**ĐẠI HỌC QUỐC GIA TP HCM**



TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

****

BÁO CÁO CUỐI KỲ

**CS114 – MÁY HỌC**

**Sinh viên thực hiện:**

Trần Đình Trung – 16521327

**Giảng viên hướng dẫn:**

Ts. Lê Đình Duy

Ths. Huỳnh Thị Thanh Thương Ths. Phạm Nguyễn Trường An

# PHẦN 1: THÔNG TIN TÓM TẮT

|  |  |
| --- | --- |
| **Tên đề tài (Tiếng Việt)** | Nhận diện chữ cái viết tay sử dụng Deep Learning (A-Z)   * Link GitHub: |
| **Họ và tên Ảnh** | Trần Đình Trung - 16521327  https://lh3.googleusercontent.com/lsGkMQj1od_7QnlW7d71erTeJymOsm7Ow6SFSRVygznPKRd_iFd1Ka1Z0X7aRHPIufrExECeXKjd2_IYFTkLFtt3Mu-nd0vwomnv66nKqF2pqwDmiqdj72aIwvLWJJF_7R-bsrFG |
| **Số buổi vắng** | Unknow |
| **Số buổi đi trễ (không điểm danh)** | Unknow |
| **Số lần Comment trên Google Classroom** | Unkonw |
| **Tóm tắt Bài tập Quá trình** |  |
| **Tóm tắt Bài tập Cuối kỳ** |  |

**PHẦN 2: BÁO CÁO CHI TIẾT ĐỒ ÁN CUỐI KỲ**

**MỤC LỤC**

[CHƯƠNG 1: LỜI MỞ ĐẦU 6](#_bookmark0)

[CHƯƠNG 2: TIỀN XỬ LÝ DỮ LIỆU 7](#_bookmark1)

* 1. [Thu thập dữ liệu 7](#_bookmark2)
  2. [Chuẩn bị dữ liệu 8](#_bookmark4)

[CHƯƠNG 3: XÂY DỰNG MÔ HÌNH 10](#_bookmark10)

* 1. [Chọn mô hình 10](#_bookmark11)
  2. [Huấn luyện 12](#_bookmark13)
  3. [Đánh giá 13](#_bookmark17)
  4. [Điều chỉnh tham số 13](#_bookmark18)

[CHƯƠNG 4: THỬ NGHIỆM MÔ HÌNH 14](#_bookmark19)

[CHƯƠNG 5: ĐỊNH HƯỚNG PHÁT TRIỂN VẤN ĐỀ 15](#_bookmark23)

[CHƯƠNG 6: LỜI KẾT 16](#_bookmark24)

**MỤC LỤC HÌNH ẢNH**

[Hình 1. Load file từ shared google drive link 7](#_bookmark3)

[Hình 2. Đọc dữ liệu từ file csv 8](#_bookmark5)

[Hình 3. Kiểm thử điểm dữ liệu 8](#_bookmark6)

[Hình 4. Phân chia data và label, đếm số lượng điểm dữ liệu 8](#_bookmark7)

[Hình 5. Xáo trộn dữ liệu 9](#_bookmark8)

[Hình 6. Phân chia dữ liệu 9](#_bookmark9)

[Hình 7. Chọn mô hình 12](#_bookmark12)

[Hình 8. Định dạng data 12](#_bookmark14)

[Hình 9. Định dạng label 13](#_bookmark15)

[Hình 10. Kết quả huấn luyện 13](#_bookmark16)

[Hình 11. Thử nghiệm mô hình 14](#_bookmark20)

[Hình 12. Lưu trữ model 14](#_bookmark21)

[Hình 13. Kết quả lưu trữ model 14](#_bookmark22)

# CHƯƠNG 1: LỜI MỞ ĐẦU

Trí tuệ nhân tạo (Artifical Intelligent) hay máy học (Machine Learning) là một lĩnh vực đang được phát triển và ứng dụng rất nhiều trong thực tiễn. Có khá nhiều ứng dụng của nó như: google translate, xe tự động, hệ thống gợi ý, nhận diện vật thể,…

Trải qua một học kỳ môn học Machine Learning – CS114.K11, tuy chỉ mới là những bước đầu tiên trong một phạm trù máy học rộng lớn, nhưng cũng đủ cho tôi có khái niệm và định hướng đúng đắn trong việc học tập và phát triển sau khi ra trường theo hướng này.

