ĐẠI HỌC QUỐC GIA HÀ NỘI
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

MACHINE LEARNING

Báo cáo đề tài số 11

Phân loại chữ số viết tay

Hà Nội - 2024

ĐẠI HỌC QUỐC GIA HÀ NỘI
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA TOÁN - CƠ - TIN HỌC

21000703 Phan Văn Thành
21000695 Nguyễn Minh Ngọc
21000713 Phạm Ngọc Uy

BÁO CÁO ĐỀ TÀI

ĐỀ TÀI 11: PHÂN LOẠI CHỮ SỐ VIẾT TAY

Giảng viên: Cao Văn Chung

Hà Nội - 2024

LỜI CẢM ƠN

Lời đầu tiên của bài báo cáo, chúng em xin gửi lời cảm ơn sâu sắc tới giảng viên hướng dẫn TS. Cao Văn Chung. Thầy đã tận tình hướng dẫn các phương pháp, nhận xét góp ý để chúng em hoàn thành bài báo cáo này. Trong quá trình tìm hiểu và hoàn thiện báo cáo, chúng em có thể chưa trình bày được sâu sắc vấn đề cũng như không thể tránh khỏi những sai sót trong cách trình bày. Chúng em mong nhận được sự góp ý xây dựng của thầy để bài làm của chúng em được hoàn thiện hơn.

Một lần nữa chúng em xin chân thành cảm ơn!

Hà Nội, ngày 05 tháng 05 năm 2024

Sinh viên

Phan Văn Thành
Nguyễn Minh Ngọc
Phạm Ngọc Uy

Mục lục

1 Lý do chọn đề tài

Hiện nay, trong thời đại Công nghệ 4.0, khi trí tuệ nhân tạo đang trở thành xu thế phát triển, các bài toán trong thực tế được đặt ra ngày càng đa dạng và phức tạp. Trong số đó, các bài toán phân loại (hay phân lớp) luôn được quan tâm và chú trọng, đây cũng là một trong những lớp bài toán quan trọng nhất của Machine Learning. Một bài toán phân loại nổi tiếng thuộc lĩnh vực xử lý ảnh là nhận dạng chữ số viết tay, trong đó mỗi chữ số được gán một trong 10 nhãn tương ứng với các số từ 0 đến 9. Ở bài báo cáo này, chúng em sẽ xem xét một bộ dữ liệu ảnh các chữ số viết tay. Thực hiện giảm số chiều và trực quan hóa bộ dữ liệu, sau đó tiến hành một số phương pháp phân loại để huấn luyện mô hình, chạy kiểm thử mô hình và đánh giá mô hình thông qua một số độ đo phổ biến. Cụ thể, bài báo cáo của chúng em gồm 5 phần:

Phần 1: Trình bày khái quát về đề tài, vấn đề cần giải quyết, các mô hình đã sử dụng.

Phần 2: Giới thiệu về bộ dữ liệu, xây dựng và thực hiện phương pháp PCA để giảm số chiều dữ liệu, sau đó trực quan hóa dữ liệu trong không gian 2 chiều và 3 chiều.

Phần 3: Xây dựng và áp dụng mô hình Naive Bayes phù hợp để huấn luyện mô hình phân loại chữ số viết tay, sau đó chạy kiểm tra với dữ liệu trong tập validation và sử dụng một số metric để đánh giá mức độ hiệu quả của mô hình.

Phần 4: Xây dựng và áp dụng mô hình CNN để huấn luyện mô hình phân loại chữ số viết tay, sau đó chạy kiểm tra với dữ liệu trong tập validation và sử dụng một số metric để đánh giá mức độ hiệu quả của mô hình.

Phần 5: Tổng kết lại một số kết quả trong bài báo cáo.

Phần 6: Danh mục tài liệu tham khảo.

2 Mô tả và trực quan hóa dữ liệu

2.1 Mô tả về bộ dữ liệu

Tập dữ liệu MNIST có nguồn gốc từ tập NIST do tổ chức National Institute of Standards and Technology (NIST) cung cấp, đây là tập dữ liệu thường dùng để đánh giá hiệu quả của các mô hình nhận dạng chữ số viết tay. Dữ liệu được chúng em tham thảo từ nguồn http://yann.lecun.com/exdb/mnist/ gồm có 4 tệp tin nén:

train-images-idx3-ubyte: Chứa 60000 mẫu dữ liệu ảnh train là ảnh các chữ số viết tay.

train-labels-idx1-ubyte: Chứa 60000 mẫu dữ liệu nhãn ứng với tập ảnh train. Nhãn của từng ảnh là các số nguyên từ 0 đến 9 ứng với chữ số được ghi trong ảnh.

t10k-images-idx3-ubyte: Chứa 10000 mẫu dữ liệu ảnh test.

t10k-labels-idx1-ubyte: Chứa 10000 mẫu dữ liệu nhãn ứng với tập ảnh test.

Dữ liệu ảnh các chữ số viết tay ở đây được lưu liên tiếp nhau và không theo định dạng ảnh, cụ thể trong cấu trúc file như sau:

Cấu trúc file **train-images-idx3-ubyte** chứa dữ liệu ảnh training:

```
[offset] [type]          [value]          [description]
0000     32 bit integer  0x00000803(2051) magic number
0004     32 bit integer  60000            number of images
0008     32 bit integer  28               number of rows
0012     32 bit integer  28               number of columns
0016     unsigned byte   ??               pixel
0017     unsigned byte   ??               pixel
........
xxxx     unsigned byte   ??               pixel
```

Cường độ Pixels được sắp xếp cạnh nhau thành dòng. Giá trị cường độ Pixel là từ 0 đến 255 (1byte) nhưng để ngược: 0 là background (trắng),và 255 là foreground (đen).

Cấu trúc file **train-labels-idx1-ubyte** chứa nhãn của dữ liệu ảnh training:

```
[offset] [type]          [value]          [description]
0000     32 bit integer  0x00000801(2049) magic number (MSB first)
0004     32 bit integer  60000            number of items
0008     unsigned byte   ??               label
0009     unsigned byte   ??               label
........
xxxx     unsigned byte   ??               label

The labels values are 0 to 9.
```

Nhãn cho ảnh là số nguyên từ 0 đến 9 (ứng với chữ số trong ảnh).

Cấu trúc của tệp dữ liệu ảnh test test tương tự như với dữ liệu training nhưng với số lượng ảnh là 10000.

Cấu trúc file **t10k-images-idx3-ubyte** chứa dữ liệu ảnh test:

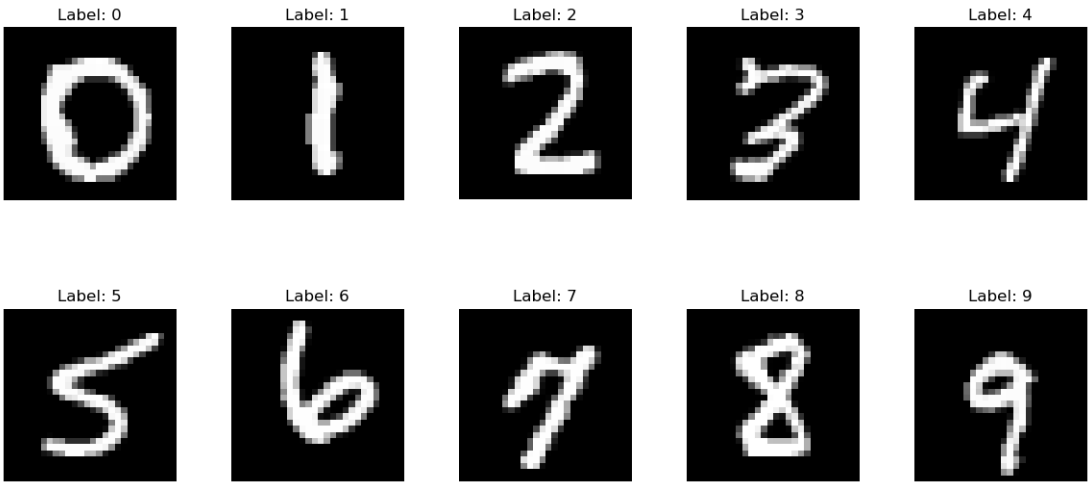
```
[offset] [type]          [value]          [description]
0000     32 bit integer  0x00000803(2051) magic number
0004     32 bit integer  10000            number of images
0008     32 bit integer  28               number of rows
0012     32 bit integer  28               number of columns
0016     unsigned byte   ??               pixel
0017     unsigned byte   ??               pixel
........
xxxx     unsigned byte   ??               pixel
```

Cấu trúc file **t10k-labels-idx1-ubyte** chứa nhãn của dữ liệu ảnh test:

```
[offset] [type]          [value]          [description]
0000     32 bit integer  0x00000801(2049) magic number (MSB first)
0004     32 bit integer  10000            number of items
0008     unsigned byte   ??               label
0009     unsigned byte   ??               label
........
xxxx     unsigned byte   ??               label
```

The labels values are 0 to 9.

