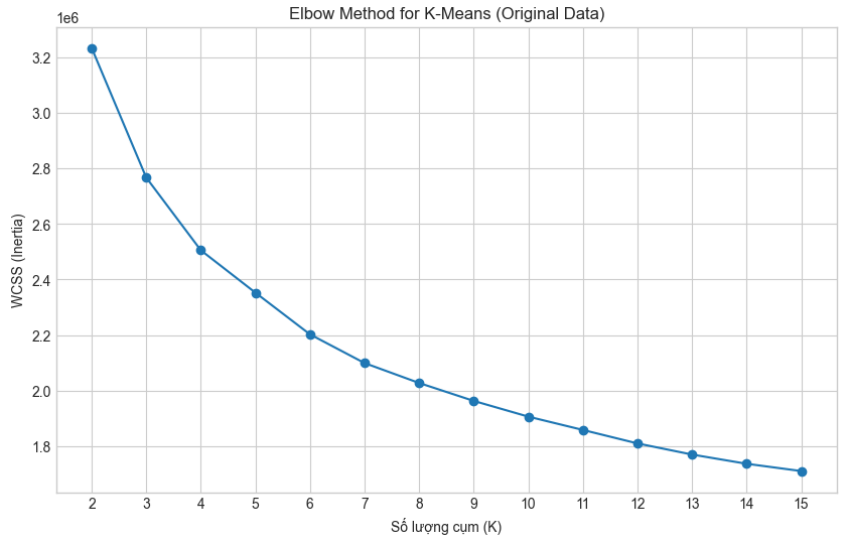
4.2 Phân cụm Dữ liệu (Unsupervised Learning)

Việc phân cụm dữ liệu được thực hiện trên tập huấn luyện bằng thuật toán K-Means và DBSCAN để khám phá cấu trúc nhóm tiềm ẩn trong dữ liệu mà không cần sử dụng nhãn lớp.

4.2.1 K-Means

K-Means được áp dụng với số cụm $K = 10$ (bằng số lớp gốc) trên cả dữ liệu gốc đã chuẩn hóa (784 chiều) và dữ liệu đã giảm chiều bằng PCA (100 chiều).

Phương pháp Elbow: Để đánh giá số cụm tối ưu K, phương pháp Elbow được sử dụng bằng cách tính tổng bình phương khoảng cách nội cụm (WCSS hay Inertia) với các giá trị K khác nhau (từ 2 đến 15) trên dữ liệu gốc. Kết quả được thể hiện trong Hình 6.



Hình 6. Phương pháp Elbow cho K-Means trên dữ liệu gốc Fashion MNIST.

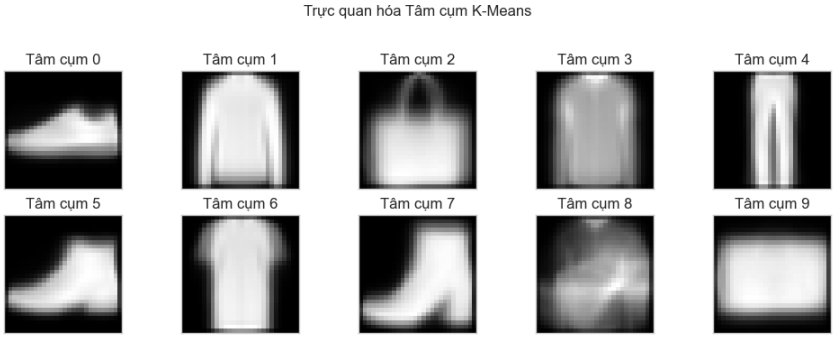
Biểu đồ Elbow (Hình 6) không cho thấy một điểm "khuỷu tay"(elbow) rõ ràng, đường WCSS giảm khá đều khi K tăng. Điều này gợi ý rằng cấu trúc cụm tự nhiên của dữ liệu có thể không hoàn toàn phù hợp với giả định của K-Means hoặc $K = 10$ không phải là lựa chọn tối ưu dựa trên tiêu chí WCSS.

Kết quả Phân cụm (K=10):

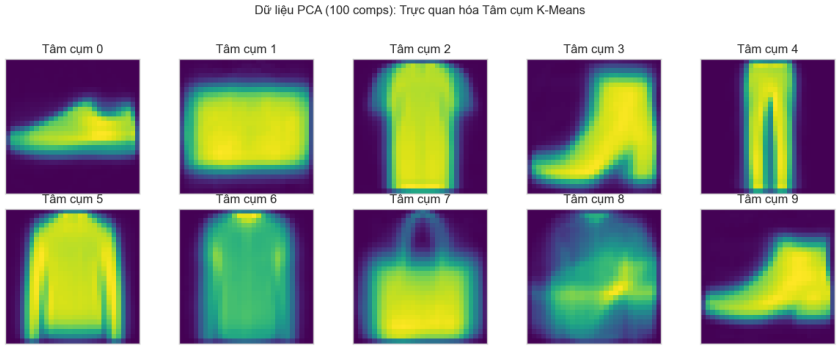
- **Trên dữ liệu gốc (784 chiều):**
  - WCSS (Inertia): 1906652.75
  - Adjusted Rand Index (ARI): 0.3479
  - Normalized Mutual Information (NMI): 0.5118
  - Homogeneity: 0.5004
  - Silhouette Score (ước tính trên 5000 mẫu): 0.15408119559288025
- **Trên dữ liệu PCA (100 chiều):**
  - WCSS (Inertia): 1548091.00
  - Adjusted Rand Index (ARI): 0.3480
  - Normalized Mutual Information (NMI): 0.5119
  - Homogeneity: 0.5005
  - Silhouette Score (ước tính trên 5000 mẫu): 0.17940902709960938