Vì sao lại chọn một đề tài khá cơ bản? Đó là nhận diện chữ cái viết tay sử dụng deep learning (A-Z). Thay vì trên lớp tôi được học các thuật toán Machine Learning cơ bản theo 2 dạng Supervised và Unsupervised như là các thuật toán: KNN, Navie Bayes, Linear Regression, Logistic Regression, Decision Tree, K- means,… và đặc biệt là ý tưởng của thuật toán Gradient Descent. Cộng thêm việc khá ấn tượng với Neural Network. Vì thế tôi quyết định chọn một đề tài theo hướng Deep learning này. Sau khi xem xét nhiều lựa chọn, là yếu tố hơn hơn hết là có bộ dữ liệu công khai phù hợp với đồ án và các sources của những coder đã nghiên cứu trước đây. Đặc biệt là đủ các bước thực hiện của một mô hình Machine Learning. Cuối cuối đưa tôi đến lựa chọn nhận diện chữ cái viết tay sử dụng Deep Learning, vừa cơ bản nhưng lại có giá trị thực tế rất cao. Cụ thể giá trị của nó làm gì sẽ được nói rõ ở Chương 5: Định hướng phát triển vấn đề. Còn bây giờ, đến phần trọng tâm thôi!

# CHƯƠNG 2: TIỀN XỬ LÝ DỮ LIỆU

Trong chương này sẽ gồm 2 bước đầu tiên của một mô hình máy học:

# Thu thập dữ liệu

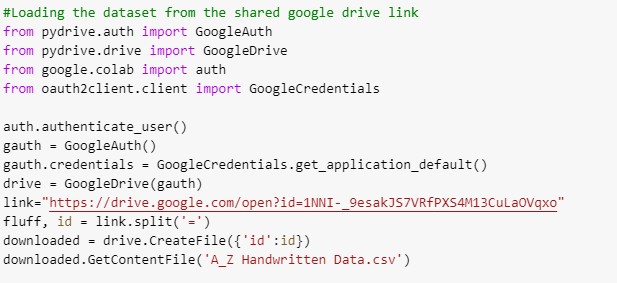
Dữ liệu thu thập sẽ là các hình ảnh chữ cái viết tay A-Z có cùng kích thước là chất lượng ảnh như nhau. Đừng nghĩ tới viết sẽ đi thu thập thủ công từng hình ảnh!

Sử dụng bộ dữ liệu sẵn có của Kaggle – Sân chơi của các Data Scientist trên khắp thế giới. Bộ dữ liệu A-Z Handwritten Alphabets in .csv format, link:

<https://www.kaggle.com/sachinpatel21/az-handwritten-alphabets-in-csv-format> với khoảng hơn 370 nghìn hình ảnh chứa 26 chữ cái Tiếng Anh kích thước chuẩn 28x28. Được định dạng bằng file .csv với mỗi điểm dữ liệu gồm 28x28 = 784 + 1 chiều. trong đó 784 chiều ứng với 784 pixel ảnh 28x28 và 1 chiều là label của nó. Label được đánh số từ 0 – 25 ứng với A – Z. Dễ dàng biến đổi các chiều dữ liệu này hình ảnh bằng thư viện matplotlib.pylot hàm imshow(). Bộ dữ liệu khoảng 666MB.

Sau khi download từ Kaggel về, lại upload ngược lên google drive tại link: <https://drive.google.com/open?id=1NNI-_9esakJS7VRfPXS4M13CuLaOVqxo>

ở chế độ công khai để dễ dạng sử dụng bởi Google Colab.