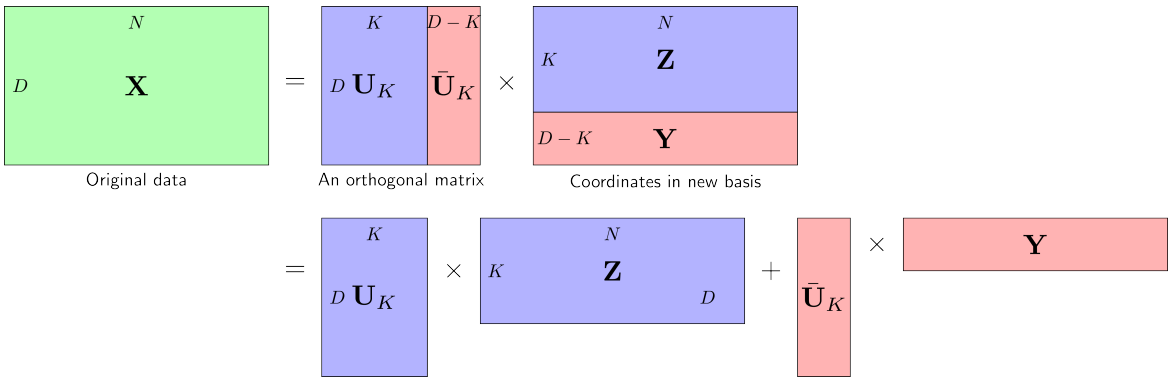
Sau đây, chúng ta sẽ cùng xem qua một số dữ liệu ảnh ứng với các nhãn từ 0 đến 9:



Hình 1: Một số dữ liệu ảnh ứng với 10 nhãn

2.2 Phân tích thành phần chính Principal Component Analysis

PCA là phương pháp đi tìm một hệ cơ sở mới sao cho thông tin của dữ liệu chủ yếu tập trung ở một số tọa độ. PCA sẽ tìm một **hệ trực chuẩn** để làm cơ sở mới.
Giả sử hệ cơ sở trực chuẩn mới là U và chúng ta muốn giữ lại K tọa độ trong hệ cơ sở này.



Hình 2:

Cơ sở mới $U = [U_K, \bar{U}_K]$ là một hệ trực chuẩn với U_K là ma trận con tạo bởi K cột đầu tiên của U. Khi đó ma trận dữ liệu được viết thành:

$$X = U_K Z + \bar{U}_K Y \tag{1}$$

Từ đây ta suy ra được:

$$\begin{bmatrix} Z \\ Y \end{bmatrix} = \begin{bmatrix} U_K^T \\ \bar{U}_K^T \end{bmatrix} X \rightarrowtail \begin{matrix} Z = U_K^T X \\ Y = \bar{U}_K^T X \end{matrix} \tag{2}$$

Ta sẽ cố gắng tìm Y sao cho Y xấp xỉ bởi một ma trận có toàn các cột như nhau. Gọi $b(bias)$ là mỗi cột của ma trận đó. Ta sẽ xấp xỉ:

$$Y \approx b1^T$$

Trong đó 1^T là vector hàng có toàn các phần tử bằng 1. Giả sử ta đã tìm được U rồi thì cần tìm b thỏa mãn:

$$b = \underset{b}{\operatorname{argmin}} ||Y - b1^T||_F^2 = \underset{b}{\operatorname{argmin}} ||\bar{U}_K^T X - b1^T||_F^2$$

Xét hàm $f(b) = \underset{b}{\operatorname{argmin}} ||\bar{U}_K^T X - b1^T||_F^2$.

Do hàm này là hàm lồi nên tìm cực tiểu toàn cục, ta đi tìm nghiệm của đạo hàm bậc nhất:

$$\nabla f(b) = 2(b1^T - \bar{U}_K^T X)1 = 0 \leftrightarrow b = \frac{\bar{U}_K^T X 1}{N} = \bar{U}_K^T \bar{x} \quad \text{với} \quad \bar{x} = \frac{X1}{N}$$

Với giá trị này của b dữ liệu ban đầu sẽ được xấp xỉ bởi:

$$X = U_K Z + \bar{U}_K Y \approx U_K Z + \bar{U}_K \bar{U}_K^T \bar{x} 1^T = \bar{X} \tag{3}$$

Từ (1) và (2) ta định nghĩa hàm mất mát như sau:

$$J = \frac{1}{N}||X - \bar{X}||_F^2 = \frac{1}{N}||\bar{U}_K Y - \bar{U}_K \bar{U}_K^T \bar{x} 1^T||_F^2 = \frac{1}{N}||\bar{U}_K \bar{U}_K^T X - \bar{U}_K \bar{U}_K^T \bar{x} 1^T||_F^2 = \frac{1}{N}||\bar{U}_K \bar{U}_K^T (X - \bar{x} 1^T)||_F^2 \tag{4}$$

Một chú ý quan trọng là nếu các cột của ma trận V tạo thành một hệ trực chuẩn thì với một ma trận W bất kỳ ta có:

$$||VW||_F^2 = ||W||_F^2 \tag{5}$$

Như ta đã nói, trong bài toán này $\bar{U}_K, U_K$ là một hệ trực chuẩn. Vì vậy áp dụng tính chất ta vừa nêu trên ta sẽ có điều này:

$$||\bar{U}_K \bar{U}_K^T||_F^2 = ||\bar{U}_K^T||_F^2 \tag{6}$$

Đặt $X' = X - \bar{x}1^T$ thay vào (4) ta có:

$$J = \frac{1}{N}||\bar{U}_K^T(X')||_F^2 \tag{7}$$

$$J = \frac{1}{N}||\bar{U}_K(X')^T||_F^2 \tag{8}$$

$$J = \frac{1}{N}\sum_{i=K+1}^{D}||X'^T u_i||_2^2 \tag{9}$$

$$J = \frac{1}{N}\sum_{i=K+1}^{D}X'^T X.u_i^T u_i \tag{10}$$

Mà $\frac{1}{N}X'^T X$ chính là ma trận hiệp phương sai S. Khi đó (10) trở thành:

$$J = \sum_{i=K+1}^{D}S.u_i^T u_i \tag{11}$$

Đến đây ta có thể thấy S là ma trận hiệp phương sai của dữ liệu và X' là dữ liệu sau khi chuẩn hóa với $x_i' = x_i - \bar{x}$ Mục tiêu tiếp theo là tìm các giá trị u_i sao cho hàm mất mát là nhỏ nhất.
Với S là ma trận hiệp phương sai nửa xác định dương, ta biến đổi (11) với $K = 0$ ta có:

$$L = \sum_{i=1}^{D}S.u_i^T u_i \tag{12}$$

$$L = \frac{1}{N}||X'^T U||_F^2 \tag{13}$$

$$L = trace\frac{1}{N}(X'^T X' U^T U) \tag{14}$$

Do U là một hệ trực chuẩn nên:

$$L = trace\frac{1}{N}(X'^T X' U^T U) = trace\frac{1}{N}(X'^T X') = trace(S) = \sum_{i=1}^{D}\lambda_i \tag{15}$$

Từ đây ta có thể thấy hàm L chính bằng tổng tất cả các giá trị riêng trên đường chéo chính của ma trận hiệp phương sai của dữ liệu ban đầu.
Vậy việc tối thiểu hàm mất mát có thể được viết như tối đa hàm sau:

$$F = L - J = \sum_{i=1}^{K}S.u_i^T u_i \tag{16}$$

Nghiệm của bài toán trên được phát biểu trong định lý sau:

F đạt giá trị lớn nhất bằng $\sum_{i=1}^{K} \lambda_i$ khi các vector riêng u_i ứng với giá trị riêng λ_i tạo thành một hệ trực chuẩn.

