ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

BÁO CÁO ĐỒ ÁN CUỐI KỲ

MÔN: MÁY HỌC

<CS114.P21>

Giáo viên hướng dẫn:

- GV. Phạm Nguyễn Trường An

Sinh viên thực hiện:

- Nhữ Đình Tiến – 23521582
- Nguyễn Công Thiết – 23521491
- Hồ Nhật Thành - 23521439

Source code: https://github.com/nhudinhtien-4w/CS114_Final-Projects.git

TP. HỒ CHÍ MINH, THÁNG 6 NĂM 2025

MỤC LỤC

DANH MỤC HÌNH

DANH MỤC BẢNG

PHẦN A: ĐỒ ÁN NHẬN DIỆN CHỮ SỐ VIẾT TAY

Chương 0: CÁC THAY ĐỔI SO VỚI KHI VẤN ĐÁP

Các câu hỏi trong quá trình vấn đáp:

- Tại sao data lại ít ảnh ?
    - ➢ Do link GitHub của nhiều nhóm gửi lên bị sai format nên không clone được.

★ Đã khắc phục được phần nào khi giờ đây số lượng ảnh được clone về đã lên con số khoảng 7900 ảnh.

- Earlystopping là gì?
    - ➢ Kỹ thuật nhằm ngăn ngừa hiện tượng overfitting (quá khớp) tức là tránh cho mô hình học quá kỹ vào mô hình huấn luyện khiến cho mô hình không học được quy luật tổng quát mà ghi nhớ luôn cả dữ liệu nhiễu, làm giảm hiệu suất của mô hình. Kỹ thuật này theo dõi hiệu suất trên tập validation trong quá trình huấn luyện và khi hiệu suất mô hình không được cải thiện sau một số epoch nhất định thì việc huấn luyện sẽ được dừng lại trước epoch được định trước.
    - ➢ VD: Như mô hình của nhóm thì epoch là 100 và khi huấn luyện thì mô hình (VGG16) dừng khi epoch đạt 29 sau khi thay đổi epoch 5 lần mà mô hình không có cải thiện về hiệu suất.
- Tại sao sử dụng mô hình VGG16?
    - ➢ Nhóm sử dụng 2 mô hình là VGG16 và ResNet50 với cùng 1 cách xử lý dữ liệu.
    - ➢ Sau quá trình huấn luyện và đánh giá, VGG16 là phiên bản cho kết quả tốt nhất. Biểu đồ cho thấy training accuracy và validation accuracy đều có mức tăng đều đặn qua từng epoch chứng tỏ mô hình đang được học tốt, và biểu đồ Loss cũng cho thấy điều tương tự khi cả training lẫn validation đều là những hàm giảm ổn định. Hàm có chút

giao động ở 5 giá trị epoch cuối. Đây là cũng là lý do xảy ra earlystopping.

- ➢ ResNet50: phiên bản này gặp vấn đề hiệu suất: biểu đồ Loss cho thấy kết quả không tốt (ổn định mức 2.3 so với khoảng 0.4 của mô hình VGG). Biểu đồ accuracy cũng cho thấy dấu hiệu mô hình không hội tụ trên tập training. Điều này xảy ra có thể do dữ liệu đầu vào là không đủ cho mô hình có thể học và đúc rút đặc trưng.

★ Đã bổ sung thêm mô hình ResNet50

Chương 1: GIỚI THIỆU BÀI TOÁN

- Mô tả: Dự đoán chữ số viết tay (từ 0-9) của các thành viên trong lớp dựa trên ảnh chữ viết tay của chính họ.
- Mục tiêu: Dự đoán chữ số được viết trong ảnh
- Input:
  + Tập dữ liệu gồm có N phần tử đã được gán nhãn:

$$D = \{(xi, yi)\}_{i=1}^{N}$$

    - Tập các nhãn D là: $L = U_{i=1}^{N} y_i$
    - Trong mỗi phần tử của D:
      - x_i là ảnh số J hoặc là một vector đặc trưng có kích thước d chiều: $x_i \in J = R^d$
      - y_i là giá trị nhãn (label) được gán cho x_i
  + Ảnh $x \in J$
- Output:
  + $\hat{y} \in L$: giá trị nhãn dự đoán của ảnh x

Hình 1. Pipeline cơ bản cho bài toán

Chương 2: DATA PROCESSING

1. Chuẩn bị dữ liệu:

1.1. Chuẩn bị tập dữ liệu huấn luyện:

- Thu thập link github được các bạn gửi lên course, sau đó tiến hành clone về máy.
- Do một vài nhóm gửi link github sai format dẫn đến việc clone thất bại.
- Thực hiện quá trình gán nhãn thủ công (gồm 10 nhãn từ "0" đến "9")
- Tổng số ảnh clone được về máy là 4250 file, trong đó:
    + Tổng số ảnh dùng để huấn luyện: 3482.
    + Tổng số ảnh dùng để đánh giá: 697.
    + Tổng số ảnh dùng để kiểm thử (được gán nhãn): 778.

Hình 2. Cấu trúc dữ liệu đầu vào sau xử lý.