Các chỉ số ARI, NMI, Homogeneity cho thấy K-Means có khả năng tìm thấy một số cấu trúc tương ứng với các lớp gốc, nhưng mức độ phù hợp chỉ ở mức trung bình. Silhouette score khá thấp cho thấy các cụm có thể bị chồng lấn hoặc không đủ dày đặc. Việc giảm chiều bằng PCA làm giảm WCSS (do không gian ít chiều hơn) và tăng tốc độ tính toán, nhưng không cải thiện đáng kể các chỉ số đánh giá chất lượng cụm (ARI, NMI).

Trực quan hóa Kết quả Gán cụm: Hình 7 và 8 minh họa các điểm dữ liệu huấn luyện (chiếu lên không gian PCA 2D) được tô màu theo ID cụm được gán bởi K-Means tương ứng trên dữ liệu gốc và dữ liệu PCA.



Hình 7. Kết quả gán cụm K-Means (K=10) trên dữ liệu gốc (trực quan hóa PCA 2D).



Hình 8. Kết quả gán cụm K-Means (K=10) trên dữ liệu PCA 100 chiều (trực quan hóa PCA 2D).

4.2.2 DBSCAN

DBSCAN được thử nghiệm với các tham số $\epsilon = 3$ và $MinPts = 5$. Do DBSCAN nhạy cảm với số chiều cao và chi phí tính toán lớn, việc thực thi được ưu tiên trên dữ liệu đã giảm chiều bằng PCA (100 chiều) trước, sau đó mới chạy trên dữ liệu gốc.

Kết quả Phân cụm:

- **Trên dữ liệu PCA (100 chiều):**
  - Số điểm nhiễu (Noise points): 18510
  - Adjusted Rand Index (ARI): 0.0209
  - Normalized Mutual Information (NMI): 0.0637
  - Homogeneity: 0.0418
  - Silhouette Score: -0.3583763539791107
- **Trên dữ liệu gốc (784 chiều):**
  - Số điểm nhiễu (Noise points): 44930
  - Adjusted Rand Index (ARI): 0.0317
  - Normalized Mutual Information (NMI): 0.1280
  - Homogeneity: 0.0845
  - Silhouette Score: -0.42203256487846375

Kết quả cho thấy DBSCAN với các tham số đã chọn hoạt động rất kém trên tập dữ liệu này. Phần lớn các điểm dữ liệu bị coi là nhiễu (noise), và các chỉ số đánh giá (ARI, NMI, Homogeneity) cực kỳ thấp. Điều này cho thấy cấu trúc mật độ của Fashion MNIST không phù hợp với giả định của DBSCAN (các vùng dày đặc được phân tách rõ ràng bởi các vùng thưa) hoặc cần phải tinh chỉnh rất kỹ các tham số ϵ và $MinPts$, vốn là một công việc khó khăn, đặc biệt với dữ liệu nhiều chiều.

4.3 Phân loại Dữ liệu (Supervised Learning)

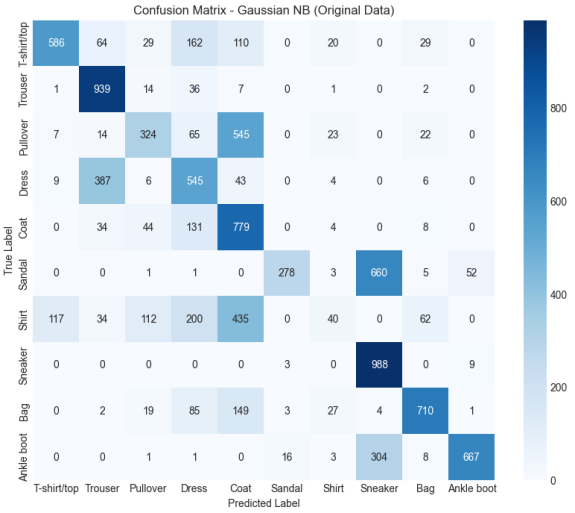
Các mô hình phân loại Naive Bayes, SVM và CNN được huấn luyện và đánh giá hiệu năng trên tập dữ liệu Fashion MNIST. Dữ liệu được chia thành tập huấn luyện (60,000 mẫu) và tập kiểm tra (10,000 mẫu). Các mô hình (trừ MNB) được huấn luyện trên cả dữ liệu gốc đã chuẩn hóa (784 chiều) và dữ liệu đã giảm chiều bằng PCA (100 chiều) để so sánh ảnh hưởng của việc giảm chiều.