Vậy từ đây ta có các bước thực hiện PCA như sau:

1. Tính vector kỳ vọng của toàn bộ dữ liệu:

$$\bar{x} = \frac{1}{N} \sum_{1}^{N} x_n$$

2. Chuẩn hóa dữ liệu:

$$x'_n = x_n - \bar{x}$$

3. Tính ma trận hiệp phương sai:

$$S = \frac{1}{N} X'^T X'$$

4. Tính các giá trị riêng λ_i và các vector riêng u_i có $||u_i||_2 = 1$. Sắp xếp các giá trị riêng theo thứ tự giảm dần.
5. Chọn ra K giá trị riêng lớn nhất và K vector riêng tương ứng để tạo thành hệ cơ sở U_K
6. Chiếu dữ liệu đã được chuẩn hóa X' xuống không gian mới. Dữ liệu mới chính là tọa độ của dữ liệu trong không gian mới:

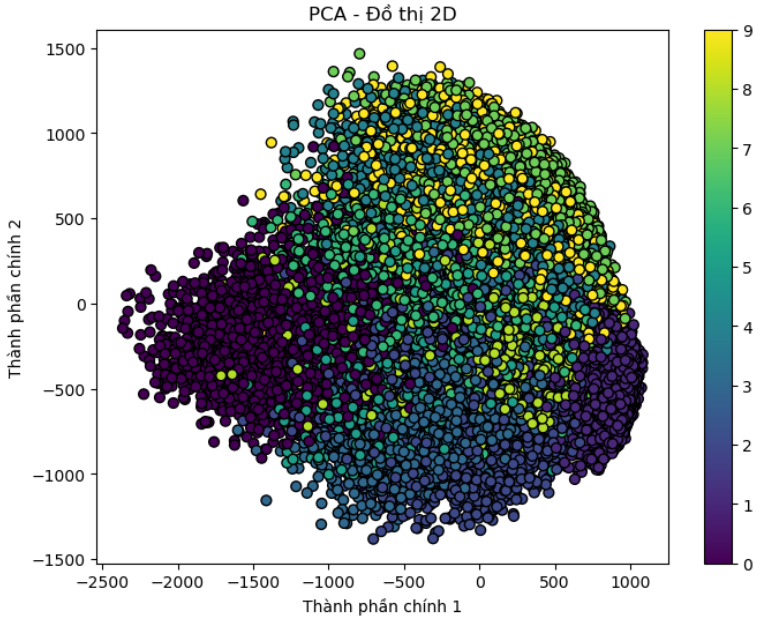
$$Z = U_K^T X'$$

$$x \approx U_K Z + \bar{x}$$

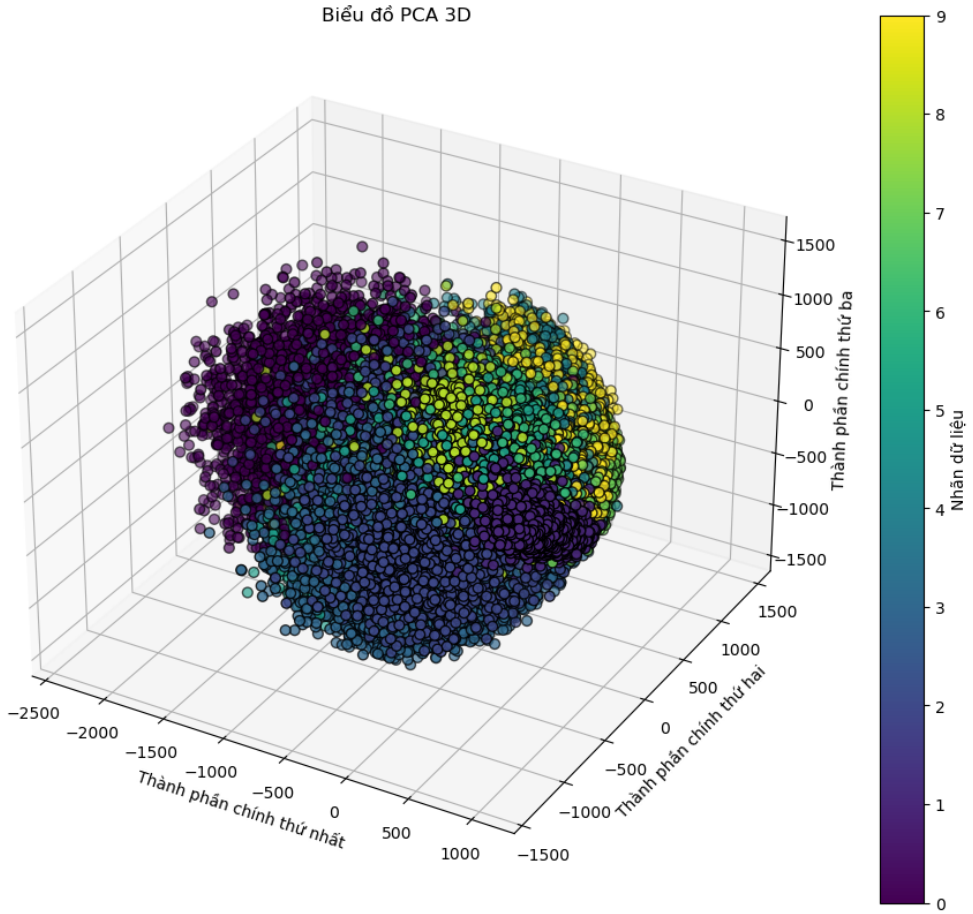
Như vậy ta đã xây dựng các bước thực hiện phương pháp phân tích thành phần chính PCA

2.3 Giảm số chiều và trực quan hóa dữ liệu

Mỗi điểm dữ liệu ứng với mỗi ảnh chữ số viết tay có số thành phần lên tới $28 \times 28 = 784$ nên ta cần giảm số chiều xuống 2 hoặc 3 để có thể trực quan hóa các điểm dữ liệu. Áp dụng phương pháp PCA đã được xây dựng ở phần (2.2) để giảm chiều dữ liệu, sau đó vẽ biểu đồ tán xạ của các điểm dữ liệu, ta thu được kết quả như sau:



Hình 3: Dữ liệu khi giảm số chiều về 2



Hình 4: Dữ liệu khi giảm số chiều về 3

3 Phân loại bằng mô hình Naive Bayes

3.1 Nội dung phương pháp

Xét bài toán phân loại với C lớp khác nhau: $\{1, 2, 3, .., C\}$. Với mỗi điểm dữ liệu $x \in \mathbb{R}^d$, thay vì tìm ra nhãn chính xác của x, ta có thể tính xác suất để x thuộc về lớp $c \in \{1, 2, 3, .., C\}$:

$$p(y = c|x) \text{ hoặc viết gọn thành } p(c|x)$$

Biểu thức $p(c|x)$ có nghĩa là xác suất để đầu ra là lớp c nếu đầu vào là điểm dữ liệu x. Từ đó nếu như ta tính được $p(c|x)$ thì ta có thể xác định được lớp c của điểm dữ liệu x bằng việc chọn ra lớp mà có xác suất rơi vào cao nhất:

$$c = \underset{c \in \{1,2,..,C\}}{argmax}\ p(c|x) \quad (1)$$

Trong thực tế, biểu thức $p(c|x)$ thường khó tính trực tiếp mà thay vào đó, chúng ta sẽ tính thông qua quy tắc Bayes:

$$p(c|x) = \frac{p(x|c)p(c)}{p(x)}$$

Do dữ liệu quan sát x không phụ thuộc vào lớp c nên do đó

$$p(c|x) \propto p(x|c)p(c)$$

Như vậy (1) trở thành bài toán xác định c sao cho:

$$c = \underset{c \in \{1,2,..,C\}}{argmax}\ p(x|c)p(c)$$

Đại lượng $p(c)$ có thể được hiểu là xác suất để một điểm bất kì rơi vào lớp c. Từ đó, $p(c)$ có thể được tính theo ước lượng hợp lý cực đại (Maximum Likelihood Estimation - MLE), tức là tỷ lệ số phần tử thuộc lớp c trên tổng số phần tử tập dữ liệu:

$$p(c) = \frac{|\{x \in X | label(x) = c\}|}{|X|}$$

Đại lượng còn lại $p(x|c)$ là phân phối xác suất của dữ liệu bên trong lớp c, thường rất khó tính toán vì x là một biến ngẫu nhiên nhiều chiều. Để giải quyết vấn đề này, người ta giả thiết các thành phần của x là độc lập với nhau, với c đã cho. Tức là:

$$p(x|c) = p(x_1, x_2, ..., x_d|c) = \prod_{i=1}^{d} p(x_i|c)$$

Thực tế giả thiết về sự độc lập của các chiều dữ liệu là quá chặt và ít khi có được với dữ liệu thực. Mặc dù vậy việc áp dụng nó lại cho những kết quả ngoài mong đợi. Giả thiết này được gọi là Naive Bayes, và phương pháp phân loại dựa trên giả thiết này có

tên là Naive Bayes Classifier. Do giả thiết Naive Bayes dẫn đến tính toán đơn giản hơn rất nhiều, nên phương pháp này rất phù hợp với dữ liệu lớn - large scale.

Ở bước training, các phân phối $p(c)$ và $p(x_i|c), i = 1, ..., d$ được xác định dựa vào training data.

Ở bước test, với một điểm dữ liệu mới x sẽ được phân vào lớp c nếu như:

$$c = \underset{c\in\{1,2,..,C\}}{argmax}\ p(c)\prod_{i=1}^{d} p(x_i|c)$$

Khi d lớn và các xác suất nhỏ, biểu thức $\prod_{i=1}^{d} p(x_i|c)$ là một số rất nhỏ, khi tính toán có thể gặp sai số. Để giải quyết điều này, ta lấy log của hàm mục tiêu và khi đó, bài toán cực trị trở thành:

$$c = \underset{c\in\{1,2,..,C\}}{argmax}\ (log(p(c) + \sum_{i=1}^{d} log(p(x_i|c)))$$

Do hàm log là hàm đồng biến nên kết quả không đổi. Việc giả sử các thành phần trong dữ liệu là độc lập với nhau giúp cho việc tính toán các phân phối $p(x_i|c)$ không mất nhiều thời gian. Việc tính toán các phân phối này phụ thuộc vào dữ liệu. Có ba loại phân phối thường dùng là Gaussian Naive Bayes, Multinomial Naive Bayes và Bernoulli Naive Bayes.

i. Gaussian Naive Bayes

Mô hình này được sử dụng chủ yếu trong loại dữ liệu mà các thành phần là các biến liên tục. Giả thiết, tại chiều dữ liệu thứ $i(i = 1, ..., d)$ và phân lớp c, dữ liệu x_i tuân theo phân bố chuẩn có kỳ vọng μ_{c_i} và phương sai $\sigma^2_{c_i}$, khi đó:

$$p(x_i|c) = p(x_i|\mu_{c_i}, \sigma^2_{c_i}) = \frac{1}{\sqrt{2\pi\sigma^2_{c_i}}} exp\left(-\frac{(x_i - \mu_{c_i})}{2\sigma^2_{c_i}}\right)$$

Trong đó bộ tham số $\theta = \{\mu_{c_i}, \sigma^2_{c_i}\}$ được xác định bằng MLE dựa trên các điểm dữ liệu Training thuộc lớp c.

ii. Multinomial Naive Bayes

Mô hình này chủ yếu được sử dụng trong phân loại văn bản mà vector đặc trưng được xây dựng dựa trên kỹ thuật Bags of Words. Trong kỹ thuật này, mỗi văn bản được biểu diễn bởi một vector có độ dài d là số từ trong từ điển. Giá trị của thành phần (tọa độ) thứ i của mỗi vector chính là số lần từ thứ i (trong từ điển) xuất hiện trong văn bản. Giá trị này có thể được tính bằng:

$$\lambda_{c_i} = p(x_i|c) = \frac{N_{c_i}}{N_c}$$

trong đó:

- N_{c_i} là là tổng số lần từ thứ i xuất hiện trong các văn bản thuộc lớp c. Nó được tính là tổng của tất cả các thành phần thứ i của các điểm dữ liệu trong phân lớp c.
- N_c là tổng số từ (kể cả lặp) xuất hiện trong phân lớp c. Tức là N_c bằng tổng độ dài tính theo từ của toàn bộ các văn bản thuộc vào lớp c. Dễ thấy rằng $N_c = \sum_{i=1}^{d} N_{c_i}$ và do đó $\sum_{i=1}^{d} \lambda_{c_i} = 1$.

  Nhược điểm của cách tính này là nếu như một từ không xuất hiện trong lớp c thì $\lambda_{c_i} = p(x_i|c) = 0$ và do đó $p(x|c) = 0$ cho dù các từ khác có tần suất xuất hiện lớn. Để tránh nhược điểm này, một kỹ thuật được gọi là Laplace smoothing được áp dụng:

$$\hat{\lambda_{c_i}} = \frac{N_{c_i} + \alpha}{N_c + d\alpha}$$

  Với α là một số dương, thường bằng 1, để tránh trường hợp tử số bằng 0. Mẫu số được cộng với $d\alpha$ nên có thể đảm bảo tổng xác suất $\sum_{i=1}^{d} \hat{\lambda_{c_i}} = 1$. Như vậy, mỗi lớp c sẽ được mô tả bằng một bộ d số dương có tổng bằng 1: $\lambda_c = \{\hat{\lambda_{c_1}}, \hat{\lambda_{c_2}}, ..., \hat{\lambda_{c_d}}\}$.

iii. Bernoulli Naive Bayes

Mô hình này được áp dụng cho các loại dữ liệu mà mỗi thành phần là một giá trị binary - bằng 0 hoặc 1. Trong trường hợp này, công thức tính các $p(x_i|c)$ như sau:

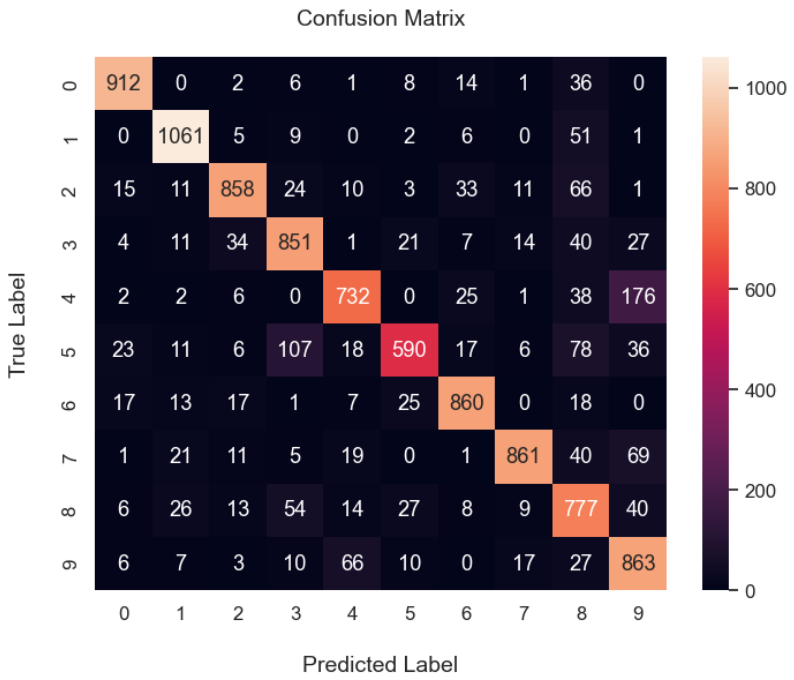
$$p(x_i|c) = p(i|c)^{x_i}(1 - p(i|c))^{1-x_i}$$

Với đại lượng $p(i|c)$ có thể hiểu là xác suất từ thứ i xuất hiện trong các văn bản của lớp c.

