

TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG



BÁO CÁO BÀI TẬP LỚN MÔN:
LINUX và Phần mềm mã nguồn mở

Tìm hiểu thư viện mã nguồn mở TensorFlow

Sinh viên thực hiện:

Hoàng Hải My	- 20176102
Nguyễn Duy Khánh	- 20187174
Mai Trần Duy	- 20198223
Nông Trần Bình Minh	- 20187187
Lê Thành Chính	- 20198209

Giáo viên hướng dẫn: TS. Trần Hải Anh

Hà Nội, tháng 1 năm 2022

MỤC LỤC

MỤC LỤC	2
LỜI NÓI ĐẦU	4
PHÂN CÔNG THÀNH VIÊN TRONG NHÓM	5
CHƯƠNG 1. GIỚI THIỆU BỘ THƯ VIỆN TENSORFLOW	6
1.1 TensorFlow là gì?	6
1.2 Lịch sử ra đời TensorFlow	7
1.3 Kiến trúc của TensorFlow	7
1.4 Cách TensorFlow hoạt động	8
1.5 Lợi ích của TensorFlow	9
1.6 Các khái niệm trong Tensorflow	10
1.6.1 Tensor	10
1.6.2 Graph	11
1.6.3 Node	11
1.6.4 Rank	12
1.6.5 Shape	12
1.6.6 Toán tử - Operator	12
1.6.7 DType	12
CHƯƠNG 2. CÀI ĐẶT THƯ VIỆN TENSORFLOW TRÊN UBUNTU 18.04 LTS	13
2.1. Fetch package mới nhất từ Internet.	13
2.2. Cài đặt Python 3	14
2.3. Cài đặt pip	15
2.4. Sử dụng pip cài đặt thư viện TensorFlow.	16
CHƯƠNG 3. SỬ DỤNG TENSORFLOW XÂY DỰNG MÔ HÌNH HỌC MÁY TRÊN DOCKER	17
1.1 Cài đặt Docker	17
1.2 Cài đặt Tensorflow trên Docker và training mô hình	17

CHƯƠNG 4. SỬ DỤNG TENSORFLOW XÂY DỰNG MÔ HÌNH HỌC

MÁY TRÊN GOOGLE COLAB	21
4.1. Xây dựng bài toán.....	21
4.2. Chuẩn bị dữ liệu.....	21
4.3. Mã nguồn của chương trình.....	21
KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN	28
TÀI LIỆU THAM KHẢO	29

LỜI NÓI ĐẦU

Ngày nay, với sự phát triển nhanh chóng của công nghệ là sự bùng nổ của lĩnh vực trí tuệ nhân tạo. Tensorflow, một thư viện phần mềm mã nguồn mở dành cho học máy đã được phát triển bởi Google. Bộ thư viện này cho phép tạo ra các biểu đồ luồng dữ liệu để xây dựng mô hình học máy và học sâu một cách thuận tiện, dễ dàng và hiệu quả. Vì vậy, thông qua môn học Linux và phần mềm mã nguồn mở, nhóm chúng em đã lựa chọn đề tài “**Tìm hiểu thư viện mã nguồn mở TensorFlow**”, và sử dụng bộ thư viện mã nguồn mở này để xây dựng và huấn luyện hai mô hình học máy. Một mô hình được xây dựng và huấn luyện trên Docker và một mô hình được huấn luyện trên Google Colab.

Bài báo cáo được chia thành 4 phần:

Phần 1: Giới thiệu bộ thư viện TensorFlow

Phần 2: Cài đặt thư viện TensorFlow trên Ubuntu 18.04 LTS

Phần 3: Sử dụng TensorFlow xây dựng mô hình học máy trên Docker

Phần 4: Sử dụng TensorFlow xây dựng mô hình học máy trên Google Colab

Hà Nội, tháng 1 năm 2022

Nhóm TENS

My, Duy, Minh, Chinh, Khánh

PHÂN CÔNG THÀNH VIÊN TRONG NHÓM

STT	Họ tên	MSSV	Công việc đóng góp	Mức độ hoàn thành
1	Hoàng Hải My	20176102	<ul style="list-style-type: none">- Phân chia công việc trong nhóm- Theo dõi tiến độ nhóm- Viết báo cáo chương 4- Căn chỉnh format báo cáo- Xây dựng mô hình nhận diện cảm xúc khuôn mặt	100%
2	Nguyễn Duy Khánh	20187174	<ul style="list-style-type: none">- Cài đặt Docker- Tìm hiểu về Docker và chạy code trên Docker- Viết báo cáo chương 3- Xây dựng mô hình phân loại hoa	100%
3	Mai Trần Duy	20198223	<ul style="list-style-type: none">- Tìm hiểu lý thuyết và cài đặt TensorFlow về Ubuntu- Viết báo cáo chương 1- Làm slide- Xây dựng mô hình nhận diện cảm xúc khuôn mặt	100%
4	Lê Thành Chính	20198209	<ul style="list-style-type: none">- Tìm hiểu lý thuyết và cài đặt TensorFlow về Ubuntu- Viết báo cáo chương 2- Làm slide- Xây dựng mô hình phân loại hoa	100%
5	Nông Trần Bình Minh	20187187	<ul style="list-style-type: none">- Tổng hợp báo cáo cả nhóm và lập báo cáo chung- Xây dựng mô hình nhận diện cảm xúc khuôn mặt	100%

CHƯƠNG 1. GIỚI THIỆU BỘ THƯ VIỆN TENSORFLOW

1.1 TensorFlow là gì?

Với sự bùng nổ của lĩnh vực Trí Tuệ Nhân Tạo – A.I, tong thập kỷ vừa qua, học máy (Machine Learning) và học sâu (Deep Learning) cũng phát triển theo cùng. Ở thời điểm hiện tại, TensorFlow chính là thư viện mã nguồn mở cho machine learning nổi tiếng nhất thế giới, được phát triển bởi các nhà nghiên cứu từ Google. Việc hỗ trợ mạnh mẽ các phép toán học để tính toán trong Machine Learning và Deep Learning đã giúp việc tiếp cận các bài toán trở nên đơn giản, nhanh chóng và tiện lợi hơn nhiều.



Hình 1: Logo TensorFlow

Các hàm được dựng sẵn trong thư viện cho từng bài toán cho phép TensorFlow xây dựng được nhiều neural network. Nó còn cho phép bạn tính toán song song trên nhiều máy tính khác nhau, thậm chí trên nhiều CPU, GPU trong cùng 1 máy hay tạo ra các dataflow graph – đồ thị luồng dữ liệu để dựng

nên các model. Nếu bạn muốn chọn con đường sự nghiệp trong lĩnh vực A.I. này, nắm rõ những điều cơ bản của TensorFlow thực sự rất quan trọng.

1.2 Lịch sử ra đời TensorFlow

Vài năm trước, khi phải xử lý lượng dữ liệu khổng lồ, deep learning bắt đầu cho thấy hiệu năng vượt trội so với tất cả các thuật toán machine learning khác. Google sớm nhận ra tiềm năng này và nghĩ rằng họ nên sử dụng deep neural network để cải thiện các dịch vụ của mình, trong đó có:

- Gmail
- Hình ảnh
- Google search engine

Vì vậy nên họ xây dựng một framework có tên là TensorFlow để các nhà nghiên cứu cũng như lập trình viên có thể làm việc cùng nhau trên model A.I. Một khi đã được phát triển hoàn chỉnh, rất nhiều người đã có thể sử dụng được TensorFlow.

Ra mắt lần đầu vào cuối năm 2015, phiên bản TensorFlow ổn định cuối cùng cũng xuất hiện vào năm 2017. Là mã nguồn mở dưới sự cho phép của Apache Open Source, giờ đây chúng ta có thể sử dụng, điều chỉnh và tái đóng góp phiên bản được điều chỉnh đó, đổi lại không cần phải trả bất cứ đồng nào cho Google.