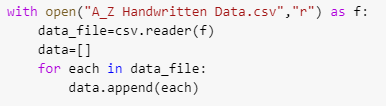


*Hình 1. Load file từ shared google drive link*

*Hình ảnh 1. Load file từ shared google drive link*

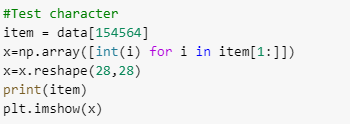
# Chuẩn bị dữ liệu

Tại bước thu thập dữ liệu, chúng ta đã có file dữ liệu với tên A\_Z Handwritten Data.csv, tiến hành đọc dữ liệu và phân chia dữ liệu:



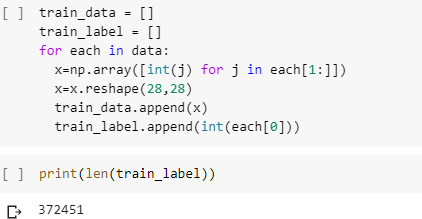
*Hình 2. Đọc dữ liệu từ file csv*

Kiểm tra xem dữ liệu có như mong muốn. Hiển thị một điểm dữ liệu bất kỳ:



*Hình 3. Kiểm thử điểm dữ liệu*

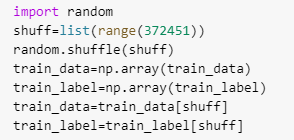
Việc cần làm tiếp sẽ là tiến hành phân chia ra 2 bộ dữ liệu data chứa 784 chiều dữ liệu pixel ảnh và label chứa nhãn của dữ liệu. 2 bộ này cùng tương đồng về index.



*Hình 4. Phân chia data và label, đếm số lượng điểm dữ liệu*

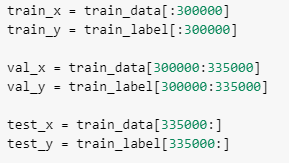
Ở đây ta có 372451 điểm dữ liệu, tiến hành phân chia testing và training.

Nhưng trước hết phải xáo trộn dữ liệu này đã để đảm bảo tính ngẫu nhiên. Mà tôi cũng không hiểu ý nghĩa tính ngẫu nhiên ở đây là gì nữa. Có lẽ một số bộ dữ liệu nó sẽ được làm theo thứ tự label, vì thế khi chia rất dễ bỏ sót các label cuối. Hầu hết các mô hình Machine Learning đều làm thế, nên cứ làm thế thôi. Việc xáo trộn phải bảo đảm index giữa data và label.



*Hình 5. Xáo trộn dữ liệu*

Tiến hành phân chia dữ liệu, vì dữ liệu đã được xáo trộn rồi giờ tiến hành phân tách thủ công với tỉ lệ tự chọn. Ở đây tôi chọn 300.000 điểm dữ liệu làm trainning set, 35.000 điểm dữ liệu làm validation set và số còn lại hơn 37 ngìn điểm dữ liệu làm test set.



*Hình 6. Phân chia dữ liệu*

Hoàn tất 2 bước đầu tiên của mô hình Machine Learning, tôi gọi chung là tiền xử lý dữ liệu.

# CHƯƠNG 3: XÂY DỰNG MÔ HÌNH

# Chọn mô hình

Như đã nói trước, tôi sẽ sử dụng Neural Network mà cụ thể là CNN (Convolutional Neural Network), một mô hình state-at-the-art cho các vấn đề nhận dạng hình ảnh như thế này. Chi tiết mô hình xem ở dưới.

Sử dụng thư viện TFLearn cùng với Tensowflow version 1 để xây dựng

CNN.

Chi tiết mô hình:

* Input: Tensor gồm 4 thành phần: Batch, height, width, in\_channels
  + Batch dùng mặc định là None, tương ứng với 32
  + Height là 28
  + Width cũng là 28
  + In\_channels là 26 (ứng với 26 class)
* Epochs chọn là 20, thông thường là lớn hơn 50 nhưng do phần cứng máy tính không đáp ứng đủ chạy đến 50 lần nên giảm tải còn 20.
* Gồm 6 lớp Convolutional layer liên tục và 2 lớp cuối Fully Connected Layer.
* Mỗi Convolutional layer gồm 2 phần chính convolutingvà pooling. Convoluting gồm 3 đầu vào cần lưu ý, số lượng filter, kích thước filter và hàm activation. Kích thước filter và hàm activation như nhau trong suốt 6 lớp. Cụ thể là filter\_size = 3 và activation là Relu. Còn số lượng filter sẽ được thay đổi với xen kẽ 32 và 62 giữa các lớp. Pooling được dùng là max\_pool với kernel\_size =2.
* Lớp Fully Connected Layer đầu tiền sẽ gồm 1024 unit. Sử dụng hàm activation là Relu. Tiến hành dropout với tỉ lệ keep\_prob = 0.8
* Lớp Fully Connected Layer sau cùng sẽ gồm 26 unit ứng với 26 lớp phân loại, và sử dụng hàm activation là softmax