4.3.1 Naive Bayes

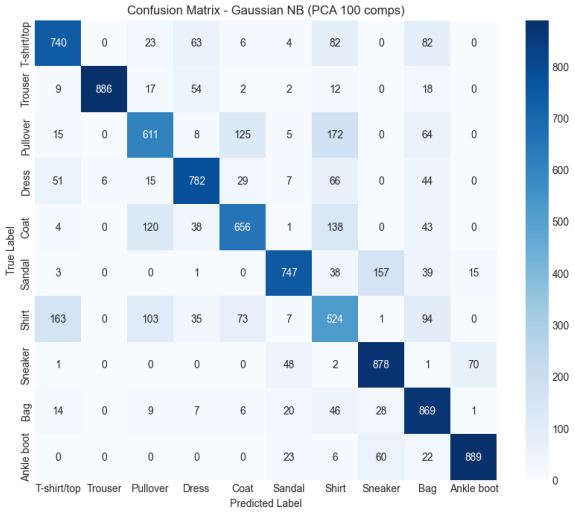
Gaussian Naive Bayes (GNB): Giả định các đặc trưng (pixel) tuân theo phân phối Gaussian và độc lập có điều kiện với lớp.

- **Trên dữ liệu gốc (784 chiều):** Độ chính xác trên tập kiểm tra đạt 59%. Ma trận nhầm lẫn được thể hiện trong Hình 9.

- **Trên dữ liệu PCA (100 chiều):** Độ chính xác trên tập kiểm tra đạt 76%. Ma trận nhầm lẫn được thể hiện trong Hình 10.



Hình 9. Confusion Matrix - GNB (Dữ liệu gốc).

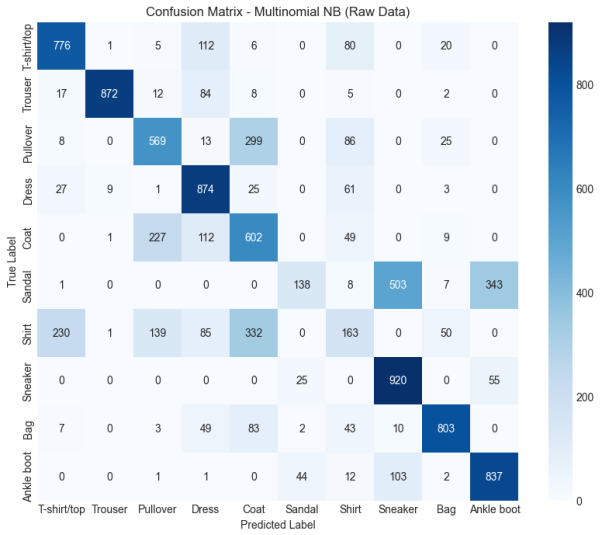


Hình 10. Confusion Matrix - GNB (Dữ liệu PCA 100 chiều).

Kết quả GNB khá thấp, đặc biệt trên dữ liệu gốc, cho thấy giả định Gaussian và độc lập không phù hợp với dữ liệu ảnh pixel. Việc giảm chiều bằng PCA giúp cải thiện đáng kể hiệu năng của GNB, có thể do các thành phần chính có phân phối gần với Gaussian hơn và giảm nhiễu.

Multinomial Naive Bayes (MNB): Thường dùng cho dữ liệu đếm, ở đây áp dụng trên dữ liệu pixel gốc (0-255) coi như các mức độ rời rạc.

- **Trên dữ liệu gốc (0-255):** Độ chính xác trên tập kiểm tra đạt 66%. Ma trận nhầm lẫn được thể hiện trong Hình 11.



Hình 11. Confusion Matrix - Multinomial NB (Dữ liệu gốc 0-255).

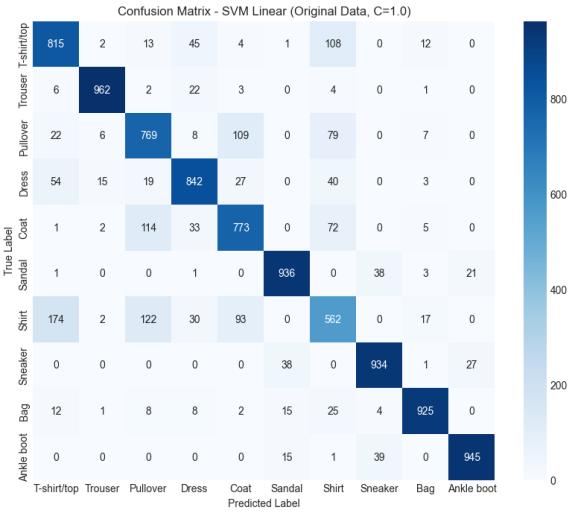
MNB cho kết quả tốt hơn đáng kể so với GNB, đạt độ chính xác trên 66%. Điều này gợi ý rằng việc xem xét giá trị pixel như các đặc trưng rời rạc hoặc tần suất phù hợp hơn giả định Gaussian trong bài toán này. Tuy nhiên, MNB vẫn bị hạn chế bởi giả định độc lập giữa các pixel.

4.3.2 Support Vector Machine (SVM)

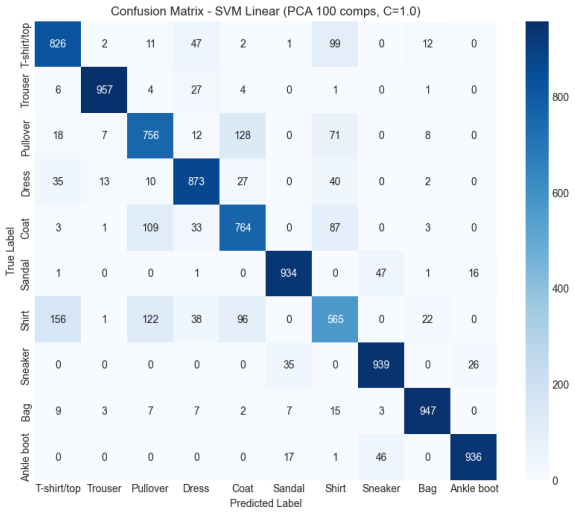
SVM được triển khai với kernel tuyến tính (Linear) và kernel RBF (Gaussian) sử dụng chiến lược One-vs-Rest cho phân loại đa lớp. Tham số $C = 1.0$ và $\gamma =$ 'scale' (cho RBF) là các giá trị mặc định được sử dụng.

