Ho Chi Minh University of Science

Information Technology (HQ Program)

INTRODUCTION TO ARTIFICIAL INTELLIGENCE

PROJECT 3

CLASS: K19CLC6

MEMBERS:

Ngô Huy Anh – **19127095**
Phùng Anh Khoa – **19127449**
Triệu Nguyên Phát – **19127505**

September 3, 2021
PROGRESS: 100%

LECTURERS:

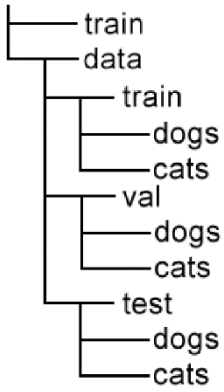
Châu Thành Đức
Phan Thị Phương Uyên
Ngô Đình Hy

I. COMPILING AND USING THE PROGRAM

1. How to compile

Các thư viện cần cài đặt: có thể thực hiện thông qua lệnh **pip install** <tên các thư viện dưới>
tensorflow
numpy
scipy
scikit-learn
tkinter
PIL

Các ứng dụng cần cài đặt:
CUDA Toolkit 11.4: https://developer.nvidia.com/cuda-toolkit
cuDNN 11.2 cho CUDA 11.4: yêu cầu tài khoản https://developer.nvidia.com/cudnn
Sau khi cài đặt **CUDA, copy** toàn bộ file trong thư mục 'cuda' sau khi giải nén **cuDNN** và **paste** vào thư mục chứa **CUDA** (Program Files/NVIDIA GPU Computing Toolkit/CUDA/v11.4/).
Tạo **solution** mới bằng **Visual Studio**, ngôn ngữ **Python**. **Add file code.py** và **checkpoint.h5** trong thư mục bài nộp vào **Project**.
Mở thư mục chứa **Project**, đưa **folder** 'train' lấy được từ **Kaggle** vào, cần tạo cây thư mục con tên 'data' như sau để khởi tạo việc xử lý dữ liệu:



Cây thư mục trong thư mục Project

Nhắc lại: Hai **folder train** khác nhau. 'train' nằm **chung nhánh** với 'data' chứa dữ liệu thô từ **Kaggle**, 'train' nằm **trong** 'data' trước tiên chỉ chứ 'dogs' và 'cats' là 2 thư mục rỗng.

2. How to use the program

Chọn **code.py** làm **file startup**.

Execute chương trình.

Chương trình khởi động với 6 lựa chọn, cụ thể như sau:

1. Process Data	Xử lý các dữ liệu thô, phân chia các hình ảnh cho các bước kế tiếp.
2. Train	Tạo mới **model** và **train** lại từ đầu, sau khi **train** sẽ ghi đè lên **file checkpoint.h5**.
3. Retrain	Sử dụng **checkpoint.h5** đã có sẵn để tiếp tục **train**.
4. Test	**Test** toàn bộ các hình ảnh trong thư mục 'test', in ra màn hình độ chính xác trung bình hiện tại.
5. Customized Test	Mở giao diện chọn hình ảnh để **test**. Sau khi chọn ảnh, nhấn **Classify Image** để chương trình thực thi việc kiểm tra và in ra màn hình đáp án là chó hay mèo.
6. Exit	Kết thúc vòng lặp chương trình. Có thể thoát an toàn.

QUAN TRỌNG:

Vui lòng thực hiện **Process Data** trước hết để chương trình có thể sử dụng dữ liệu sau khi đã xử lý để train.

II. DATA

1. Distinguishing 3 types of sets

- ### Training set

Training set là tập dữ liệu có gắn nhãn chi tiết và được dùng để **huấn luyện cho mô hình,** giúp cho mô hình xác định được những thông số cần thiết của dữ liệu. Với những gì học được, mô hình được kỳ vọng sẽ đưa ra những dự đoán chính xác cho các bộ dữ liệu mới.