Như vậy chúng ta đã xây dựng được các mô hình phân loại Naive Bayes. Sau đây, chúng ta sẽ áp dụng mô hình Naive Bayes phù hợp để phân loại ảnh chữ viết tay.

3.2 Phân loại ảnh chữ số viết tay bằng mô hình Naive Bayes

Do dữ liệu chữ viết tay Training được cho ở dạng các vector với mỗi thành phần là các giá trị của pixel tương ứng nên dữ liệu đầu vào là dữ liệu rời rạc. Với đầu ra là 10 lớp, ta sử dụng mô hình Multinomial Naive Bayes để giải quyết bài toán phân loại chữ số viết tay. Huấn luyện mô hình bằng phương pháp Multinomial Naive Bayes đã được xây dựng ở mục 3.1 và chạy kiểm tra với tập test, chúng ta thu được kết quả như sau:



Hình 5: Confusion Matrix

```
Accuracy (Naive Bayes): 0.8365
Precision (Naive Bayes): 0.8433162997126132
Recall (Naive Bayes): 0.8334531845906966
              precision    recall  f1-score   support

           0       0.92      0.93      0.93       980
           1       0.91      0.93      0.92      1135
           2       0.90      0.83      0.86      1032
           3       0.80      0.84      0.82      1010
           4       0.84      0.75      0.79       982
           5       0.86      0.66      0.75       892
           6       0.89      0.90      0.89       958
           7       0.94      0.84      0.88      1028
           8       0.66      0.80      0.72       974
           9       0.71      0.86      0.78      1009

    accuracy                           0.84     10000
   macro avg       0.84      0.83      0.84     10000
weighted avg       0.84      0.84      0.84     10000
```

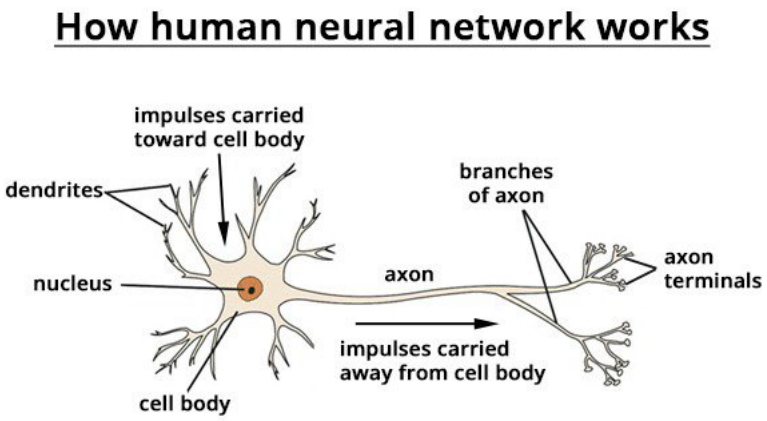
Hình 6: Các metrics đánh giá mô hình

Độ chính xác cũng như Precision và Recall của mô hình đạt được khá cao với $83,65\%; 84,33\%; 83,35\%$ tương ứng.

4 Phân loại bằng mô hình CNN

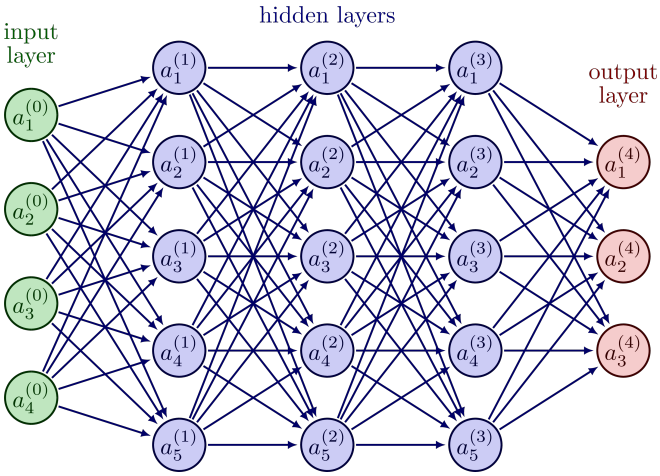
4.1 Mô hình Neural Network

Neural Network là một hệ thống tính toán được xây dựng dựa trên sự hoạt động của các Nơ-ron trong hệ thần kinh của con người. Trong đó, mỗi Nơ-ron nhận dữ liệu đầu vào từ các sợi nhánh (dendrities) và truyền dữ liệu đầu ra qua sợi trục (axon) đến các sợi nhánh của các Nơ-ron khác, qua đó các Nơ-ron được liên kết với nhau.



Hình 7: Cách thức hoạt động của mạng Nơ-ron trong hệ thần kinh của con người

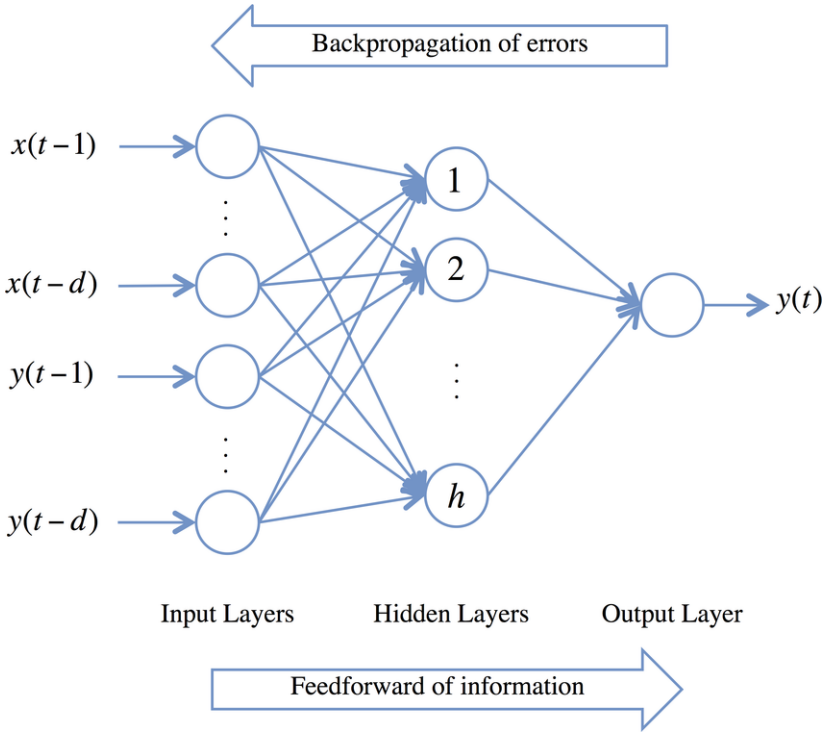
Nhờ việc mô phỏng hoạt động của các Nơ-ron trong hệ thần kinh con người, các mô hình Neural Network có thể thích ứng với các điều chỉnh từ dữ liệu đầu vào, khai thác các mối quan hệ cơ bản trong tập dữ liệu và đưa ra kết quả dự báo rất chính xác trên tất cả các tập dữ liệu lớn. Do đó, các mô hình Neural Network được ứng dụng rộng rãi trong nhiều lĩnh vực trong cuộc sống hiện nay. Cấu trúc cơ bản của một mô hình Neural Network có thể được mô tả như sau:



Hình 8: Cấu trúc của mô hình Neural Network

Layer đầu tiên được gọi là Input layer, các layer ở giữa được gọi là Hidden layer và layer cuối cùng được gọi là Output layer. Trong các layer có chứa các Node và mỗi node trong Hidden layer và Output layer được liên kết với các node ở layer trước đó qua các hệ số w (weight) và mỗi node đều có hệ số b (bias) riêng. Ở mỗi node diễn ra 2 quá trình là tính tổng tuyến tính từ các node được liên kết ở layer trước đó thông qua hệ số b và các trọng số w, sau đó qua hàm kích hoạt (activation function) để xác định giá trị tại node đó.