Chi tiết về các thuật ngữ dùng ở trên khá nhiều và không phù hợp để đưa vào báo cáo này. Và tôi nghĩ lượng kiến thức của bản thân có thể trả lời các vấn đề liền quan tới mô hình này ở mức căn bản.

Vì sao lại lựa chọn mô hình này? Vì tôi kế thừa từ các tài liệu tham khảo các mô hình CNN áp dụng cho nhận dạng ký tự, số, ký hiệu v.v… và tài liệu tham khảo chính. Những thông số của nó có thể được thay đổi để làm bước thứ 6 của mô hình là điều chỉnh tham số, nhưng điều này là không cần thiết ở đồ án lần này, nó tốn rất nhiều thời gian và phần cứng máy tính.



*Hình 7. Chọn mô hình*

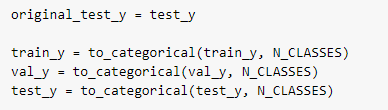
# Huấn luyện

Trước tiên cần phải định dạng lại data đầu vào cho phù hợp với mô hình:



*Hình 8. Định dạng data*

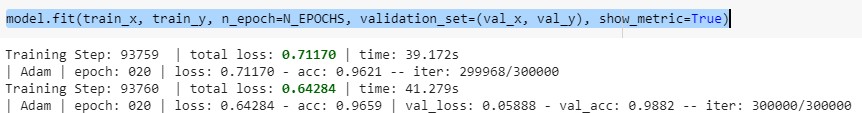
Tương tự với label, chuyển về dạng onehot vector:



*Hình 9. Định dạng label*

Đã xong mọi thứ, tiến hành huấn luyện bằng lệnh sau: model.fit(train\_x, train\_y, n\_epoch=N\_EPOCHS, validatio n\_set=(val\_x, val\_y), show\_metric=True)

Kết quả thu được:



*Hình 10. Kết quả huấn luyện*

# Đánh giá

Dựa vào hình trên ta có các nhận xét sau:

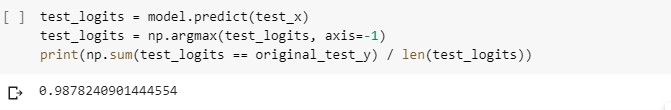
* + - Sau 20 lần thực hiện kết quả thu được độ chính xác với Vadition set là 98.82%
    - Thời gian mỗi lần thực hiện rơi vào khoảng 40s
    - Đây là 1 kết quả tốt với Epoch chỉ có 20. Thông thường ở mức Epoch 50 kết quả thu được sẽ tốt hơn rất nhiều rơi vào tầm 99.25 - 99.75%

# Điều chỉnh tham số

Như đã nói ở mục 3.1, việc điều chỉnh tham số trong đồ án này là không cần thiết vì lý do ở mức độ đồ án, thời gian và chi phí bỏ ra quá lớn