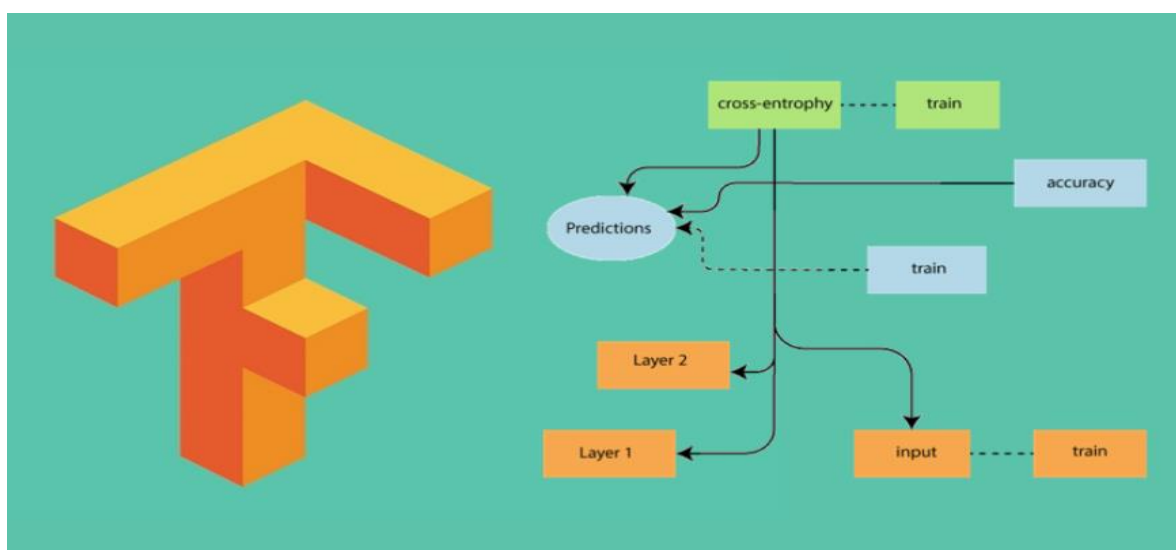
1.3 Kiến trúc của TensorFlow

Kiến trúc TensorFlow hoạt động được chia thành 3 phần:

- Tiền xử lý dữ liệu
- Xây dựng model
- Huấn luyện(train) và ước tính model

1.4 Cách TensorFlow hoạt động

TensorFlow cho phép các lập trình viên tạo ra dataflow graph, cấu trúc mô tả làm thế nào dữ liệu có thể di chuyển qua một biểu đồ, hay một sê-ri các node đang xử lý. Mỗi node trong đồ thị đại diện một operation toán học, và mỗi kết nối hay edge giữa các node là một mảng dữ liệu đa chiều, hay còn được gọi là ‘tensor’.



Hình 2: Cấu tạo hệ thống

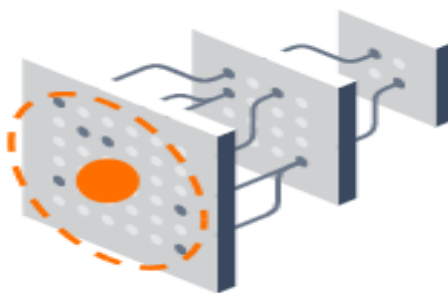
TensorFlow cung cấp tất cả những điều này cho lập trình viên theo phương thức của ngôn ngữ Python. Vì Python khá dễ học và làm việc, ngoài ra còn cung cấp nhiều cách tiện lợi để ta hiểu được làm thế nào các high-level abstractions có thể kết hợp cùng nhau. Node và tensor trong TensorFlow là các đối tượng Python, và các ứng dụng TensorFlow bản thân chúng cũng là các ứng dụng Python.

Các operation toán học thực sự thì không được thi hành bằng Python. Các thư viện biến đổi có sẵn thông qua TensorFlow được viết bằng các binary C++ hiệu suất cao. Python chỉ điều hướng lưu lượng giữa các phần và cung cấp các high-level abstraction lập trình để nối chúng lại với nhau.

TensorFlow 2.0, được ra mắt vào tháng 10 năm 2019, cải tiến framework theo nhiều cách dựa trên phản hồi của người dùng, để dễ dàng và hiệu quả hơn khi làm việc cùng nó (ví dụ: bằng cách sử dụng các Keras API liên quan đơn giản cho việc train model). Train phân tán để chạy hơn nhờ vào API mới và sự hỗ trợ cho TensorFlow Lite cho phép triển khai các mô hình trên khá nhiều nền tảng khác nhau. Tuy nhiên, nếu đã viết code trên các phiên bản trước đó của TensorFlow thì bạn phải viết lại, đôi lúc một ít, đôi lúc cũng khá đáng kể, để tận dụng tối đa các tính năng mới của TensorFlow 2.0.

1.5 Lợi ích của TensorFlow

Lợi ích dễ thấy nhưng quan trọng nhất mà TensorFlow cung cấp cho việc lập trình machine learning chính là abstraction. Thay vì phải đối phó với những tình huống rườm rà từ việc thực hiện triển khai các thuật toán, hay tìm ra cách hợp lý để chuyển output của một chức năng sang input của một chức năng khác, giờ đây bạn có thể tập trung vào phần logic tổng thể của một ứng dụng hơn. TensorFlow sẽ chăm sóc phần còn lại thay cho bạn.