- ### Validation set

Validation set là tập dữ liệu có gắn nhãn, được tách ra một phần từ **Training set** nhưng nó được dùng để kiểm tra độ chính xác của mô hình **trong khi** quá trình huấn luyện diễn ra.

Trong quá trình huấn luyện, mô hình sẽ phân loại **output** của từng **input** từ **Training set** đồng thời tính toán chi phí **loss** và cập nhật giá trị **accuracy**. Nhưng khi lấy dữ liệu của bộ **Validation set** để kiểm thử độ chính xác, các chi phí sẽ không được tính toán và cập nhật.

Do đó, **Validation set** sẽ giúp cho mô hình tránh khỏi trường hợp bị **Overfitting – Mô hình dự đoán chính xác đến mức tối đa với tập Training set, dẫn đến không thể tổng quát hóa và dự đoán chính xác đối với tập Testing set** (Sẽ được đề cập chi tiết hơn ở phần sau)

- ### Testing set

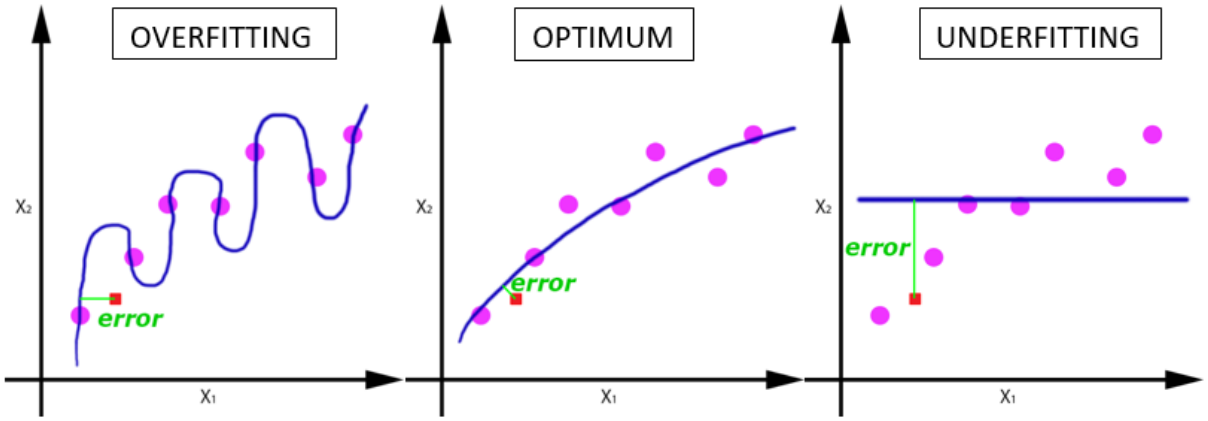
Đối với các dự án thực tế, những dự đoán được mô hình đưa ra chưa thể xác định được độ chính xác ngay tại thời điểm đó (Dự đoán giá cổ phiếu tăng hay giảm, ...).

Do đó, khác với **Training set** và **Validation set**, **Test set** là tập dữ liệu không được gán nhãn và được dùng để **kiểm tra độ chính xác của mô hình sau khi** quá trình huấn luyện diễn ra xong và **trước khi mô hình được triển khai vào các dự án thực tế**.

- **Summary**

Dataset	Update weights	Is labeled	Description
Training set	Yes	Yes	Dùng để huấn luyện mô hình
Validation set	No	Yes	Dùng để kiểm tra độ chính xác của mô hình **trong quá trình huấn luyện**
Test set	No	No	Dùng để kiểm tra độ chính xác của mô hình **sau khi quá trình huấn luyện diễn ra xong**

2. Detecting and preventing the network from being over/underfit



- ## Overfitting problem

Overfitting là hiện tượng mô hình học được quá cụ thể (quá khớp) với bộ dữ liệu trong **Training set** nhưng không dự đoán hoặc phân loại được chính xác bộ dữ liệu mới dẫn đến việc mô hình bị lệch nhiều so với bộ dữ liệu trong **Testing set**.