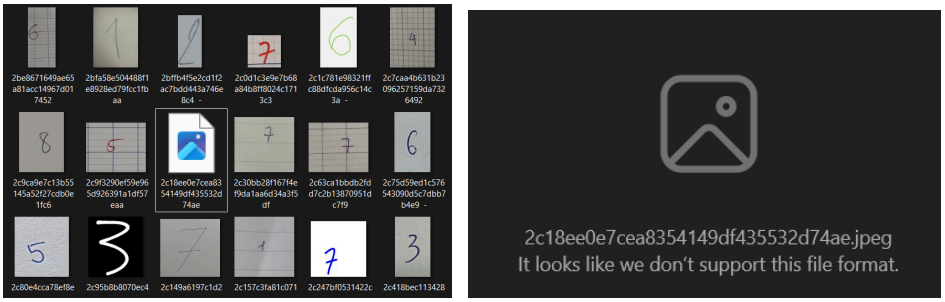
1.2. Tập dữ liệu cần được dự đoán:

- Được cung cấp sẵn bởi giáo viên qua đường link Drive:

  https://drive.google.com/file/d/1UhMPzU-84KYH1-ujisJj08kb4Ji7gBLe/view

- Mô tả cấu trúc dữ liệu tập dự đoán:

+ Tập dữ liệu gồm khoảng 10000 ảnh với nhiều kích thước khác nhau, không được gán nhãn với đa dạng định dạng: jpg, png, heic, .v.v
+ Một vài ảnh trong số đó không không thể đọc được vì lý do nào đó.



Hình 3. Mô tả dữ liệu của tập huấn luyện.

2. Load ảnh từ folder:

2.1. Tìm danh sách nhãn (label):

- Lấy danh sách thư mục con, sắp xếp và tạo bản đồ nhãn (label_map), gán mỗi class thành một số nguyên.

```
{'0': 0, '1': 1, '2': 2, '3': 3, '4': 4, '5': 5, '6': 6, '7': 7, '8': 8, '9': 9}
```

Hình 4. Bảng ánh xạ

2.2. Duyệt file ảnh trong từng folder:

- Mở từng file ảnh trong thư mục con.
- Lấy đường dẫn những file có đuôi là " .jpg", ".jpeg", ".png".
- Sử dụng imread() được hỗ trợ bởi thư viện OpenCV để đọc ảnh màu, kết quả trả về là mảng NumPy có kênh BGR
- Nếu file bị lỗi hoặc không thể đọc đường thì bỏ qua.

Chương 3: TRAINING

1. Trước khi bắt đầu huấn luyện:

1.1. Tiền xử lý ảnh:

1.1.1. Chuyển đổi kênh màu:

- Sử dụng hàm cvtColor() được hỗ trợ bởi thư viện OpenCV để chuyển đổi từ kênh màu mặc định của OpenCV là BGR thành kênh màu RGB.

1.1.2. Resize lại kích thước ảnh:

- Sử dụng hàm resize() được hỗ trợ bởi thư viện OpenCV để thay đổi kích thước ảnh thành phù hợp.

1.1.3. One-hot encoder:

- Chuyển nhãn đã được mã hóa từ số nguyên sang một vector xác suất có số chiều bằng số lớp, phù hợp với hàm mất mát categorical_crossentropy và đầu ra softmax của mô hình huấn luyện.
- VD: Có 4 ảnh với 4 nhãn là lần lượt là 1, 2, 3, 4 thì sau quá trình one-hot, ta được: 1 -> [1, 0, 0, 0] (100% thuộc lớp 1], tương tự với 2, 3 và 4.

1.2. Tách dữ liệu:

- Dùng hàm train_test_spit được hỗ trợ bởi thư viện scikit-learn để chia tập dữ liệu ban đầu thành 2 tập train-validation với tỉ lệ 8-2.
- Dùng tham số stratify để giữ tỉ lệ giữa các class cân bằng.

1.3. Chuẩn bị model:

- Import mô hình học sâu VGG16 từ thư viện tensorflow đã được pre-train với tập ImageNet, đồng thời loại bỏ khối "Top" để thêm các layer phân loại riêng.
- Đóng băng toàn bộ layer, chỉ fine-tune khối "Top".
- Các layer được thêm mới bao gồm:
  + 1 lớp Flatten()
  + 1 lớp Dropout()
  + 2 lớp Dense(), trong đó lớp Dense() cuối cùng là đầu ra.

1.4. Định nghĩa các tham số huấn luyện:

1.4.1. Đối với VGG16:

1.4.1.1. Trọng số:

- Load mô hình học sâu VGG16 đã được pre-trained với tập dữ liệu ImageNet.

1.4.1.2. Kích thước ảnh:

- Sau khi thử nghiệm với nhiều kích thước khác nhau, 128x128 cho ra kết quả tốt nhất.

Kích thước	Độ chính xác cao nhất
32x32	0.42
64x64	0.69
128x128	0.81
224x224	0.8

Bảng 1. So sánh kết quả với các kích thước ảnh khác nhau khi dùng mạng VGG16.