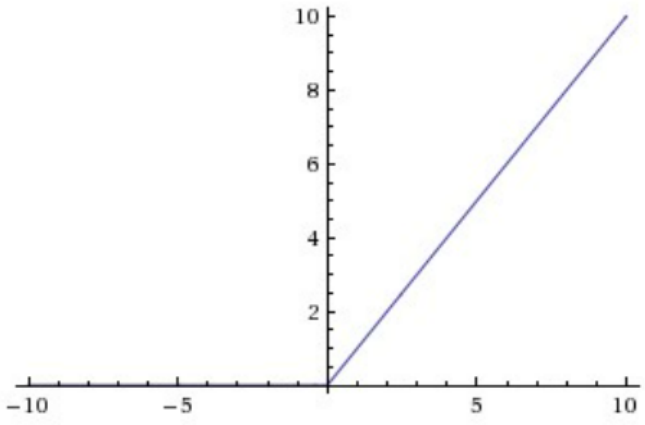
Quá trình tính các node ở các layer theo chiều tiến, đi từ Input layer tới Output layer được gọi là quá trình Feedforward. Quá trình tìm các trọng số w và hệ số b ở các layer được thực hiện nhờ thuật toán tối ưu Gradient descent và thuật toán Backpropagation (Lan truyền ngược).



Hình 9: Quá trình Feedforward và Backpropagation

4.2 Các hàm kích hoạt được sử dụng trong mô hình

4.2.1 Hàm ReLu



Hình 10: Đồ thị hàm ReLu

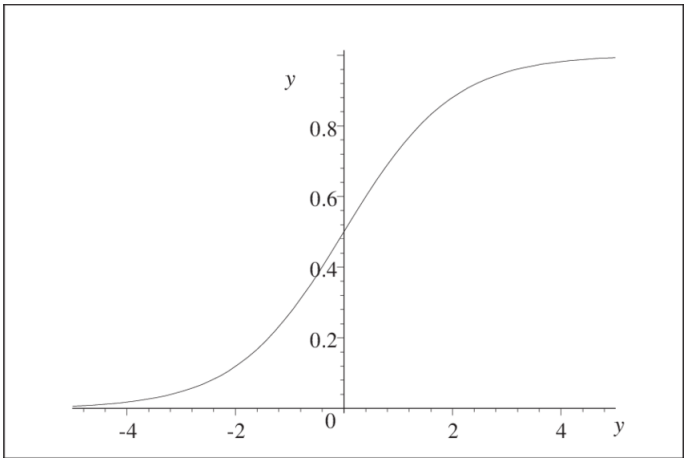
ReLU (Rectified Linear Unit) được sử dụng rộng rãi gần đây vì tính đơn giản của nó. Đồ thị của hàm ReLU được minh họa trên Hình 10.

Công thức toán học:

$$f(z) = \max(0, z)$$

ReLU được chứng minh giúp cho việc training các Deep Networks nhanh hơn rất nhiều, lý do bởi vì ReLU được tính toán gần như tức thời và gradient của nó cũng được tính cực nhanh với gradient bằng 1 nếu đầu vào lớn hơn 0, bằng 0 nếu đầu vào nhỏ hơn 0.

4.2.2 Hàm Softmax



Hình 11: Đồ thị hàm Softmax

Chúng ta cần một mô hình xác suất sao cho với mỗi input $\mathbf{x}$, a_i thể hiện xác suất để input đó rơi vào class i. Vậy điều kiện cần là các a_i phải dương và tổng của chúng bằng 1. Để có thể thỏa mãn điều kiện này, chúng ta cần nhìn vào mọi giá trị z_i và dựa trên quan hệ giữa các z_i này để tính toán giá trị của a_i. Ngoài các điều kiện a_i lớn hơn 0 và có tổng bằng 1, chúng ta sẽ thêm một điều kiện cũng rất tự nhiên nữa, đó là: giá trị $z_i = \mathbf{w}_i^T \mathbf{x}$ càng lớn thì xác suất dữ liệu rơi vào class i càng cao. Điều kiện cuối này chỉ ra rằng chúng ta cần một hàm đồng biến ở đây.

Chú ý rằng z_i có thể nhận giá trị cả âm và dương. Một hàm số mượt đơn giản có thể chắc chắn biến z_i thành một giá trị dương, và hơn nữa, đồng biến, là hàm $\exp(z_i) = e^{z_i}$ Điều kiện mượt để thuận lợi hơn trong việc tính đạo hàm sau này. Điều kiện cuối cùng, tổng các a_i bằng 1 có thể được đảm bảo nếu:

$$a_i = \frac{\exp(z_i)}{\sum_{j=1}^{C} \exp(z_j)}, \quad \forall i = 1, 2, \ldots, C$$

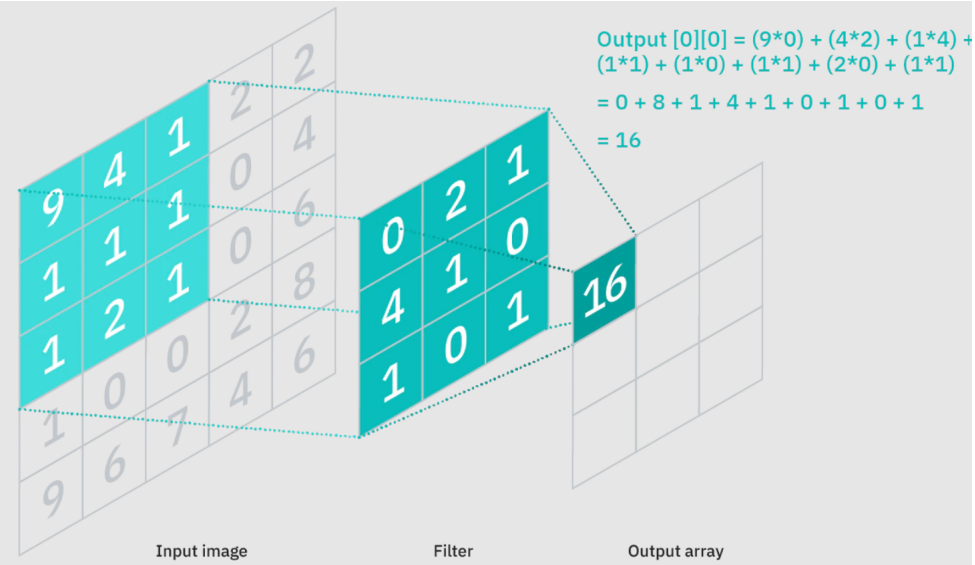
Hàm số này được gọi là softmax function.

4.3 Phép tính Convolution

Convolution (Tích chập) là một kỹ thuật quan trọng trong xử lý ảnh, giúp biến đổi ma trận dữ liệu ảnh đầu vào để làm rõ và tách ra các đặc tính của hình ảnh mà vẫn bảo toàn tính tương quan giữa dữ liệu ảnh đầu vào và đầu ra.

Phép tính Convolution sử dụng một ma trận vuông cỡ k với k là một số lẻ gọi là Kernel hay Filter. Với mỗi phần tử x_{ij} trong ma trận ảnh ban đầu X, ta xác định một ma trận vuông X_{ij} cỡ k nhận x_{ij} là phần tử trung tâm (giao của đường chéo chính và đường chéo phụ) và tính tổng các phần tử của ma trận thu được nhờ thực hiện phép tính Element-wise giữa ma trận X_{ij} với Kernel. Kết quả thu được là phần tử $y_{(i-\frac{k-1}{2})(j-\frac{k-1}{2})}$

của ma trận Y. Ma trận Y thu được sau phép tính Convolution với ma trận X vuông cỡ n và Kernel cỡ k là một ma trận vuông cỡ $n - k + 1$.