Phát hiện mô hình bị overfit:

Người xây dựng **model** có thể dự đoán được mô hình có đang bị **overfit** hay không dựa vào giá trị '**accuracy**' và chi phí '**loss'** thông qua việc kiểm tra trên bộ dữ liệu trong **Validation set**. Nếu các chỉ số tìm được của **Validation set** tệ hơn so với chỉ số của **Training set** thì khả năng cao mô hình đang bị **overfitting**.

Thông thường, **overfitting** xảy ra khi dùng mô hình quá phức tạp để mô phỏng **training data**. Điều này đặc biệt xảy ra khi lượng dữ liệu dành cho việc **training** quá nhỏ trong khi độ phức tạp của mô hình quá cao.

Các kỹ thuật giúp tránh bị overfit:

- ### Sử dụng tập Validation set

Như đã đề cập ở mục **Validation set,** sử dụng và phân chia tỷ lệ hợp lý cho tập **Validation set** có thể giúp mô hình tránh được việc bị **overfitting**.

Trong trường hợp dữ liệu quá ít, có thể thực hiện các thay đổi **cropping**, **rotating**, **flipping**, **zooming** trên bộ dữ liệu để gia tăng thêm dữ liệu.

Ngoài ra, kỹ thuật này còn có một bản nâng cấp khác là **K-fold cross-validation** với lượng dữ liệu trong tập **validation** là nhỏ nhưng chất lượng mô hình được đánh giá trên **k** tập **validation** khác nhau.

- ### Đơn giản hóa model

Như đã đề cập ở trên, **overfitting** thường xảy ra khi dùng mô hình quá phức tạp để mô phỏng **training data**. Do đó, việc sử dụng **model** đơn giản hơn - Giảm số lớp khỏi mô hình hoặc số lượng **neural** trong **Networks** cũng có thể giúp cho **model** tránh được việc bị **overfit**.

- ### Dropout

Dropout là một phương pháp tắt ngẫu nhiên các **units** trong **Networks**. Tức là cho các **unit** giá trị bằng không và tính toán lại **forward** và **back propagation** bình thường trong khi **training**. Việc này không những giúp lượng tính toán giảm đi mà còn giảm khả năng bị **overfit**.

- ## Underfitting problem

Trái ngược với **Overfitting**, **Underfitting** là hiện tượng mô hình học được không khớp với bất kì dữ liệu nào vì mô hình không thể học được gì từ bộ dữ liệu.

Thông thường, **underfitting** xảy ra đến từ việc mô hình được sử dụng không phù hợp so với vấn đề đang giải quyết. Ngoài ra, trong trường hợp dữ liệu đầu vào quá phức tạp nhưng mô hình được sử dụng lại quá đơn giản cũng có thể xảy ra **underfitting**.

Còn một trường hợp hiếm gặp, khi dữ liệu đầu vào chưa cung cấp đủ những đặc điểm của vấn đề yêu cầu (Chẳng hạn như quên gắn nhãn cho dữ liệu, ...) thì dĩ nhiên mô hình không thể học được gì từ bộ dữ liệu.

Phát hiện mô hình bị underfit:

Người xây dựng **model** có thể dự đoán được mô hình có đang bị **underfit** hay không dựa vào giá trị '**accuracy**' và chi phí '**loss**' Nếu hai chí phí quá chênh lệch, tức giá trị '**accuracy**' của mô hình quá thấp còn chi phí '**loss**' quá cao thì khả năng cao mô hình đang bị **Underfitting**.

Các kỹ thuật giúp tránh bị underfit:

- ### Tăng cường các thành phần trong model

Như đã đề cập ở trên, việc xảy ra **underfitting** có thể do mô hình không phù hợp với vấn đề đang giải quyết hoặc mô hình quá đơn giản so với dữ liệu đầu vào. Do đó việc gia tăng thêm các lớp **layer**, các số lượng **neuron** trong **Networks** hoặc thậm chí thay đổi luôn cả **model** có thể giải quyết được tình trạng **underfitting**.