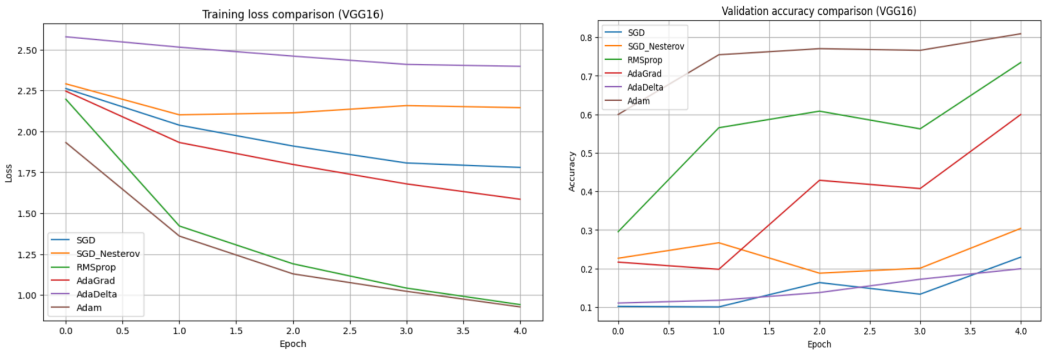
1.4.1.3. Khối 'head':

- Đóng băng tất cả layer cũ trừ khối 'head' để customize khối này.
- Cấu trúc của khối:
  - + 1 lớp Flatten()
  - + 1 lớp Dropout()
  - + 2 lớp Dense(), trong đó lớp Dense() đầu tiên giúp học các đặc trưng phân biệt, lớp còn lại là đầu ra dùng để phân loại.

➢ Do mô hình VGG16 có cấu trúc phức tạp dẫn đến việc không thể kiểm thử từng tham số vì vấn đề thiếu tài nguyên. Do đó các tham số được sử dụng trong khối 'head' được tham khảo từ nguồn: https://akanshasaxena.com/challenge/deep-learning/day-11/

1.4.1.4. Optimizer:

- Thực hiện so sánh giữa ADAM và các Optimizer khác, cả chỉ số Loss và Accuracy đều cho ra kết quả tốt nhất.



Hình 5. Loss và Accuracy của các Optimizer với mạng VGG16.

1.4.2. Đối với mạng ResNet50:

1.4.2.1. Trọng số:

- Load mô hình học sâu ResNet50 đã được pre-trained với tập dữ liệu ImageNet.

1.4.2.2. Kích thước ảnh:

- Sau khi thử nghiệm với nhiều kích thước khác nhau, kết quả cho thấy kích thước 224x224 cho ra kết quả tốt nhất.

Kích thước	Độ chính xác trên tập test
32x32	0.14
64x64	0.28
128x128	0.33
224x224	0.34

Bảng 2. So sánh kết quả với các kích thước khác nhau khi dùng mạng ResNet50

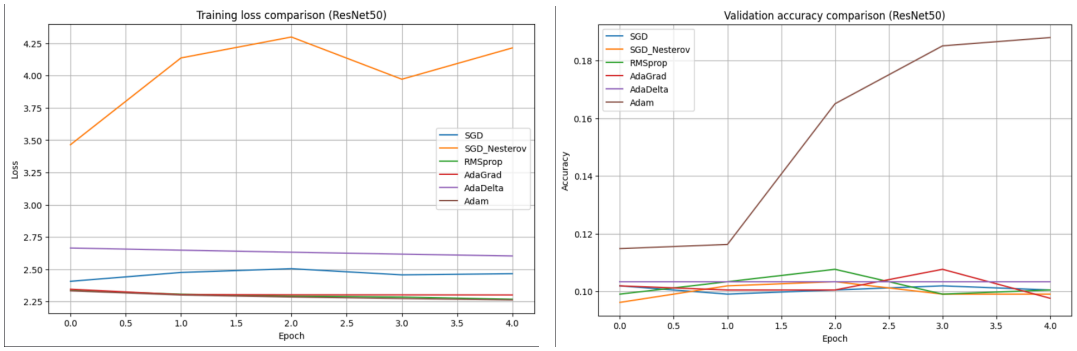
1.4.2.3. Khối 'head':

- Đóng băng tất cả layer cũ trừ khối 'head' để customize khối này.
- Cấu trúc của khối:
  - + 1 lớp GlobalAveragePooling2D()
  - + 1 lớp Flatten()
  - + 1 lớp Dense(), trong đó lớp Dense() là đầu ra dùng để phân loại.

➢ Do mô hình ResNet50 có cấu trúc phức tạp dẫn đến việc không thể kiểm thử từng tham số vì vấn đề thiếu tài nguyên. Do đó các tham số được sử dụng trong khối 'head' được tham khảo từ nguồn: https://viblo.asia/p/gioi-thieu-mang-resnet-vyDZOa7R5wj

1.4.2.4. Optimizer:

- Thực hiện so sánh giữa ADAM và các Optimizer khác, tuy cả ADAM và RMSprop đều cho giá trị Loss tương đương nhưng với Accuracy, ADAM lại vượt trội hơn hoàn toàn so với phần còn lại.



Hình 6. Loss và Accuracy của các Optimizer với mạng ResNet50.