Hình 3: Xây dựng mô hình dễ dàng

Ngoài ra TensorFlow còn cung cấp các tiện ích bổ sung cho các lập trình viên cần debug cũng như giúp bạn tự suy xét các ứng dụng TensorFlow. Chế độ eager execution cho phép bạn đánh giá và sửa đổi từng operation của biểu đồ một cách riêng biệt và minh bạch, thay vì phải dựng toàn bộ biểu đồ dưới dạng

một đối tượng độc lập vốn khá mơ hồ hay phải đánh giá chung tổng thể. Cuối cùng, một tính năng khá độc đáo của TensorFlow là TensorBoard. TensorBoard cho phép bạn quan sát một cách trực quan những gì TensorFlow đang làm.



Hình 4: Tự do sáng tạo

TensorFlow còn có nhiều cải tiến từ sự hậu thuẫn từ các ekip thương mại hạng A tại Google. Google không những tiếp lửa cho tiến độ nhanh chóng cho sự phát triển đằng sau dự án, mà còn tạo ra nhiều phục vụ độc đáo xung quanh TensorFlow để nó dễ dàng deploy và sử dụng: như silicon TPU mình đã nói ở trên để tăng tốc hiệu suất đám mây Google, một online hub cho việc chia sẻ các model được tạo với framework, sự hiện diện của in-browser và gắn gửi với mobile của framework, và nhiều hơn thế nữa...

1.6 Các khái niệm trong Tensorflow

1.6.1 Tensor

Tên của TensorFlow được đưa ra trực tiếp là nhờ vào framework cốt lõi của nó: Tensor. Trong TensorFlow, tất cả các tính toán đều liên quan tới các tensor. Một tensor là một vector hay ma trận của n-chiều không gian đại diện cho tất cả loại dữ liệu. Tất cả giá trị trong một tensor chứa đựng loại dữ liệu giống hệt nhau với một shape đã biết (hoặc đã biết một phần). Shape của dữ liệu chính là chiều của ma trận hay mảng.

Một tensor có thể được bắt nguồn từ dữ liệu input hay kết quả của một tính toán. Trong TensorFlow, tất cả các hoạt động được tiến hành bên trong một graph – biểu đồ. Biểu đồ là một tập hợp tính toán được diễn ra liên tiếp. Mỗi operation được gọi là một op node (operation node) và được kết nối với nhau.

Biểu đồ phát thảo các op và kết nối giữa các node. Tuy nhiên, nó không hiển thị các giá trị. Phần edge của các node chính là tensor, một cách để nhập operation với dữ liệu.

1.6.2 Graph

TensorFlow sử dụng framework dạng biểu đồ. Biểu đồ tập hợp và mô tả tất cả các chuỗi tính toán được thực hiện trong quá trình training. Biểu đồ cũng mang rất nhiều lợi thế:

- Nó được làm ra để chạy trên nhiều CPU hay GPU, ngay cả các hệ điều hành trên thiết bị điện thoại.
- Tính di động của biểu đồ cho phép bảo toàn các tính toán để bạn sử dụng ngay hay sau đó. Biểu đồ có thể được lưu lại để thực thi trong tương lai.
- Tất cả tính toán trong biểu đồ được thực hiện bằng cách kết nối các tensor lại với nhau. Một tensor có một node và một edge. Node mang operation toán học và sản xuất các output ở đầu cuối. Các edge giải thích mối quan hệ input/output giữa các node.

1.6.3 Node

Vì Tensorflow mô tả lại dòng chảy của dữ liệu thông qua graph nên mỗi một điểm giao cắt trong graph thì được gọi là Node. Vì các Node chính là điểm đại diện cho việc thay đổi của dữ liệu nên việc lưu trữ lại tham chiếu của các Node này là rất quan trọng.

1.6.4 Rank

Rank là số bậc của tensor. Ví dụ Tensor = [1] thì có rank = 1, Tensor = [[3,4],[5,6]] thì sẽ có rank = 2. Việc phân rank này khá quan trọng vì nó đồng thời cũng giúp phân loại dữ liệu của Tensor. Khi các rank đặc biệt cụ thể, Tensor có những tên gọi riêng như sau:

- Scalar: Khi Tensor có rank bằng 0
- Vector: Vector là một Tensor rank 1. .
- Matrix: Đây là một Tensor rank 2 hay mảng hai chiều theo khái niệm của Python
- N-Tensor: Khi rank của Tensor tăng lên lớn hơn 2, chúng được gọi chung là N-Tensor.