- ### Xem và cập nhật lại bộ dữ liệu

Như đã đề cập ở trên, trường hợp dữ liệu đầu vào chưa được cung cấp đủ các đặc điểm cần thiết cũng là nguyên nhân dẫn đến mô hình bị **underfit**. Do đó cần phải xem xét việc cập nhật và làm mới dữ liệu đầu vào khi cảm thấy hiện tượng này đang xảy ra.

- ### Dropout

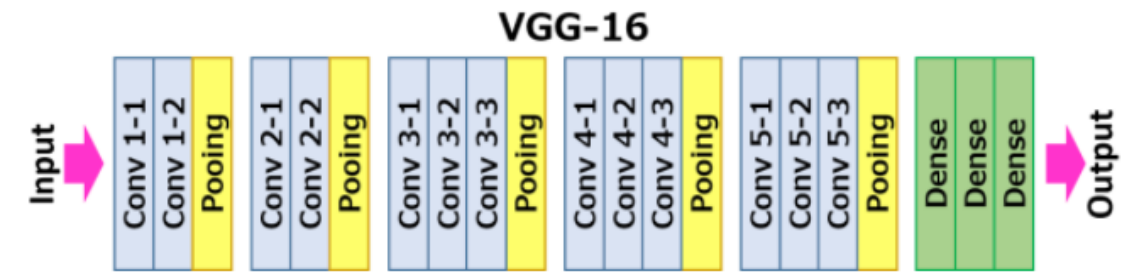
Xem trong mục Overfitting.

III. MODEL

1. Describe the components of the model you use and why you chose it

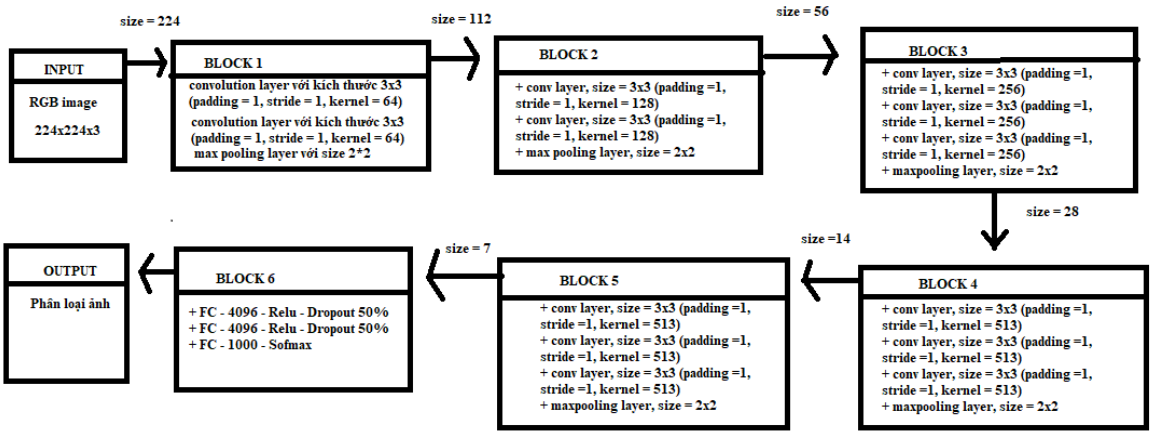
- **The components of the model**

Ở đây, mô hình được sử dụng là **VGG16**, đây là một mô hình **convolutional neural network**. Cấu trúc của mô hình như sau:



VGG16

mô hình gồm dữ liệu đầu vào, dữ liệu đầu ra, **6 block** với **21 lớp**, mỗi **block** được ngăn với nhau bởi một lớp **maxpooling**.



tất cả các lớp **convolution** có **activation** là **ReLu,** càng qua các **BLOCK** thì size ảnh càng giảm nhưng **kernel** tăng, và vào **BLOCK** cuối thì dữ liệu mới được **flatten** và vào **fully connected (FC)**

- **Reason to choose**

VGG16 là convolutional neural network model đạt độ chính xác 92.7%, top-5 test trong dữ liệu ImageNet gồm 14 triệu hình ảnh thuộc 1000 lớp khác nhau.