2. Trong quá trình huấn luyện:

- Cài đặt số epoch tối đa là 100 do vấn đề tài nguyên và tiết kiệm thời gian.
- Trong mỗi epoch sẽ in ra:
  + Loss và Accuracy trên tập train
  + Loss và Accuracy trên tập validation
- Sử dụng EarlyStopping để dừng sớm:
  + Quan sát giá trị val_loss để biết khi nào dừng.
  + Nếu val_loss không giảm trong 5 epochs liên tiếp thì quá trình training sẽ kết thúc và trả về model với trọng số tốt nhất.

3. Sau khi quá trình huấn luyện kết thúc:

Mô hình được lưu lại thành file '.keras' để tái sử dụng cho việc dự đoán.

Chương 4: EVALUATION

1. Các độ đo được sử dụng:

1.1. Accuracy:

- Accuracy cho biết độ chính xác tổng thể của mô hình, được sử dụng trong cả quá trình train/valid và test.
- Công thức:

$$Acc = \frac{Số\ lượng\ ảnh\ dự\ đoán\ đúng}{Tổng\ số\ mẫu\ ảnh}$$

1.2. Recall:

- Recall của lớp i cho biết tổng số mẫu mô hình dự đoán thuộc lớp i trên toàn bộ tập dữ liệu.
- Công thức:

$$Recall = \frac{TP}{TP+FN}$$

- $Macro\ Recall = \frac{1}{C}\sum_{i=1}^{C}\frac{TP_i}{TP_i + FN_i}$

1.3. Precision:

- Precision của lớp i cho biết trong tổng số mẫu mô hình dự đoán thuộc lớp i có bao nhiêu mẫu thật sự đúng.
- Công thức:

$$Precision = \frac{TP}{TP+FP}$$

- $Macro\ Precision = \frac{1}{C}\sum_{i=1}^{C}\frac{TP_i}{TP_i + FP_i}$

1.4. F1-score:

- Độ đo hài hòa giữa precision và recall, tổng hợp cả hai yếu tố: chính xác và đầy đủ.
- Công thức:

$$F1 = \frac{2*Precision*Recall}{Precision + Recall}$$

- $Macro\ F1 = \frac{2}{C}\sum_{i=1}^{C}\frac{Precision_i . Recall_i}{Precision_i + Recall_i}$

2. Phương pháp đánh giá:

Với các kích thước ảnh phù hợp đã được xác định trước đó, mỗi mô hình được chạy với cùng tham số, và nhóm ghi nhận các chỉ số như:

- Macro Recall
- Macro Precision
- Macro F1
- Accuracy

Sau đó nhóm chọn ra mô hình có chỉ số tốt nhất để sử dụng cho giai đoạn dự đoán cuối cùng.

3. Kết quả đánh giá:

Mô hình	Kích thước ảnh	Macro Recall	Macro Precision	Macro F1	Accuracy	Kết quả thực nghiệm (điểm trên wecode)
VGG16	128x128	0.81	0.82	0.81	0.81	85
ResNet50	224x224	0.34	0.48	0.31	0.34	35

Bảng 3. Bảng kết quả các tiêu chí đánh giá của 2 mô hình

Chương 5: KẾT LUẬN

1. **Về kết quả:**

- Tương quan kết quả wecode cho thấy mô hình của nhóm chưa thực sự hiệu quả so với đa số các nhóm khi kết quả của đa số nhóm đều trên 90 điểm (nhóm được 85)
- Tuy nhiên, mô hình vẫn có sự hội tụ trên tập dữ liệu và kết quả cho thấy mô hình cũng có sự phân loại hình ảnh ở một mức độ nào đó.

2. **Về phương hướng phát triển:** Để nâng cao hiệu quả của mô hình, cần thực hiện các cải tiến ở nhiều khía cạnh:

- Về mặt dữ liệu huấn luyện: dùng các biện pháp tăng cường dữ liệu có kiểm soát (flip, motion blur, .v.v) để tăng tính đa dạng của dữ liệu cũng như tăng thêm số epoch huấn luyện để cái thiện hiệu suất mô hình
- Về mặt xử lý dữ liệu: Kiểm thử với nhiều phương pháp xử lý dữ liệu hơn.
- Cố gắng khắc phục tình trạng thiếu tài nguyên để huấn luyện dữ liệu.

3. **Cảm nghĩ của nhóm:**

- Điểm tốt:
    + Hiểu được quy trình từ chuẩn bị dữ liệu, xây dựng mô hình, huấn luyện, đánh giá, đến triển khai.
    + Biết cách dùng transfer learning với các backbone phức tạp (VGG16, ResNet50).
    + Hiểu rằng ngoài các siêu tham số (Optimization, .v.v) thì kích thước của ảnh đầu vào cũng có ảnh hưởng nhất định đến hiệu suất của mô hình.
- Điểm cần cải thiện:
    + Tuy nhóm đã có sự cải thiện về số lượng ảnh đã được clone về từ github của các bạn (7900 ảnh so với 4250 trước đây) nhưng do thời gian có hạn, nhóm đã không thể huấn luyện với bộ dữ liệu mới. Do đó hiệu suất mô hình vẫn không có sự thay đổi so với trước.