SVM Linear:

- **Trên dữ liệu gốc (784 chiều):** Độ chính xác đạt 85%. Ma trận nhầm lẫn: Hình 12.
- **Trên dữ liệu PCA (100 chiều):** Độ chính xác đạt 85%. Ma trận nhầm lẫn: Hình 13.



Hình 12. Confusion Matrix - SVM Linear (Dữ liệu gốc).

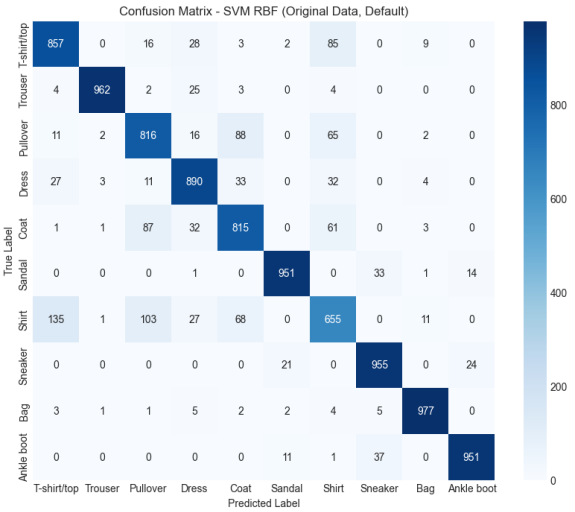


Hình 13. Confusion Matrix - SVM Linear (Dữ liệu PCA 100 chiều).

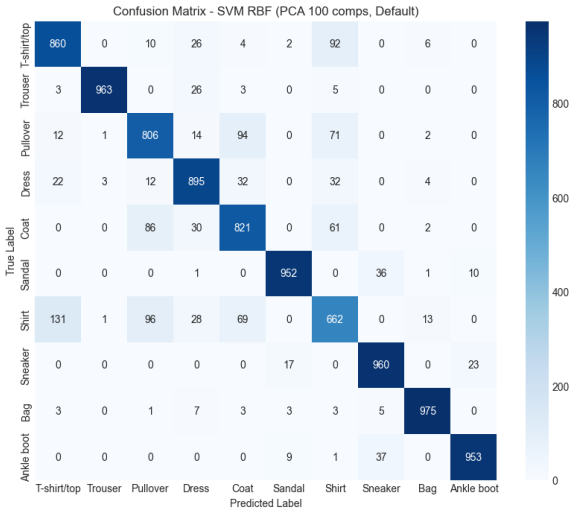
SVM Linear cho kết quả tốt, vượt trội hơn hẳn Naive Bayes. Giảm chiều bằng PCA không làm thay đổi đáng kể độ chính xác nhưng giúp giảm thời gian huấn luyện rất nhiều.

SVM RBF:

- **Trên dữ liệu gốc (784 chiều):** Độ chính xác đạt 88%. Ma trận nhầm lẫn: Hình 14.
- **Trên dữ liệu PCA (100 chiều):** Độ chính xác đạt 88%. Ma trận nhầm lẫn: Hình 15.



Hình 14. Confusion Matrix - SVM RBF (Dữ liệu gốc).



Hình 15. Confusion Matrix - SVM RBF (Dữ liệu PCA 100 chiều).

SVM với kernel RBF cho hiệu năng rất tốt, đạt gần 90% độ chính xác, cao hơn SVM Linear. Điều này cho thấy dữ liệu Fashion MNIST không hoàn toàn tách biệt tuyến tính và kernel phi tuyến RBF giúp tìm ra biên phân lớp phức tạp và hiệu quả hơn. Tương tự kernel Linear, PCA không ảnh hưởng nhiều đến độ chính xác nhưng tăng tốc độ huấn luyện. Kết quả chi tiết theo từng lớp cho SVM RBF (trên dữ liệu gốc) có thể được xem trong báo cáo phân loại sinh ra từ quá trình thực thi.

4.3.3 Mạng Nơron Tích Chập (CNN)

Một mô hình Mạng Nơron Tích Chập (CNN) được xây dựng và huấn luyện cho bài toán phân loại. Dữ liệu ảnh được giữ nguyên kích thước 2D (28x28) và thêm một chiều kênh (channel) để phù hợp với đầu vào của CNN.