Hình 12: Phép tính Convolution

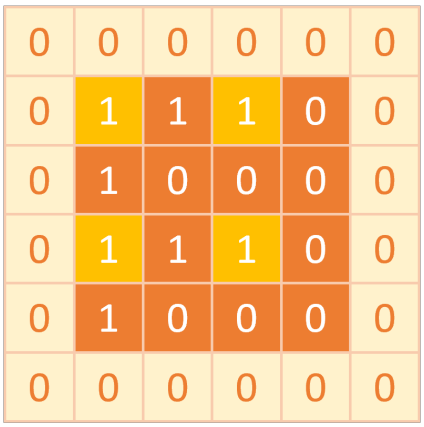
Có thể thấy sau mỗi phép tính Convolution, ma trận Y thu được có kích thước nhỏ hơn ma trận X ban đầu. Để ma trận Y thu được có kích thước bằng ma trận X ban đầu, ta thêm $\frac{k-1}{2}$ vector $\mathbf{0}$ vào mỗi phía ngoài cùng của ma trận X. Khi đó, ma trận $\hat{X}$ mới sẽ là một ma trận vuông cỡ $n+k-1$ và ma trận Y thu được nhờ phép tính Convolution với ma trận X và Kernel cỡ k sẽ có kích thước là $n+k-1-k+1 = n$. Việc thêm m vector $\mathbf{0}$ vào mỗi phía của ma trận X trong phép tính Convolution được đặc trưng bởi tham số **padding** = m.

0	0	0	0	0	0
0	1	1	1	0	0
0	1	0	0	0	0
0	1	1	1	0	0
0	1	0	0	0	0
0	0	0	0	0	0

Hình 13: Phép tính Convolution với padding = 1

Phép tính Convotional còn được đặc trưng bởi tham số **strike**. Trong các phép tính

Convolution được đề cập ở trên, tham số $strike = 1$, tức là ta sẽ xét các phần tử x_{ij} lần lượt liền kề nhau để xác định ma trận X_{ij} và thực hiện phép tính Element-wise. Khi $strike = m$, ta chỉ xét các phần tử ma trận nhận phần tử $x_{(\frac{k-1}{2}+i\times m)(\frac{k-1}{2}+j\times m)}$ làm phần tử trung tâm.



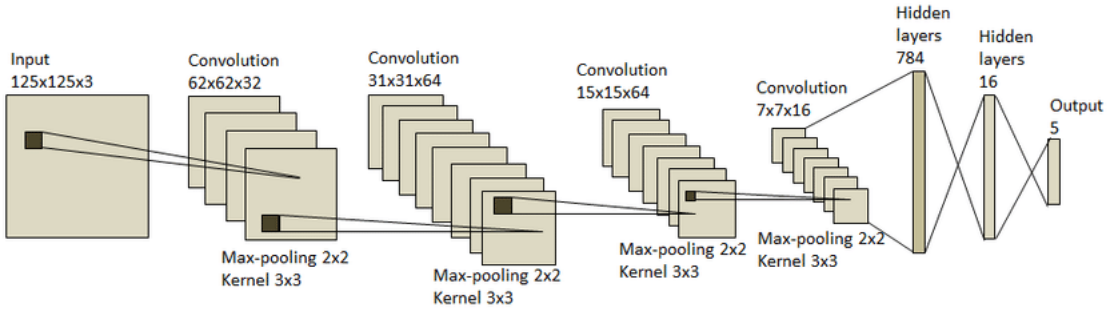
Hình 14: Phép tính Convolution với padding = 1 và strike = 2

4.4 Mạng Nơ-ron tích chập - CNN

Đối với mô hình Neural Network đã được trình bày ở mục 4.1, các Hidden layer ở giữa Input layer và Output layer được gọi là Fully Connected Layer, nghĩa là các node ở các layer trong Hidden layer được kết nối với tất cả các node ở layer trước đó thông qua các trọng số và bias. Đối với dữ liệu là một ảnh, số lượng node cần truyền vào Input layer là rất lớn. Ví dụ đối với ảnh màu 28x28 thì tensor biểu diễn ảnh này có số chiều là 28x28x3. Để biểu thị hết thông tin về ảnh thì số node truyền vào Input layer là 28x28x3 nodes.

Để kết nối các node ở hidden layer với tất cả các node ở input layer thì số lượng trọng số và bias là rất lớn, chưa kể là cần kết nối các node giữa các hidden layer. Từ quan sát thấy rằng các pixel ở gần nhau thường có liên kết với nhau hơn là những pixel ở xa. Mô hình CNN cải tiến mô hình Neural Network thông thường bằng việc áp dụng phép tính convolution vào layer để giải quyết vấn đề về số lượng tham số mà vẫn giữ được được các đặc trưng của ảnh.

Một cách tổng quát, mô hình CNN sẽ gồm một số Convolutional layer là các layer sử dụng phép tính Convolution để rút trích các đặc trưng của ảnh. Các Pooling layer được đặt ở giữa các Convolutional layer để giảm kích thước dữ liệu ảnh nhưng vẫn giữ được các thuộc tính quan trọng. Sau khi trải qua các tầng Convolutional và Pooling, dữ liệu ảnh sẽ không còn quá lớn. Khi đó, dữ liệu ở dạng tensor sẽ đi qua Flatten layer để được làm phẳng trở thành dữ liệu dạng vector. Cuối cùng, dữ liệu ở dạng vector được đi qua các tầng fully connected nhằm kết hợp các đặc trưng của ảnh để đưa ra kết luận ở Output layer.



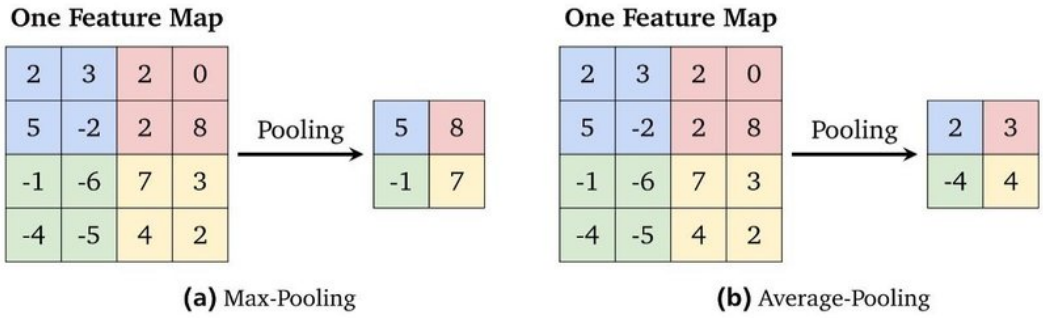
Hình 15: Mô hình CNN

Convolutional layer

Ở layer này, tensor ảnh sẽ được các bộ lọc Filter (Kernel) quét qua để thực hiện phép tính Convolution. Với mỗi bộ lọc Filter khác nhau ta sẽ học được những đặc trưng khác nhau của ảnh, nên trong mỗi Convolutional layer ta sẽ dùng nhiều Filter để học được nhiều đặc trưng của ảnh. Hàm kích hoạt được sử dụng là hàm ReLu để chuyển các giá trị âm trong phép tính Convolution về giá trị 0.

Pooling layer

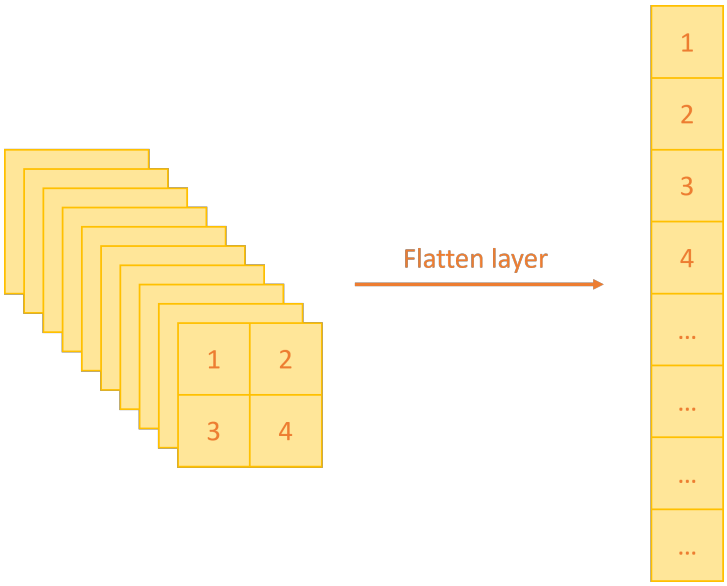
Layer này được đặt ở giữa các Convolutional layer nhằm giảm kích thước dữ liệu ảnh bằng cách sử dụng một ma trận $k \times k$ quét qua toàn bộ dữ liệu ảnh. Khác với phép tính Convolution, khi ma trận của Pooling layer quét qua các vùng ảnh, chỉ có một giá trị đại diện cho vùng ảnh đó được giữ lại, giá trị này sẽ thể hiện cho đặc trưng của vùng ảnh được quét. Hai loại Pooling layer phổ biến là Max-Pooling (Lấy giá trị max trong vùng ảnh được quét) và Average-Pooling (Lấy giá trị trung bình của tất cả điểm dữ liệu trong vùng ảnh được quét).