# CHƯƠNG 4: THỬ NGHIỆM MÔ HÌNH

Tiến hành thử nghiệm với test set

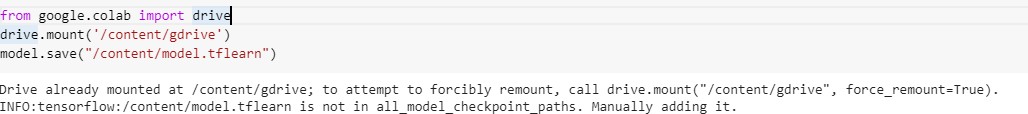


*Hình 11. Thử nghiệm mô hình*

Kết quả thu được với tỉ lệ là 98,78% tỉ lệ đúng

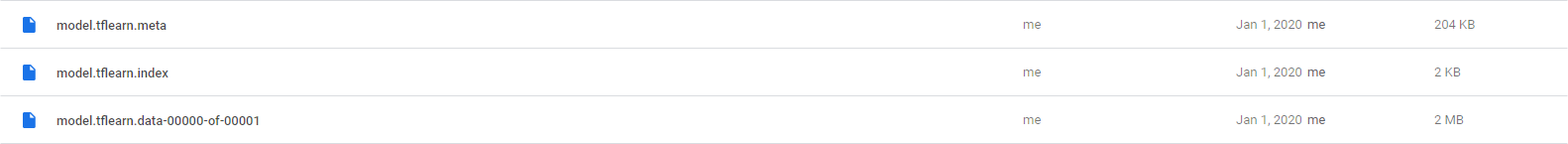
Một Model khá tốt để sử dụng, tuy nhiên vẫn cần nhiều thời gian và tài nguyên hơn để có thể nâng lên được hơn 99.5%.

Một điều quan trọng hơn nữa đó là lưu trữ lại model:



*Hình 12. Lưu trữ model*

Kết quả thu được sẽ tạo ra 3 file như sau:



*Hình 13. Kết quả lưu trữ model*

Sử dụng 3 file này để tái sử dụng cho các vấn đề liên quan bằng lệnh: model.load()

# CHƯƠNG 5: ĐỊNH HƯỚNG PHÁT TRIỂN VẤN ĐỀ

Ngay từ khi chọn đồ án Nhận dạng chữ cái viết tay này, em nhận thấy rõ điểm áp dụng thực tế cao của đồ án này.

Kết hợp cùng OpenCV và Segmentation có thể phát triển rất nhiều ứng dụng thực tế đã, đang, và sẽ có thể phát triển trong tương lai như sau:

\*Các ứng dụng đã và đang được áp dụng:

* Hệ thống chuyển đổi văn bản được chụp bằng camera chuyển đổi thành dữ liệu text, dễ dàng xử lý cho các vấn đề như dịch, tìm kiếm, quét thông tin khách hàng…

Ví dụ: Google Lens, Camsanner, Text Fairy, SmartOCR…

* Hệ thống nhận diện bảng số xe giao thông

Ví dụ: Áp dụng trong hệ thống thành phố thông minh hoặc phát hiện mất an toàn giao thông

* Hệ thống nhập dữ liệu theo dạng chữ viết tay

Ví dụ: Google Handwriting Input cho phép người dùng Adroid gõ chữ bằng cách viết các từ giống như viết trên giấy

* Hệ thống xác minh chữ ký
* Hệ thống Bank-Check Processing
* Writer Recognition
* Hệ thống Postal Address Interpretation
* Online Recognition
* Offline Recognition

Việc hiểu biết về mạng CNN nói chung, và nhận dạng chữ viết tay nói riêng sẽ giúp ích rất nhiều cho định hướng việc làm sau khi hoàn thành chương trình học.

# CHƯƠNG 6: LỜI KẾT

Chân thành cảm ơn các thầy cô giảng dạy môn học CS114.K11! Em đến với môn học với lý do muốn có cơ hội tìm hiểu sâu hơn về máy học, tìm được một định hướng gì mới hơn. Tham gia lớp học, không phải vì chưa hoàn thành chương trình học phần này mà là vì mong muốn của bản thân. Nhưng thật lòng bản thân em cảm thấy khá tiếc nuối. Có lẽ đây là một chương trình thử nghiệm và em không thoải mái về chương trình này. Em chân thành xin lỗi đến các thầy cô vì khi không tham gia đều đặn các lớp giảng dạy từ nửa học kỳ trở đi và không hoàn thành hết tất cả các bài tập, cụ thể là ML IN PRATICE #1.

Quay lại với đồ án lần này, đây là một trong 3 mô hình Neural Network em đã tự nghiên cứu và xây dựng cùng với Genetic CNN và Neural Network để tự động chơi game đơn giản tương tự T-rex Run của google. Và định hướng tiến đến DeepFake Detection để làm khóa luận tốt nghiệp.

Em xin kết thúc báo cáo tại đây! Chúc tất cả thầy cô và các bạn sinh viên nếu có đọc được có một sức khỏe dồi dào, cuộc sống hạnh phúc và thành công trong công việc!