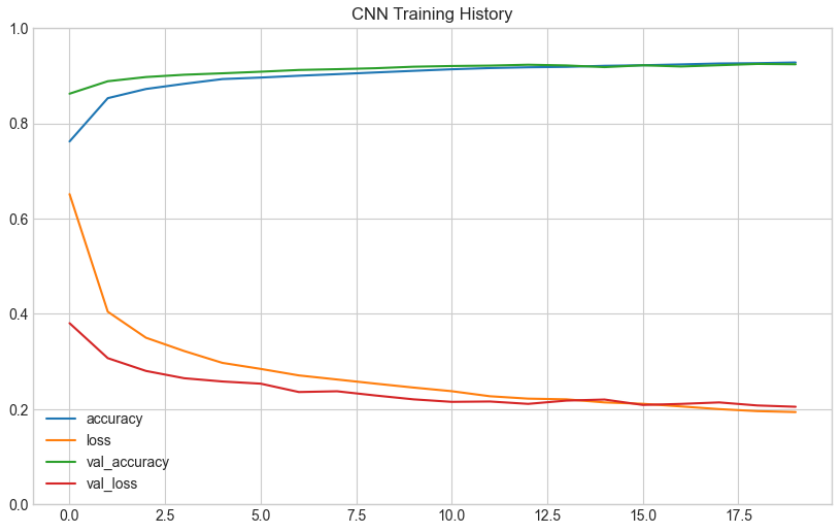
Kiến trúc Mô hình: Mô hình CNN bao gồm 3 tầng tích chập (Conv2D với 32, 32, 64 filters, kernel 3x3, hàm kích hoạt ReLU, padding 'same') xen kẽ với 3 tầng gộp cực đại (MaxPool2D, pool size 2x2). Sau đó là 2 tầng Dropout (tỷ lệ 0.4) để chống quá khớp (overfitting), một tầng Flatten, một tầng kết nối đầy đủ (Dense) với 128 units (ReLU) và tầng Dense cuối cùng với 10 units (softmax) cho đầu ra phân loại. Mô hình sử dụng optimizer 'adam' và hàm mất mát 'categorical_crossentropy'. Tóm tắt kiến trúc được trình bày trong Bảng 1.

Bảng 1. Tóm tắt kiến trúc mô hình CNN được sử dụng.

Layer (type)	Output Shape	Param #
conv2d_15 (Conv2D)	(None, 28, 28, 32)	320
max_pooling2d_15 (MaxPooling2D)	(None, 14, 14, 32)	0
conv2d_16 (Conv2D)	(None, 14, 14, 32)	9,248
max_pooling2d_16 (MaxPooling2D)	(None, 7, 7, 32)	0
onv2d_17 (Conv2D)	(None, 7, 7, 64)	18,496
max_pooling2d_17 (MaxPooling2D)	(None, 3, 3, 64)	0
dropout_10 (Dropout)	(None, 3, 3, 64)	0
flatten_5 (Flatten)	(None, 576)	0
dense_10 (Dense)	(None, 128)	73,856
dropout_11 (Dropout)	(None, 128)	0
dense_11 (Dense)	(None, 10)	1,290
Total params:		**103,210**

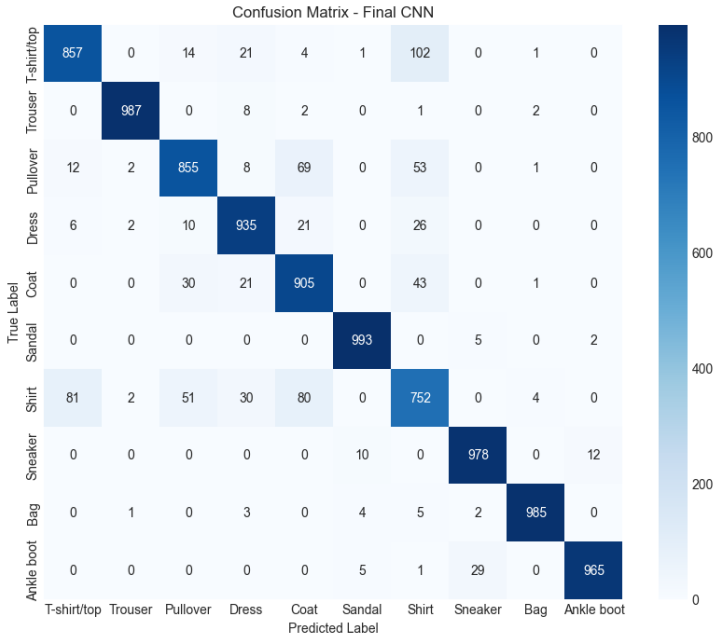
Huấn luyện và Đánh giá: Mô hình được đánh giá sơ bộ bằng K-Fold Cross Validation (K=5) trên tập huấn luyện, mỗi fold huấn luyện trong 4 epochs. Độ chính xác trung bình trên tập validation đạt khoảng 89.60% (+/- 0.35%), cho thấy mô hình học khá tốt và ổn định ngay cả với số epoch ít.

Sau đó, mô hình cuối cùng được huấn luyện trên toàn bộ 60,000 ảnh huấn luyện trong 20 epochs. Lịch sử huấn luyện (loss và accuracy trên tập huấn luyện và validation) được thể hiện trong Hình 16.