1.6.5 Shape

Shape là một tuple có số chiều bằng với rank của Tensor tương ứng dùng để mô tả lại cấu trúc của Tensor đó.

Ví dụ: Tensor = [[[1,1,1],[178,62,74]]] sẽ có Shape = (1,2,3), Tensor = [[1,1,1],[178,62,74]] sẽ có Shape = (2,3).

1.6.6 Toán tử - Operator

Được viết tắt là op, khái niệm Operator là toán tử được dùng để thực thi Tensor tại node đó. Các toán tử này có thể là Hằng số, Biến số, Phép cộng, Phép nhân.

1.6.7 DType

Đây là kiểu dữ liệu của các phần tử trong Tensor. Vì một Tensor chỉ có duy nhất một thuộc tính DType nên từ đó cũng suy ra là chỉ có duy nhất một kiểu DType duy nhất cho toàn bộ các phần tử có trong Tensor hiện tại.

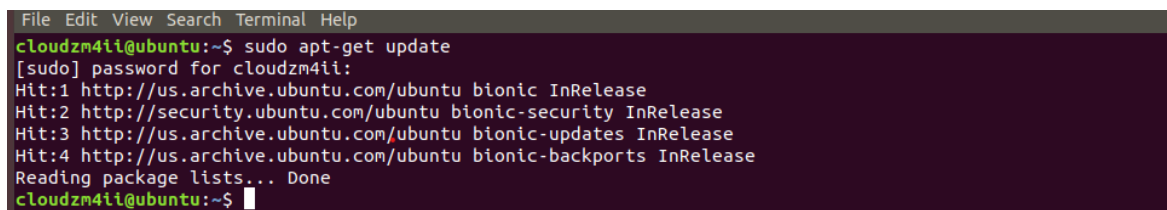
CHƯƠNG 2. CÀI ĐẶT THƯ VIỆN TENSORFLOW TRÊN UBUNTU 18.04 LTS

2.1. Fetch package mới nhất từ Internet.

Advanced Packaging Tool, hay APT, là phần mềm miễn phí dùng để quản lý việc cài đặt phần mềm trên Linux. APT làm đơn giản các thủ tục quản lý phần mềm trên các máy tính tựa Unix bằng cách tự động hóa việc tải về, cấu hình và cài đặt các gói phần mềm, cả ở dạng biên dịch sẵn (dạng binary) hoặc biên dịch mã nguồn.

APT ban đầu được thiết kế như là một giao diện cho dpkg để làm việc với các gói .deb của Debian, nhưng nó cũng đã được thay đổi để có thể làm việc với hệ thống RPM thông qua apt-rpm. Dự án Fink đã chuyển APT lên hệ điều hành Mac OS X với một số chức năng quản lý gói, và APT cũng đã có trên OpenSolaris (bao gồm bản phân phối Nexenta OS).

- Câu lệnh sử dụng: Sudo apt-get update



```
File Edit View Search Terminal Help
cloudzm4ii@ubuntu:~$ sudo apt-get update
[sudo] password for cloudzm4ii:
Hit:1 http://us.archive.ubuntu.com/ubuntu bionic InRelease
Hit:2 http://security.ubuntu.com/ubuntu bionic-security InRelease
Hit:3 http://us.archive.ubuntu.com/ubuntu bionic-updates InRelease
Hit:4 http://us.archive.ubuntu.com/ubuntu bionic-backports InRelease
Reading package lists... Done
cloudzm4ii@ubuntu:~$
```

Hình 5: Sử dụng câu lệnh sudo apt-get update

- Kiểm tra các package của python.