2. Point out the advantages and disadvantages of the model

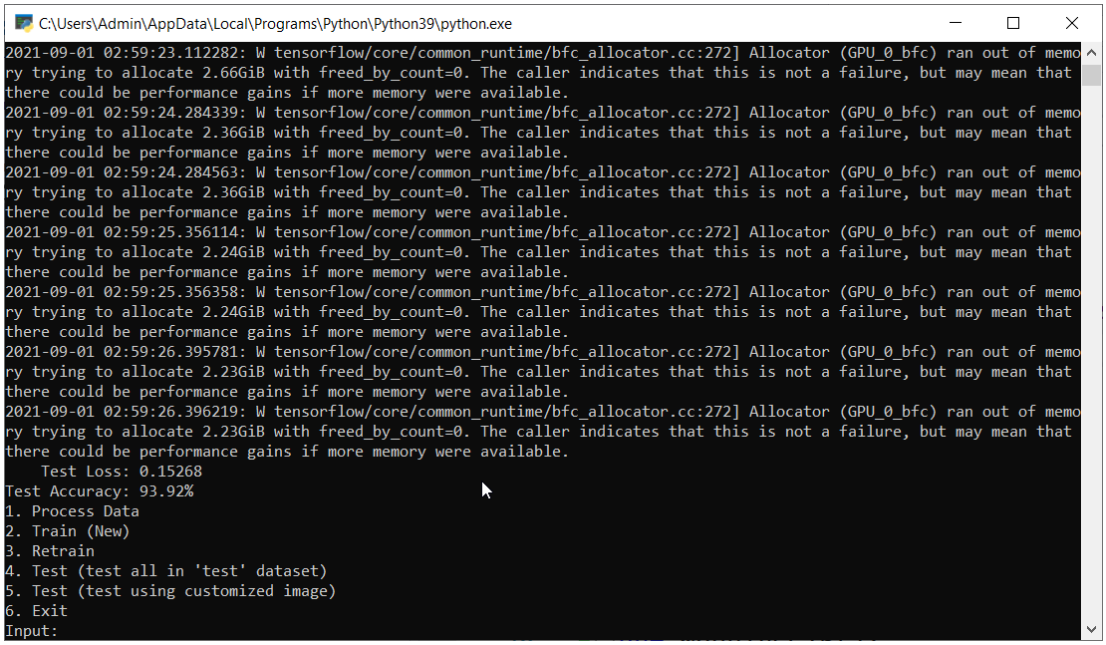
- **Advantage**

Độ chính xác cao.

- **Disadvantage**

Có số lượng **parameter** quá lớn, nên quá trình train lâu, làm cho model khó chạy trong giai đoạn suy luận kết quả.