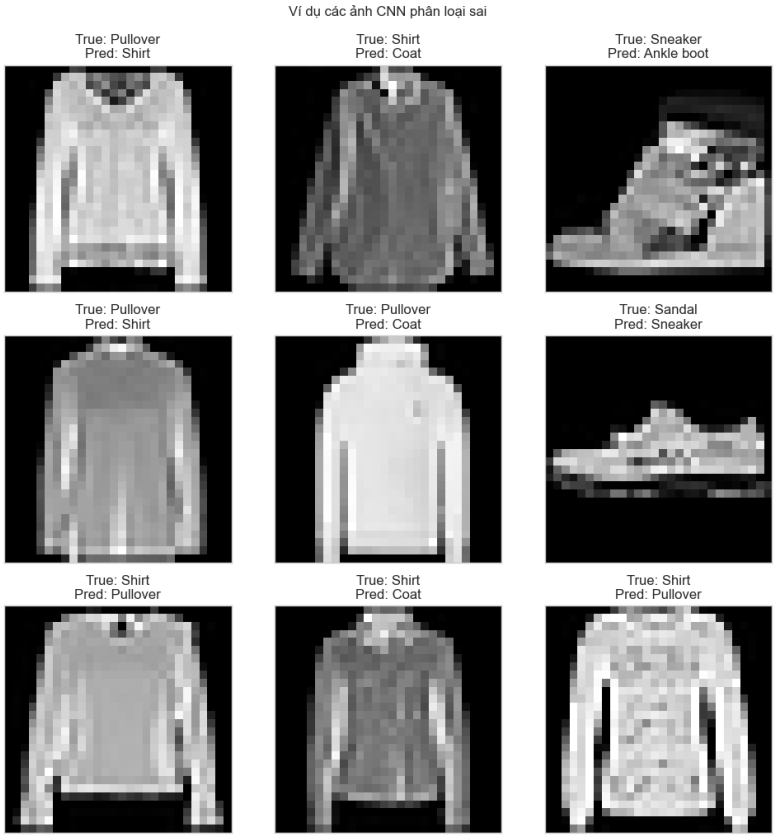
Hình 16. Lịch sử huấn luyện mô hình CNN (20 epochs).

Mô hình cuối cùng được đánh giá trên 10,000 ảnh kiểm tra. Độ chính xác đạt 92%. Ma trận nhầm lẫn và kết quả chi tiết theo lớp được thể hiện trong Hình 17.



Hình 17. Confusion Matrix - Mô hình CNN cuối cùng trên tập kiểm tra.

CNN cho kết quả tốt nhất trong số các mô hình được thử nghiệm, khẳng định ưu thế của mạng tích chập trong việc tự động học các đặc trưng phân cấp và phức tạp từ dữ liệu ảnh. Ma trận nhầm lẫn cho thấy hầu hết các lớp được phân loại chính xác, các lỗi sai sót chủ yếu tập trung vào các cặp lớp có hình dạng tương tự nhau (ví dụ: các loại áo, giày/boot). Hình 18 minh họa một số ví dụ mà mô hình CNN phân loại sai.



Hình 18. Một số ví dụ ảnh bị mô hình CNN phân loại sai.

Phân tích lỗi CNN: Để hiểu rõ hơn về các trường hợp phân loại sai, Hình 19 trực quan hóa sự phân bố của các mẫu dự đoán đúng (màu xanh) và sai (màu đỏ) trên tập kiểm tra trong không gian PCA 2 chiều.

Hình 19. Phân bố lỗi phân loại của CNN trên tập kiểm tra (trực quan hóa PCA 2D).

Quan sát Hình 19, các điểm bị phân loại sai (màu đỏ) có xu hướng nằm ở các vùng ranh giới giữa các cụm lớp hoặc bị lẫn vào các cụm của lớp khác. Điều này cho thấy mô hình gặp khó khăn chủ yếu ở các mẫu không rõ ràng hoặc nằm gần biên quyết định. Không có sự hình thành cụm lỗi rõ rệt ở một khu vực cụ thể nào, cho thấy lỗi phân bố tương đối ngẫu nhiên tại các vùng khó phân biệt.

4.4 Chuyển đổi Phân loại sang Hồi quy

Thử nghiệm được thực hiện để chuyển đổi bài toán phân loại sang hồi quy. Mục tiêu là dự đoán xác suất một mẫu thuộc về một lớp cụ thể, dựa trên đầu ra của mô hình phân loại tốt nhất (CNN). Lớp được chọn làm mục tiêu là Lớp 0 ('T-shirt/top'). Giá trị xác suất do tầng softmax của CNN dự đoán cho lớp này trên tập huấn luyện và kiểm tra được sử dụng làm biến mục tiêu y_{reg} cho các mô hình hồi quy.

Hai mô hình hồi quy được thử nghiệm: Linear Regression và Support Vector Regression với kernel RBF. Các mô hình này được huấn luyện trên:

- Dữ liệu gốc đã chuẩn hóa (784 chiều).
- Dữ liệu đã giảm chiều bằng PCA xuống còn $N_{reg} = \lfloor 784/3 \rfloor = 261$ chiều.

Hiệu năng được đánh giá bằng Sai số Bình phương Trung bình (Mean Squared Error - MSE) và Hệ số Xác định R-squared (R2). Kết quả được tổng hợp trong Bảng 2.

Bảng 2. Kết quả các mô hình hồi quy dự đoán xác suất lớp 'T-shirt/top'.