+ Mô hình vẫn chưa tốt so với các nhóm khác (nhiều nhóm đạt hơn 90 điểm, thậm chí có nhóm đạt được 99 điểm trên scoreboard), điều này có thể do mô hình học chưa đủ, dữ liệu chưa đa dạng, hoặc các mạng học sâu được chọn chưa đủ tốt với tập dữ liệu.

Phần B: ĐỒ ÁN DỰ ĐOÁN ĐIỂM.

Phần 0: CÁC THAY ĐỔI SO VỚI KHI VẤN ĐÁP

Trả lời câu hỏi của thầy cho bài tập dự đoán điểm wecode:

- Trích xuất đặc trưng của mô hình là gì ?
    + Quá trình chuyển đổi dữ liệu thô thành dữ liệu có ý nghĩa trong quá trình train mô hình
    + Mô hình dự đoán điểm của nhóm trích đặc trưng như: tổng số test, tổng số test đúng, tỉ lệ test đúng, số lần nộp trung bình, bộ nhớ trung bình, số test bị sai
- Có bao nhiêu đặc trưng ?
    + Mô hình dự đoán điểm quá trình của nhóm bao gồm 17 đặc trưng theo sinh viên
        - num_assignments: mã bài tập
        - num_problems: mã câu hỏi
        - num_submissions: tổng số lượt nộp bài
        - avg_score: điểm trung bình
        - max_score: điểm cao nhất bài
        - avg_penalty: trung bình hệ số phạt
        - avg_wrong_tests: trung bình số test sai
        - num_correct: tổng số câu đúng nộp đúng hạn
        - num_late: tổng lần nộp trễ
        - num_score_status: tổng số bài được chấm điểm
        - active_days: ngày nộp bài
        - mean_correct_rate: điểm trung bình các bài nộp
        - mean_time_per_test: trung bình số lần nộp
        - mean_mem_per_test: trung bình bộ nhớ mỗi bài nộp
        - score_ratio: tỉ lệ lượt được chấm điểm
        - correct_ratio: tỉ lệ lượt đúng

    - late_ratio: tỉ lệ nộp trễ
  + Trong 17 đặc trưng thì bao gồm 3 đặc trưng "tỉ lệ" được tính thêm từ các đặc trưng có sẵn.
- Đặc trưng nào theo nhóm là quan trọng ?
  + Những đặc trưng quan trọng bao gồm những đặc trưng tỉ lệ: tỉ lệ bài được chấm điểm, tỉ lệ bài nộp đúng; điểm trung bình. Đây là những đặc trưng ảnh hưởng rõ nhất đến năng lực làm bài của sinh viên

Chương 1: GIỚI THIỆU BÀI TOÁN

Mô tả: Dự đoán điểm Thực hành IT001 của sinh viên dựa trên lịch sử nộp bài trên hệ thống Wecode. Dữ liệu đầu vào chứa thông tin về từng lần nộp bài của sinh viên như: bài được giao (assignment), bài cụ thể (problem), trạng thái nộp bài (được chấm điểm hay không), kết quả chạy (pass/fail), % số testcase đúng, thời gian và bộ nhớ sử dụng, cùng với thời gian nộp và chấm bài.

Mục tiêu: Dự đoán điểm Thực hành IT001 cho từng sinh viên dựa trên hành vi và kết quả nộp bài trên Wecode.

Input:

- Tập dữ liệu gồm nhiều dòng tương ứng với các lần nộp bài của sinh viên, mỗi dòng bao gồm các thông tin sau:
- assignment_id: mã bài tập
- problem_id: mã bài cụ thể
- student_id: mã số sinh viên đã được ẩn danh
- is_final: lần nộp này có được tính điểm hay không
- status: kết quả chạy chương trình (có chạy được hay không)
- coefficient: hệ số phạt do nộp trễ hạn
- pre_score: % số testcase đúng (đã làm tròn và chuyển sang số nguyên)
- created_at: thời gian nộp bài (không có thông tin năm)
- updated_at: thời gian chấm bài
- judgement: thông tin chi tiết kết quả chấm (thời gian, bộ nhớ sử dụng cho từng testcase, số testcase sai)

Ngoài ra, nhóm còn được cung cấp thêm các file điểm thật của khoảng 800 sinh viên, bao gồm:

- th-public.csv: điểm Thực hành
- qt-public.csv: điểm Quá trình

- tbtl-public.csv: điểm Trung bình tích lũy

Output:

- Điểm thực hành IT001 được mô hình dự đoán cho từng sinh viên có trong file annonimized.csv.
- Ràng buộc: phải dự đoán điểm cho tất cả các mã số sinh viên trong file annonimized.csv, kể cả khi không có thông tin rõ ràng về điểm của họ. Nếu không nộp dự đoán cho một sinh viên nào đó, hệ thống sẽ xem như dự đoán điểm của người đó là 0.

Tiêu chí đánh giá: Kết quả dự đoán được đánh giá bằng hệ số xác định R^2 score.