- Câu lệnh sử dụng: Sudo apt-cache search python | grep ^python3

```
cloudzm4i@ubuntu:~$ sudo apt-cache search python | grep ^python3
python3 - interactive high-level object-oriented language (default python3 version)
python3-alabaster - Configurable sidebar-enabled Sphinx theme (Python 3)
python3-alembic - lightweight database migration tool for SQLAlchemy - Python 3.x
python3-all - package depending on all supported Python 3 runtime versions
python3-all-dbg - package depending on all supported Python 3 debugging packages
python3-all-dev - package depending on all supported Python 3 development packages
python3-amqp - Low-level AMQP client (Python3 version)
python3-apparmor - AppArmor Python3 utility library
python3-apport - Python 3 library for Apport crash report handling
python3-apt - Python 3 interface to libapt-pkg
python3-apt-dbg - Python 3 interface to libapt-pkg (debug extension)
python3-aptdaemon - Python 3 module for the server and client of aptdaemon
python3-aptdaemon.gtk3widgets - Python 3 GTK+ 3 widgets to run an aptdaemon client
python3-asn1crypto - Fast ASN.1 parser and serializer (Python 3)
python3-astroid - rebuild a new abstract syntax tree from Python's AST (Python3)
python3-attr - Attributes without boilerplate (Python 3)
python3-automat - Self-service finite-state machines for the programmer on the go
python3-babel - tools for internationalizing Python applications - Python 3.x
python3-blinker - fast, simple object-to-object and broadcast signaling library
python3-brlapi - Braille display access via BRLTTY - Python3 bindings
python3-bs4 - error-tolerant HTML parser for Python 3
python3-bsddb3 - Python interface for Berkeley DB (Python 3.x)
python3-bsddb3-dbg - Python interface for Berkeley DB (debug extension, Python 3.x)
python3-bson - Python3 implementation of BSON for MongoDB
python3-bson-ext - C-coded extension to the python3-bson package
python3-cairo - Python3 bindings for the Cairo vector graphics library
python3-cairo-dbg - Python3 bindings for the Cairo vector graphics library (debug extension)
python3-cairo-dev - Python3 cairo bindings: development files
python3-cairo-doc - Python 3 cairo bindings: documentation files
python3-certifi - root certificates for validating SSL certs and verifying TLS hosts (python3)
python3-cffi-backend - Foreign Function Interface for Python 3 calling C code - runtime
python3-cffi-backend-dbg - Foreign Function Interface for Python 3 calling C code (Debug version)
python3-chardet - universal character encoding detector for Python3
python3-checkbox-ng - PlainBox based test runner (Python 3 library)
python3-checkbox-ng-doc - PlainBox based test runner (documentation)
python3-checkbox-support - collection of Python modules used by PlainBox providers
python3-click - Simple wrapper around optparse for powerful command line utilities - Python 3.x
python3-cliff - command line interface formulation framework - Python 3.x
python3-cmd2 - enhanced Python cmd module - Python 3.x
python3-colorama - Cross-platform colored terminal text in Python - Python 3.x
python3-commandnotfound - Python 3 bindings for command-not-found.
python3-configobj - simple but powerful config file reader and writer for Python 3
```

Hình 6: Kiểm tra các package của Python3

2.2. Cài đặt Python 3

Cấu hình để chuyển đổi giữa các Python3 Version, với Ubuntu 18.04LTS đang mặc định là Python 3.6:

- Câu lệnh sử dụng: Sudo apt-get install python 3

```

cloudzm4i@ubuntu:~$ sudo apt-get install python 3.9
Reading package lists... Done
Building dependency tree
Reading state information... Done
Note, selecting 'libllvm3.9v4' for regex '3.9'
Note, selecting 'libghc-wai-handler-launch-dev-3.0.2.3-973be' for regex '3.9'
Note, selecting 'libghc-control-monad-loop-dev-0.1-71349' for regex '3.9'
Note, selecting 'libghc-presburger-prof-1.3.1-43796' for regex '3.9'
Note, selecting 'llvm-3.9' for regex '3.9'
Note, selecting 'libghc-blaze-svg-prof-0.3.6.1-4b349:i386' for regex '3.9'
Note, selecting 'python-rfc3339' for regex '3.9'
Note, selecting 'libghc-uri-bytestring-prof-0.2.3.3-9e9d6' for regex '3.9'
Note, selecting 'libghc-web-routes-prof-0.27.12-359a6:i386' for regex '3.9'
Note, selecting 'libghc-microlens-platform-dev-0.3.9.0-e66d5:i386' for regex '3.9'
Note, selecting 'libghc-cryptonite-conduit-prof-0.2.2-6b309:i386' for regex '3.9'
Note, selecting 'libghc-brick-prof-0.18-389a7' for regex '3.9'
Note, selecting 'clang-3.9-examples' for regex '3.9'
Note, selecting 'libghc-http-prof-4000.3.9-d5180' for regex '3.9'
Note, selecting 'libghc-bmp-dev-1.2.6.3-43595:i386' for regex '3.9'
Note, selecting 'lldb-3.9-dev' for regex '3.9'
Note, selecting 'clang-3.9' for regex '3.9'
Note, selecting 'libghc-finite-field-prof-0.8.0-38985' for regex '3.9'
Note, selecting 'clang-3.9-doc' for regex '3.9'
Note, selecting 'libghc-hinotify-dev-0.3.9-71a93:i386' for regex '3.9'
Note, selecting 'libghc-fb-dev-1.1.1-df3f9' for regex '3.9'
Note, selecting 'clang-include-fixer-3.9' for regex '3.9'
Note, selecting 'python-clang-3.9' for regex '3.9'
Note, selecting '389-ds-console-doc' for regex '3.9'
Note, selecting 'libghc-zlib-bindings-dev-0.1.1.5-b3092' for regex '3.9'
Note, selecting 'libghc-hopenpgp-dev-2.5.5-1a309:i386' for regex '3.9'
Note, selecting 'llvm-3.9-dev' for regex '3.9'
Note, selecting 'libghc-microlens-platform-dev-0.3.9.0-e24f6' for regex '3.9'
Note, selecting 'libghc-microlens-platform-prof-0.3.9.0-e66d5:i386' for regex '3.9'
Note, selecting 'llvm-3.9-doc' for regex '3.9'
Note, selecting 'libghc-flexible-defaults-prof-0.0.1.2-3c998:i386' for regex '3.9'
Note, selecting 'libghc-operational-prof-0.2.3.5-53a9b' for regex '3.9'
Note, selecting 'libllvm3.9v4-dbg' for regex '3.9'
Note, selecting '389-ds-base-libs' for regex '3.9'
Note, selecting 'nvidia-349-updates' for regex '3.9'
Note, selecting 'libghc-authenticate-oauth-dev-1.6-33d9a:i386' for regex '3.9'
Note, selecting 'libghc-memory-prof-0.14.11-2319e:i386' for regex '3.9'
Note, selecting 'libghc-readargs-prof-1.2.3-90a3b' for regex '3.9'
Note, selecting 'libghc-fb-prof-1.1.1-df3f9' for regex '3.9'
Note, selecting 'libghc-abstract-par-prof-0.3.3-13090:i386' for regex '3.9'
Note, selecting 'libghc-snap-server-dev-1.0.3.3-9e27b' for regex '3.9'
Note, selecting 'libghc-microlens-ghc-prof-0.4.8.0-213e9' for regex '3.9'
Note, selecting 'libghc-debian-prof-3.93.2-c01dd:i386' for regex '3.9'

```

Hình 7: Cài đặt Python3

2.3. Cài đặt pip

PIP là một trình quản lý thư viện cho Python, viết tắt của từ Preferred Installer Program. Đây là một tiện ích dòng lệnh cho phép bạn cài đặt, cài đặt lại hoặc gỡ cài đặt các gói PyPI bằng một dòng lệnh đơn giản và dễ hiểu: pip.

- Câu lệnh sử dụng: Sudo apt-get install python3-pip

TENS – Tìm hiểu thư viện mã nguồn mở TensorFlow

[illegible]