Mô hình	Dữ liệu	MSE	R2 Score
Linear Regression	Gốc (784 chiều)	0.020565	0.707406
Linear Regression	PCA (261 chiều)	0.020664	0.705996
SVR (RBF Kernel)	Gốc (784 chiều)	0.007984	0.886400
SVR (RBF Kernel)	PCA (261 chiều)	0.007364	0.895230

Nhận xét: Các giá trị R2 score thu được khá cao (ví dụ: R2 $>$ 0.7 cho SVR), cho thấy các mô hình hồi quy có khả năng học được mối quan hệ giữa đặc trưng ảnh và xác suất thuộc lớp 'T-shirt/top' do CNN dự đoán. Mô hình SVR với kernel RBF cho kết quả tốt hơn (MSE thấp hơn, R2 cao hơn) so với Hồi quy tuyến tính, phản ánh mối quan hệ có

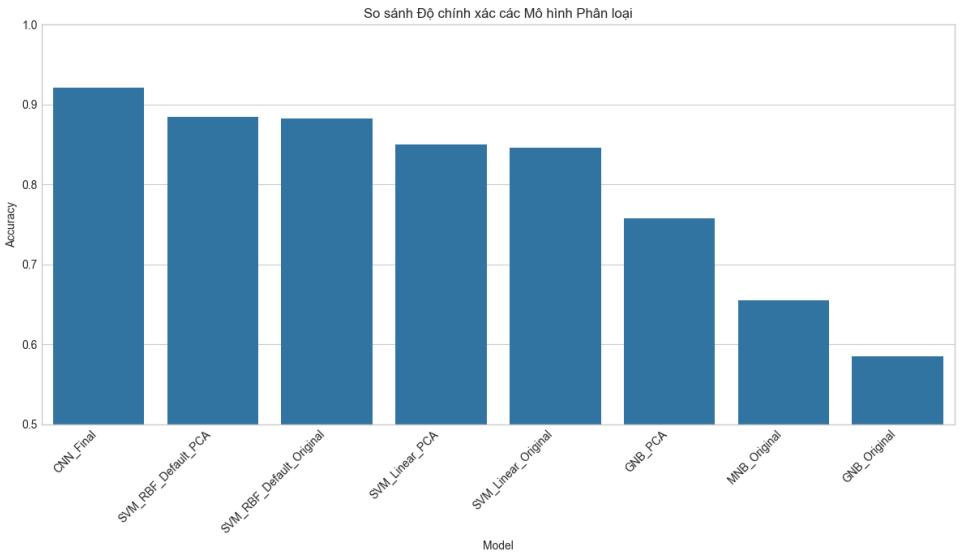
thể là phi tuyến. Việc giảm chiều xuống còn 261 thành phần PCA trong khi giảm đáng kể thời gian huấn luyện, đặc biệt là đối với SVR. Kết quả này cho thấy việc chuyển đổi sang bài toán hồi quy là khả thi và có thể cung cấp thêm thông tin về độ chắc chắn của dự đoán phân loại.

4.5 Tổng kết và So sánh Kết quả Phân loại

Bảng 3 và Hình 20 tổng hợp độ chính xác trên tập kiểm tra của các mô hình phân loại đã thử nghiệm.

Bảng 3. Tổng hợp độ chính xác trên tập kiểm tra của các mô hình phân loại.

Mô hình	Độ chính xác (%)
CNN (Final, 20 epochs)	92.12
SVM (RBF, Original, C=1.0, gamma=scale)	88.29
SVM (Linear, Original, C=1.0)	84.63
Gaussian Naive Bayes (PCA 100 comps)	75.82
Multinomial Naive Bayes (Raw Data)	65.54
Gaussian Naive Bayes (Original Data)	58.56



Hình 20. So sánh độ chính xác của các mô hình phân loại trên tập kiểm tra.

Kết quả thực nghiệm cho thấy:

- CNN là mô hình vượt trội nhất cho bài toán nhận dạng ảnh Fashion MNIST, đạt độ chính xác cao nhất.
- SVM, đặc biệt với kernel RBF, cũng là một lựa chọn mạnh mẽ, cho kết quả rất tốt và chỉ đứng sau CNN.
- Multinomial Naive Bayes cho kết quả khá tốt, phù hợp hơn Gaussian Naive Bayes.
- Gaussian Naive Bayes có hiệu năng thấp nhất, tuy nhiên được cải thiện đáng kể khi dùng PCA.
- Việc giảm chiều bằng PCA (100 chiều) giúp tăng tốc độ đáng kể cho SVM và GNB mà không làm giảm (hoặc chỉ giảm nhẹ) độ chính xác.
- Phân tích lỗi cho thấy các mô hình (như CNN) thường gặp khó khăn ở các mẫu nằm giữa ranh giới các lớp.
- Việc chuyển đổi bài toán sang hồi quy để dự đoán xác suất lớp là khả thi và các mô hình hồi quy như SVR có thể học được mối quan hệ này.