Chương 2: DATA PROCESSING

1. Thư viện sử dụng

Trong quá trình xử lý và chuẩn bị dữ liệu, các thư viện Python sau được sử dụng:

pandas: dùng để đọc, xử lý, làm sạch và ghép nối các bảng dữ liệu dạng CSV.

numpy: hỗ trợ thao tác số học và xử lý mảng hiệu quả.

json (nội tại Python): dùng để phân tích chuỗi JSON từ cột judgement.

re: hỗ trợ xử lý chuỗi bằng biểu thức chính quy (nếu cần xử lý sâu hơn trong judgement).

datetime: chuyển đổi và phân tích thời gian từ các cột created_at, updated_at.

2. Các file dữ liệu

2.1. File annonimized.csv

- Chứa thông tin chi tiết về từng lần nộp bài của sinh viên, với các trường:
- assignment_id, problem_id: mã bài tập và bài cụ thể
- username: mã định danh ẩn danh của sinh viên
- is_final: có tính điểm hay không
- status: trạng thái nộp bài (SCORE, WRONG_ANSWER, COMPILE ERROR, v.v.)
- pre_score: % testcase đúng (dưới dạng số nguyên)
- coefficient: hệ số phạt nộp trễ
- language_id: ngôn ngữ lập trình
- created_at, updated_at: thời gian nộp và chấm bài
- judgement: thông tin chi tiết (JSON) gồm:
- "times": danh sách thời gian mỗi testcase
- "mems": danh sách bộ nhớ mỗi testcase
- "verdicts": kết quả chấm, ví dụ {"WRONG":10}

2.2. File th-public.csv

Chứa điểm Thực hành IT001 thật của một số sinh viên:

- hash: mã sinh viên (ẩn danh), khớp với username trong annonimized.csv
- TH: điểm thực hành (thang điểm 10)

3. Ghép nối dữ liệu

Để tạo tập huấn luyện, nhóm chúng em đã:

3.1. Tiền xử lý annonimized.csv:

☐ Đổi tên cột

Các cột trong file có tên dài do sử dụng hàm concat(...) từ hệ thống. Ta cần đổi lại tên cho dễ thao tác, ví dụ:

concat('it001', assignment_id) → assignment_id

concat('it001', username) → username

v.v...

☐ Chuyển đổi định dạng ngày giờ

Hai cột created_at và updated_at có định dạng không chuẩn (thiếu năm).

Ta thêm thủ công năm (2025) và chuyển sang định dạng datetime, giúp trích xuất các đặc trưng thời gian như ngày, giờ, độ trễ khi chấm.

☐ Làm sạch dữ liệu

Kiểm tra và xử lý các giá trị thiếu, lỗi hoặc không hợp lệ (nếu có).

Kiểm tra định dạng cột số, đặc biệt là pre_score, coefficient.

☐ Xử lý chuỗi JSON trong judgement

Cột judgement chứa chuỗi JSON gồm thông tin về thời gian, bộ nhớ và kết quả từng testcase.

Cần parse chuỗi này để trích xuất các thông tin định lượng có ý nghĩa (sẽ chi tiết ở bước sau).

☐ Tạo các đặc trưng phụ

Tạo thêm các cột nhị phân giúp mô hình hiểu rõ hơn hành vi của người học:

- is_final: lần nộp có tính điểm không
- is_correct: lần nộp đạt 100% testcase hay không
- is_late: lần nộp bị trừ điểm (hệ số < 100)
- is_scored: lần nộp có status là SCORE
- day: trích ra ngày từ created_at để đếm số ngày hoạt động

3.2. Tính toán các thống kê theo sinh viên:

- Tổng số lần nộp
- Tỷ lệ đúng cao nhất
- Số lần nộp tính điểm
- Tỷ lệ các lỗi (sai toàn bộ, chạy lỗi, ...)

3.3. Ghép với điểm thật từ th-public.csv:

Dùng lệnh merge theo mã username và hash.

Kết quả thu được là tập dữ liệu có các đặc trưng đầu vào (X) và nhãn đầu ra là điểm thực hành (y = TH).

Chương 3: TRAINING

1. Trước khi bắt đầu huấn luyện:

1.1. Trích xuất đặc trưng từ cột judgement

Từ thông tin trong cột judgement (dạng JSON), hệ thống tiến hành trích xuất các đặc trưng bao gồm:

- Tổng số testcase
- Số testcase đúng / sai
- Tỷ lệ testcase đúng
- Thời gian trung bình mỗi testcase
- Bộ nhớ trung bình mỗi testcase

Trong quá trình xử lý, nếu chuỗi judgement bị lỗi hoặc không đọc được, các đặc trưng được gán giá trị 0 để tránh lỗi phân tích.