Hình 16: Pooling layer

Flatten layer

Sau khi đi qua các Convolution layer và Pooling layer, dữ liệu ảnh đã được rút trích các đặc trưng quan trọng và khối lượng dữ liệu không còn lớn như dữ liệu ban đầu, mô hình sẽ sử dụng Flatten layer để dàn phẳng tensor thành 1 vector nhằm chuẩn bị dữ liệu đầu vào cho các tầng Fully connected.



Hình 17: Flatten layer

Fully connected layer

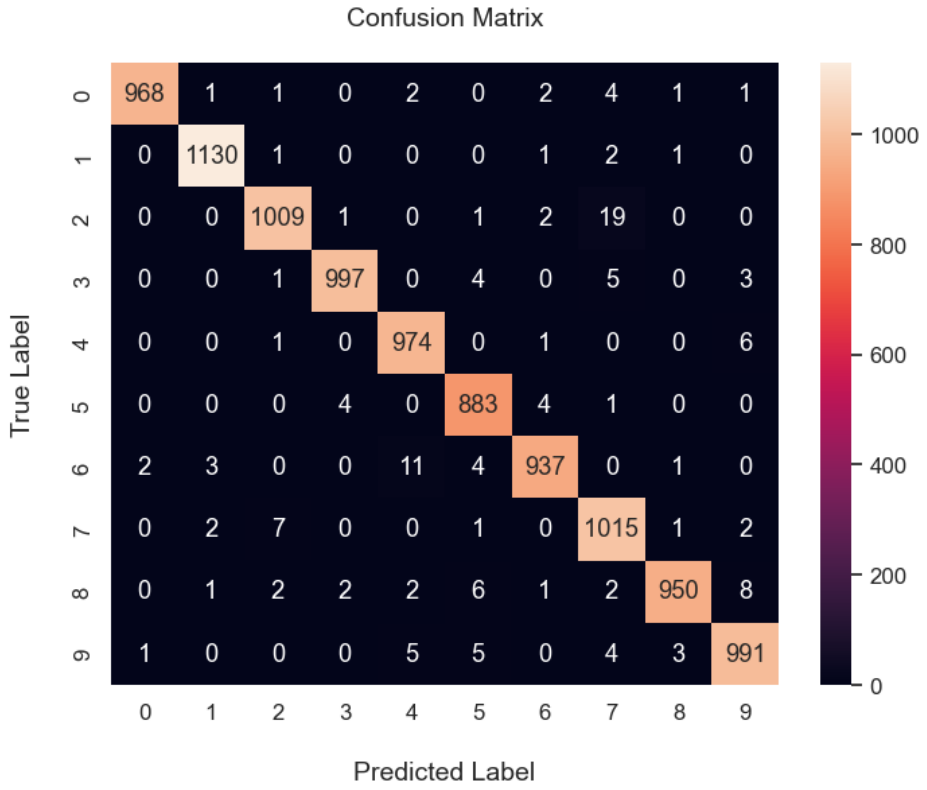
Layer này tương tự với Hidden layer trong mô hình Neural Network. Khi đã có dữ liệu dạng vector từ Flatten layer, các Fully connected layer được sử dụng để kết hợp các đặc trưng của ảnh và đưa ra kết luận ở Output layer.

4.5 Phân loại ảnh chữ số viết tay bằng mô hình CNN

Chúng em sẽ sử dụng mô hình CNN từ thư viện TensorFlow, với kiến trúc như sau:

- 3 tầng tích chập (Convolutional layer) kết hợp với Pooling (Max-Pooling layers), Kernel của các Convolutional layer có kích thước là 3×3, kích thước Pooling là 2×2. Sau mỗi tầng tích chập, hàm kích hoạt được sử dụng là ReLU.
- Tầng tiếp theo là Flatten layer dùng để duỗi các tensor thành vector.
- Sau Flatten layer là 2 tầng Fully connected với số Units lần lượt là 256 và 128. Hàm kích hoạt được sử dụng là ReLU.
- Cuối cùng là một tầng phân loại ở đầu ra, sử dụng hàm Softmax vì cần phân loại 10 lớp.

Huấn luyện mô hình và chạy kiểm tra với tập Test, chúng em thu được kết quả như sau:



Hình 18: Confusion Matrix

```
Accuracy (CNN): 0.9854
Precision (CNN): 0.9853839371951321
Recall (CNN): 0.9852908696718632
              precision    recall  f1-score   support

           0       1.00      0.99      0.99       980
           1       0.99      1.00      0.99      1135
           2       0.99      0.98      0.98      1032
           3       0.99      0.99      0.99      1010
           4       0.98      0.99      0.99       982
           5       0.98      0.99      0.98       892
           6       0.99      0.98      0.98       958
           7       0.96      0.99      0.98      1028
           8       0.99      0.98      0.98       974
           9       0.98      0.98      0.98      1009

    accuracy                           0.99     10000
   macro avg       0.99      0.99      0.99     10000
weighted avg       0.99      0.99      0.99     10000
```

Hình 19: Các metrics đánh giá mô hình

Độ chính xác cũng như Precision và Recall của mô hình đạt được rất cao với tương ứng 98,54%; 98,54% và 98,53%. So với mô hình Naive Bayes, độ chính xác đã được cải thiện hơn rất nhiều.

5 Tổng kết

Như vậy, qua bài báo cáo giữa kỳ này, chúng em xin tổng kết một số kết quả như sau:

- Mô tả về bộ dữ liệu ảnh chữ số viết tay. Xây dựng được phương pháp PCA và áp dụng để giảm số chiều dữ liệu sau đó trực quan hóa bộ dữ liệu.
- Xây dựng được các mô hình phân loại Naive Bayes, sau đó áp dụng mô hình phân loại Naive Bayes phù hợp với bộ dữ liệu để huấn luyện mô hình. Chạy kiểm thử mô hình trên tập dữ liệu Test sau đó tính toán một số metric đánh giá mức độ hiệu quả của mô hình. Có thể nói mô hình phân loại Multinomial Naive Bayes hoạt động khá tốt với giá trị của 3 metric đánh giá đều đạt trên 80%.
- Xây dựng được mô hình CNN và áp dụng để phân loại ảnh chữ số viết tay. So với mô hình phân loại Naive Bayes, mô hình CNN đạt được độ chính xác cao hơn rất nhiều với giá trị của 3 metric đánh giá đều đạt trên 98%. Kết quả này cho thấy tính ưu việt của mô hình CNN trong bài toán phân loại nói chung và bài toán xử lý ảnh nói riêng.

6 Tài liệu tham khảo

[1] Vũ Hữu Tiệp, *Machine Learning cơ bản*, https://machinelearningcoban.com/ebook/

[2] Nguyễn Thanh Tuấn, *Deep Learning cơ bản*, https://nttuan8.com/sach-deep-learning-co-ban/

[3] https://machinelearningcoban.com/2017/06/15/pca/

[4] http://yann.lecun.com/exdb/mnist/

[5] https://www.researchgate.net/figure/Feedforward-Backpropagation-Neural-Network-architecture_fig3_303744090/

[6] https://www.researchgate.net/figure/The-proposed-CNN-model-architecture_fig2_336805909

[7] https://www.researchgate.net/figure/Example-for-the-max-pooling-and-the-average-pooling-with-a-filter-size-of-22-and-a_fig15_337336341