5 Kết Luận

Báo cáo này đã thực hiện việc áp dụng và đánh giá một loạt các thuật toán học máy và học sâu cho bài toán nhận dạng 10 loại trang phục trên tập dữ liệu Fashion MNIST. Quá trình thực nghiệm bao gồm tiền xử lý dữ liệu, giảm chiều và trực quan hóa, phân cụm không giám sát, và phân loại có giám sát, cùng với các phân tích chuyên sâu hơn về lỗi mô hình và khả năng chuyển đổi bài toán. Các kết quả chính thu được có thể tóm tắt như sau:

Về phân cụm (Unsupervised Learning):

- Thuật toán **K-Means** cho thấy khả năng nhóm các điểm dữ liệu có cấu trúc tương đối (ARI ≈ 0.35), nhưng phương pháp Elbow không cung cấp điểm tối ưu rõ ràng cho số cụm K. Hiệu năng phân cụm chỉ ở mức trung bình so với nhãn gốc.
- Thuật toán **DBSCAN**, với các tham số đã thử nghiệm, tỏ ra không phù hợp với cấu trúc mật độ của Fashion MNIST, dẫn đến việc phần lớn dữ liệu bị coi là nhiễu và các chỉ số đánh giá chất lượng cụm rất thấp.
- Việc **giảm chiều bằng PCA** trước khi phân cụm K-Means giúp tăng tốc độ tính toán nhưng không cải thiện đáng kể chất lượng cụm đo bằng các chỉ số như ARI hay NMI.

Về phân loại (Supervised Learning):

- **Mạng Nơron Tích Chập (CNN)** thể hiện hiệu năng vượt trội nhất, đạt độ chính xác cao nhất trên tập kiểm tra. Điều này khẳng định sức mạnh của CNN trong việc tự động trích xuất các đặc trưng phân cấp và phức tạp từ dữ liệu ảnh. Phân tích lỗi cho thấy các dự đoán sai thường xảy ra ở các vùng ranh giới giữa các lớp hoặc với các mẫu không điển hình.
- **Support Vector Machine (SVM)**, đặc biệt với **kernel RBF**, cũng cho kết quả rất tốt, chỉ đứng sau CNN. Kết quả này tốt hơn đáng kể so với kernel Linear, cho thấy dữ liệu có tính phi tuyến mà kernel RBF xử lý hiệu quả.
- Các biến thể **Naive Bayes** cho kết quả thấp hơn. Multinomial Naive Bayes phù hợp hơn Gaussian Naive Bayes. Hạn chế chính của Naive Bayes là giả định độc lập giữa các pixel, vốn không hoàn toàn đúng trong thực tế ảnh.

- Việc **giảm chiều bằng PCA (100 chiều)** giúp tăng tốc độ huấn luyện đáng kể cho SVM và Gaussian Naive Bayes mà không làm giảm (hoặc chỉ giảm nhẹ) độ chính xác cuối cùng.

Về chuyển đổi bài toán sang Hồi quy:

- Thử nghiệm cho thấy hoàn toàn khả thi khi sử dụng đầu ra xác suất của mô hình CNN làm mục tiêu cho bài toán hồi quy.
- Các mô hình hồi quy như **Linear Regression** và đặc biệt là **SVR (RBF)** có thể học được mối quan hệ giữa đặc trưng ảnh và xác suất thuộc lớp mục tiêu với mức độ thành công khá cao (dựa trên MSE và R2 score). Điều này mở ra hướng tiếp cận để ước lượng độ tin cậy của dự đoán phân loại.

Tóm lại, đối với bài toán nhận dạng ảnh tương đối phức tạp như Fashion MNIST, các mô hình học sâu như CNN thể hiện ưu thế rõ rệt nhờ khả năng học đặc trưng mạnh mẽ. Các phương pháp cổ điển mạnh như SVM với kernel phù hợp (RBF) cũng là một giải pháp thay thế rất hiệu quả. Các thuật toán phân cụm như K-Means có thể cung cấp một số hiểu biết ban đầu về cấu trúc dữ liệu nhưng khó đạt được độ chính xác cao như các phương pháp có giám sát. Naive Bayes, mặc dù đơn giản và nhanh, bị hạn chế bởi các giả định của nó và không phải là lựa chọn tối ưu cho loại dữ liệu này. Phân tích lỗi và chuyển đổi bài toán cung cấp thêm các góc nhìn giá trị về hoạt động và khả năng của mô hình.

Hướng phát triển tương lai: Để cải thiện hơn nữa kết quả, có thể xem xét các hướng sau:

- Thử nghiệm các kiến trúc CNN phức tạp hơn (ví dụ: ResNet, VGG) hoặc tinh chỉnh sâu hơn kiến trúc hiện tại.
- Áp dụng kỹ thuật tăng cường dữ liệu (data augmentation) để tăng sự đa dạng của tập huấn luyện và cải thiện khả năng tổng quát hóa của mô hình CNN.
- Tinh chỉnh sâu hơn các siêu tham số cho SVM (C, gamma) và CNN (tốc độ học, số lớp, kích thước bộ lọc,...) bằng các kỹ thuật như Grid Search hoặc Random Search.
- Sử dụng các phương pháp giảm chiều phi tuyến khác như UMAP để trực quan hóa dữ liệu, có thể cho thấy cấu trúc phân tách lớp tốt hơn t-SNE hoặc PCA trong một số trường hợp.

Tài liệu tham khảo

[1] Xiao, Han; Rasul, Kashif; Vollgraf, Roland (2017). *Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms*. arXiv:1708.07747 [cs.LG]. https://github.com/zalandoresearch/fashion-mnist

[2] Vũ Hữu Tiệp (2018). *Machine Learning cơ bản*. https://machinelearningcoban.com/

[3] Zhang, Aston; Lipton, Zachary C.; Li, Mu; Smola, Alexander J. (dịch giả: cộng đồng AI Việt Nam). *Đắm mình vào học sâu (Dive into Deep Learning - bản tiếng Việt)*. https://d2l.aivivn.com/