1.2. Xử lý các cột cơ bản

Một số cột được xử lý lại để tạo thêm các đặc trưng mới:

Chuyển đổi cột thời gian created_at sang định dạng chuẩn datetime, bổ sung cột ngày nộp (day)

Tạo cột nhị phân:

- is_correct: lần nộp đạt điểm tuyệt đối (100%)
- is_late: lần nộp bị phạt (hệ số < 100)
- is_scored: lần nộp được chấm điểm (status = SCORE)

1.3. Tổng hợp đặc trưng theo sinh viên

Dữ liệu ban đầu ở cấp độ lần nộp bài, được gom lại ở cấp độ sinh viên. Với mỗi sinh viên, thống kê các đặc trưng sau:

- Số lượng bài tập khác nhau (num_assignments)
- Số lượng bài cụ thể (num_problems)
- Tổng số lần nộp (num_submissions)

- Trung bình và điểm cao nhất (avg_score, max_score)
- Trung bình hệ số nộp trễ (avg_penalty)
- Trung bình số testcase sai trên mỗi lần nộp (avg_wrong_tests)
- Số lần nộp đạt điểm tuyệt đối (num_correct)
- Số lần nộp trễ (num_late)
- Số lần nộp được chấm điểm (num_score_status)
- Số ngày sinh viên có hoạt động (active_days)
- Trung bình tỉ lệ testcase đúng (mean_correct_rate)
- Trung bình thời gian và bộ nhớ mỗi testcase (mean_time_per_test, mean_mem_per_test)
- Sau đó, thêm các tỉ lệ đặc trưng:
- Tỉ lệ nộp được chấm điểm (score_ratio)
- Tỉ lệ nộp đúng hoàn toàn (correct_ratio)
- Tỉ lệ nộp trễ (late_ratio)

1.4. Ghép điểm thực hành

Tập đặc trưng sau khi tổng hợp được ghép với file th-public.csv để gắn nhãn là điểm thực hành IT001. Việc ghép được thực hiện dựa trên username.

1.5. Tách tập huấn luyện và đánh giá

Sau khi ghép dữ liệu đặc trưng với bảng điểm thực hành thật, hệ thống tách dữ liệu thành hai phần:

Tập huấn luyện (train_data): gồm các sinh viên đã có điểm thực hành (TH ≠ null). Đây là dữ liệu dùng để huấn luyện mô hình.

Tập cần dự đoán (test_data): gồm các sinh viên chưa có điểm thực hành (TH = null). Mô hình sau huấn luyện sẽ được dùng để dự đoán điểm cho nhóm này.

Từ tập huấn luyện, hệ thống tách thành:

- X_train: gồm 17 đặc trưng đầu vào đã được xử lý ở các bước trước
- y_train: điểm thực hành thật (biến mục tiêu)

- Tập X_test cũng được chuẩn bị tương tự nhưng không có nhãn.

2. Bắt đầu huấn luyện mô hình và chọn siêu tham số:

Mô hình hồi quy chính được lựa chọn là Random Forest Regressor – một mô hình học máy thuộc nhóm ensemble learning (học tổ hợp), hoạt động dựa trên việc xây dựng nhiều cây quyết định ngẫu nhiên và tổng hợp kết quả để dự đoán.

Ưu điểm:

- Mạnh mẽ với dữ liệu nhiễu
- Tự động khai thác tương tác phi tuyến giữa các đặc trưng
- Không yêu cầu chuẩn hóa đặc trưng đầu vào
- Hạn chế overfitting so với cây đơn

Thay vì cố định các tham số mô hình, hệ thống sử dụng GridSearchCV để tìm bộ siêu tham số tối ưu, nhằm tăng hiệu suất dự đoán. Cụ thể:

Bộ siêu tham số thử nghiệm:

- n_estimators: số lượng cây trong rừng (thử các giá trị 100, 200, 300)
- max_depth: độ sâu tối đa của mỗi cây (thử 8, 10, 12)
- min_samples_split: số mẫu tối thiểu để chia nhánh (thử 2, 5, 10)

```
Best params: {'max_depth': 12, 'min_samples_split': 10, 'n_estimators': 200}

Best R^2 score (CV): 0.35425448090327144
```

Hình 7: Các siêu tham số tìm dược sau quá trình GridSearchCV

3. Các mô hình khác:

Ngoài Random Forest (RF), nhóm còn thử nghiệm với 2 mô hình khác là Linear Regression (LR) và Support Vector Regression (SVR), kết quả thực nghiệm cho thấy mô hình tốt nhất là được chọn ban đầu:

Tên mô hình	Điểm R^2	Siêu tham số
RF	0.3543	max_depth = 12 min_sample_split = 10 n_estimators = 200
LR	0.2677	x
SVR	0.1926	C = 10 epsilon = 0.5 kernel = rbf

Bảng 4: Các mô hình với các siêu tham số cho ra kết quả tốt nhất

4. Sau khi quá trình huấn luyện kết thúc:

Dùng mô hình tốt nhất để thực hiện dự đoán điểm của những sinh viên chưa có điểm trên tập dữ liệu được cung cấp bởi giảng viên.

Chương 4: EVALUATION

1. Độ đo sử dụng: R² Score

Trong bài toán dự đoán điểm số (bài toán hồi quy), hệ số đánh giá chính được sử dụng là R² score (coefficient of determination).

Ý nghĩa:

R² cho biết mức độ mô hình giải thích được phương sai của dữ liệu thực tế.