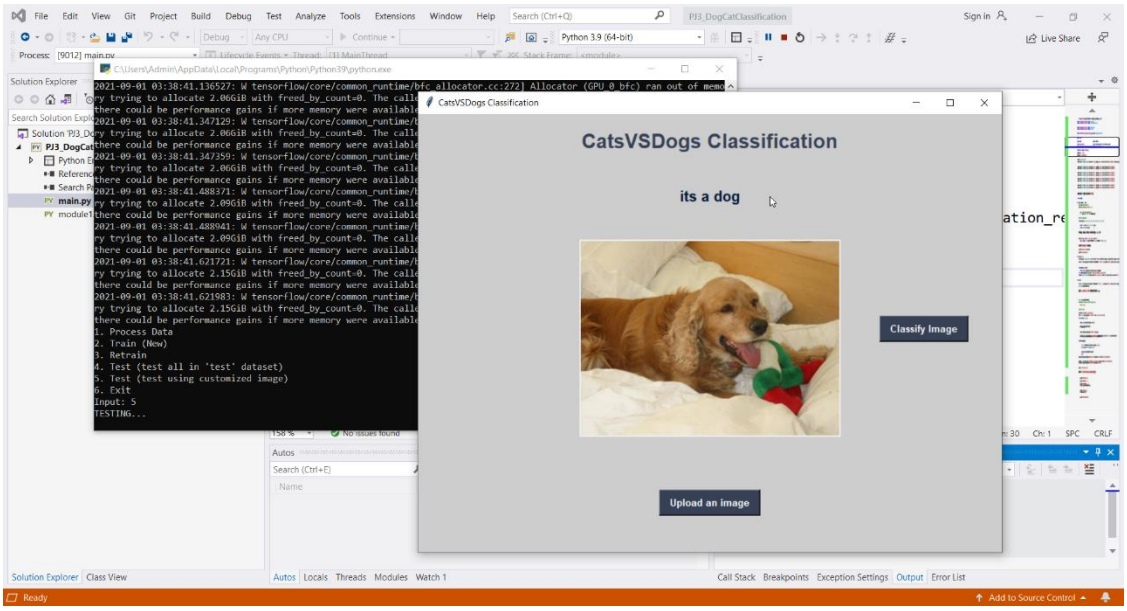
IV. RESULT



Kết quả sau khi test toàn bộ ảnh trong thư mục 'test'

Model trên đã được train **3 lần**, mỗi lần **5 epochs**. **Accuracy** nhận được sau khi test toàn bộ file trong thư mục đạt **gần 94%**. Thời gian train mỗi epoch vào khoảng **10~20 phút** khi sử dụng GPU. Toàn bộ thời gian train rơi vào khoảng **3~4 giờ** đồng hồ.
Để thấy chi tiết, có thể chọn option 5 để tự kiểm định.

Khi hình ảnh khá rõ ràng, chương trình có thể dự đoán chính xác.



Tuy nhiên, khi chưa đủ rõ, chương trình có vẻ vẫn ...chưa hoạt động ổn lắm.

Con gì đây?

Một số phương án có thể dùng để cải thiện độ chính xác hiện tại:

- Retrain nhiều lần hơn, tuy nhiên có thể gây overfitting.
- Sử dụng kiến trúc mới hơn (VGG19)

V. REFERENCES

1. DATA

- **Distinguish 3 types of training set, validation set, test set:**

Deep Lizard: Train, Test, & Validation Sets explained
Tarang Shah: About Train, Validation and Test Sets in Machine Learning
Songhao Wu: What is the difference between Test and Validation sets? | by Songhao Wu | CodeX | Medium

- **Describe detect and prevent overfitting and underfitting problem:**

Vũ Hữu Tiệp: Machine Learning cơ bản (machinelearningcoban.com)
Deep Lizard: Overfitting in a Neural Network explained
Deep Lizard: Underfitting in a Neural Network explained

2. MODEL

- **Architecture and Component of VGG16**

Muneeb ul Hassan - Neuro Hive: VGG16 model
Phạm Đình Khánh - Khoa học dữ liệu
Nguyễn Thanh Tuấn - nttuan8: Bài 5 - giới thiệu xử lý ảnh, Bài 6 - CNN là gì

- **Implementation of VGG16**

Lorenzo Baraldi - Github: code VGG16
Mathworks: VGG16 CNN
Rohit Thakur - Towardsdatascience: VGG16 implementation in keras
Nguyễn Phúc Lương - Viblo: ứng dụng CNN trong bài toán phân loại ảnh

- **Create a Keras Model**

Py image Search: 3 ways to create a keras model