Giá trị nằm trong khoảng (-∞, 1], trong đó:

R² = 1: mô hình dự đoán hoàn hảo

R² = 0: mô hình không tốt hơn dự đoán trung bình

R² < 0: mô hình còn tệ hơn cả việc đoán giá trị trung bình

Công thức	$R^2 = 1 - \frac{SS_{res}}{SS_{tot}}$
SS_{res}: Tổng bình phương sai số dự đoán	$SS_{res} = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2$
SS_{tot}: Tổng bình phương sai số so với trung bình	$SS_{tot} = \sum_{i=1}^{n}(y_i - \bar{y})^2$

Bảng 5: Bảng công thức tính độ đo R²

2. Phương pháp đánh giá:

Dùng cross-validation 3-fold (chia dữ liệu thành 3 phần), đo lường hiệu quả mô hình bằng chỉ số R² score – cho biết mức độ mô hình giải thích được phương sai của dữ liệu thực tế.

Cách thức thực hiện:

- Với mỗi tổ hợp tham số, hệ thống huấn luyện mô hình 3 lần (với 3 fold khác nhau)
- Trung bình R^2 trên 3 fold được tính làm chỉ số đánh giá
- Tổ hợp nào có điểm R^2 cao nhất sẽ được chọn

3. Kết quả đánh giá

Tiêu chí đánh giá	Giá trị
Bộ siêu tham số tốt nhất	max_depth = 12 min_samples_split = 10 n_estimators = 200
R^2 score tốt nhất	0.354
Kết quả thực tế (kiểm tra trên wecode)	37

Bảng 6. Bảng kết quả các tiêu chí đánh giá của mô hình Random Forest trong dự đoán điểm thực hành IT001

Chương 5: KẾT LUẬN

1. Về kết quả

Kết quả dự đoán điểm thực hành IT001 cho thấy mô hình của nhóm đạt được $R^2 = 0.354$ và 37 điểm trên hệ thống chấm, thấp hơn so với nhóm có điểm cao nhất là 44 điểm, và nhìn chung vẫn dưới mặt bằng chung của các nhóm đạt kết quả tốt.

Tuy nhiên, mô hình vẫn có dấu hiệu hội tụ ổn định trên tập huấn luyện, cho thấy pipeline huấn luyện và xử lý đặc trưng của nhóm đã có hiệu quả nhất định. Mức độ phân biệt giữa các sinh viên (dựa trên hành vi nộp bài và lịch sử chấm điểm) tuy chưa cao nhưng đã được mô hình học ở một mức độ cơ bản.

2. Về hướng phát triển

Mặc dù mô hình đã học được một phần hành vi làm bài của sinh viên, nhưng khả năng giải thích phương sai điểm thực hành vẫn còn hạn chế. Điều này cho thấy mô hình còn dư địa để cải thiện thông qua:

Mở rộng và làm giàu đặc trưng đầu vào (ví dụ: phân tích sâu hơn nội dung judgement, thời điểm nộp bài, thống kê theo tuần,...)

Thử nghiệm thêm các mô hình hồi quy khác như XGBoost, LightGBM

Kết hợp nhiều nguồn dữ liệu hơn nếu có thể (như điểm giữa kỳ, hoạt động trên lớp,...)

Kết quả này cho thấy nhóm đã xây dựng được quy trình dự đoán đầy đủ và có hiệu quả nhất định, dù vẫn còn cần tinh chỉnh thêm để đạt hiệu suất tối ưu hơn.

3. Về cảm nghĩ của nhóm

Nhóm đánh giá rằng đề tài dự đoán điểm thực hành IT001 là một bài toán hay và có tính ứng dụng thực tiễn cao, đồng thời cũng khá thử thách vì yêu cầu kết hợp giữa hiểu dữ liệu giáo dục, kỹ năng xử lý dữ liệu phức tạp và triển khai mô hình hồi quy hiệu quả.

Trong quá trình thực hiện, nhóm đã hiểu rõ hơn về:

Cách xây dựng pipeline trong một bài toán học máy căn bản

Tầm quan trọng của trích xuất đặc trưng phù hợp từ dữ liệu thô, đặc biệt với cột judgement chứa thông tin dạng JSON

Việc lựa chọn mô hình, tối ưu siêu tham số, và đánh giá bằng chỉ số R^2

Tuy kết quả dự đoán chưa đạt mức cao nhất, nhưng thông qua bài toán này, nhóm đã:

- Nắm được quy trình huấn luyện và đánh giá mô hình hồi quy
- Cải thiện khả năng xử lý dữ liệu thực tế
- Có thêm kinh nghiệm làm việc nhóm và tự nghiên cứu tài liệu liên quan

Đây là một trải nghiệm học tập rất thực tế và bổ ích, giúp nhóm tiến gần hơn với việc áp dụng máy học vào các bài toán giáo dục và dữ liệu người dùng.

TÀI LIỆU THAM KHẢO

1. https://akanshasaxena.com/challenge/deep-learning/day-11/
2. https://viblo.asia/p/gioi-thieu-mang-resnet-vyDZOa7R5wj
3. https://scikit-learn.org/stable/modules/generated/sklearn.metrics.r2_score.html
4. https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVR.html#sklearn.svm.SVR
5. https://scikit-learn.org/stable/modules/linear_model.html
6. https://scikit-learn.org/stable/modules/ensemble.html#random-forests-and-other-randomized-tree-ensembles
7. ADAM: A METHOD FOR STOCHASTIC OPTIMIZATION - Diederik P. Kigma, Jimmy Lei Ba.