Hình 8: Cài đặt pip

2.4. Sử dụng pip cài đặt thư viện TensorFlow.

```

C:\msdcs\4\ubuntu>python3 -m pip install tensorflow
Collecting tensorflow
  Downloading https://files.pythonhosted.org/packages/4de/90fb2e0412ae996b4432885f677ad6241c08bcc5fe7724d69/tensorflow-1.14.0-cp36-cp36m-manylinux_x86_64.whl (109.2MB)
100% |#####| 109.2MB 3.8/s
Collecting tensorflow-estimator<1.15.0,>=1.14.0rc0 (from tensorflow)
  Downloading https://files.pythonhosted.org/packages/3cd/52180a5b1ca1f067fbc8119341b0ae21a07156911132e0e71bfed0518d/tensorflow_estimator-1.14.0-py2.py3-none-any.whl (488kB)
100% |#####| 491kB 915kB/s
Collecting astor>=0.6.0 (from tensorflow)
  Downloading https://files.pythonhosted.org/packages/c3/88/7eef84f48fa04fbd0730e02dcea4ba6c381b7ac1420856cd8cc0a39/astor-0.8.1-py2.py3-none-any.whl
Collecting keras-preprocessing>=1.0.5 (from tensorflow)
  Downloading https://files.pythonhosted.org/packages/79/4c/7c3275a01e2ef9368a892926ab9323bb13d55794881e35734862b378a7/Keras_Preprocessing-1.1.2-py2.py3-none-any.whl (42kB)
100% |#####| 51kB 4.5MB/s
Collecting absl-py>=0.7.0 (from tensorflow)
  Downloading https://files.pythonhosted.org/packages/2c/3e/03e913df3af430ee32e41221b294e37952e06acc96781c417ac25d400324/absl_py-1.0.0-py3-none-any.whl (126kB)
100% |#####| 133kB 2.5MB/s
Collecting keras-applications>=1.0.6 (from tensorflow)
  Downloading https://files.pythonhosted.org/packages/77/c3/19762fdcfce2877ae91082ed6342d71b28fbfd9dea3d2f96a882ce099b03f/Keras_Applications-1.0.8-py3-none-any.whl (50kB)
100% |#####| 51kB 2.8MB/s
Collecting grpcio>=1.8.6 (from tensorflow)
  Downloading https://files.pythonhosted.org/packages/c6/bf/57cd38ff3ac80f47cbe50618fe4502f90b41a56f5d9e248ee574e14687/grpcio-1.43.0.tar.gz (21.5MB)
100% |#####| 21.5MB 27kB/s
Collecting six>=1.10.0 (from tensorflow)
  Downloading https://files.pythonhosted.org/packages/d9/5a/7ec31adbe875f2abb91bd04c2dc52792b5a01506781dbcf7291daf11/six-1.16.0-py2.py3-none-any.whl
Collecting termcolor>=1.1.0 (from tensorflow)
  Downloading https://files.pythonhosted.org/packages/8a/4b/470be15647d0eb9f10e2a511bf3fbb8c1e0b14e9e4fab46173aa70f981/termcolor-1.1.0.tar.gz
Collecting tensorboard<1.15.0,>=1.14.0 (from tensorflow)
  Downloading https://files.pythonhosted.org/packages/91/2d/2ed63349a078cd9c8a9ba50ebd50123daf18cfbca1492f9084169b89d9/tensorboard-1.14.0-py3-none-any.whl (3.1MB)
100% |#####| 3.2MB 29MB/s
Collecting wheel>=0.26 (from tensorflow)
  Downloading https://files.pythonhosted.org/packages/27/d6/003e593296a85fde0d16ed962795b2f8709c3ee2bca4fd0fe55c6d00/wheel-0.37.1-py2.py3-none-any.whl
  Downloading https://files.pythonhosted.org/packages/a3/de/c648ef683192e62cc03f40b19eeda382c49b5baf34d88b931c4c74ac/google_pasta-0.2.0-py3-none-any.whl (57kB)
100% |#####| 61kB 875kB/s
Collecting numpy>=1.14.5 (from tensorflow)
  Downloading https://files.pythonhosted.org/packages/45/b2/6c7545bb7a3875446304c70680a4a0d947328125d81bf12beaa092c3ae3/numpy-1.19.5-cp36-cp36m-manylinux_x86_64.whl (13.4MB)
100% |#####| 13.4MB 44kB/s
Collecting protobuf>=3.6.1 (from tensorflow)
  Downloading https://files.pythonhosted.org/packages/1c/12/7479e0e493198416269bfaa05cbb2fc23d7f6ee1ab514c6f4cde50a31/protobuf-3.19.1-py2.py3-none-any.whl (162kB)
100% |#####| 163kB 2.9MB/s
Collecting wrapt>=1.11.1 (from tensorflow)
  Downloading https://files.pythonhosted.org/packages/e2/0f/89c3c2d8ba0e709a34d71507a78be443e229f1321d1e154c70f44150c/wrapt-1.13.3-cp36-cp36m-manylinux_x86_64-manylinux_2_12_x86_64-manylinux2010_x86_64.whl (78kB)
100% |#####| 81kB 3.2MB/s
Collecting gast>=0.2.0 (from tensorflow)
  Downloading https://files.pythonhosted.org/packages/5f/1c/b59500a088c3d9d061c5ca62b9df5e0964764472af4ed82a182958a0922c5/gast-0.5.3-py3-none-any.whl

```

Hình 9: Cài đặt thư viện TensorFlow

CHƯƠNG 3. SỬ DỤNG TENSORFLOW XÂY DỰNG MÔ HÌNH HỌC MÁY TRÊN DOCKER

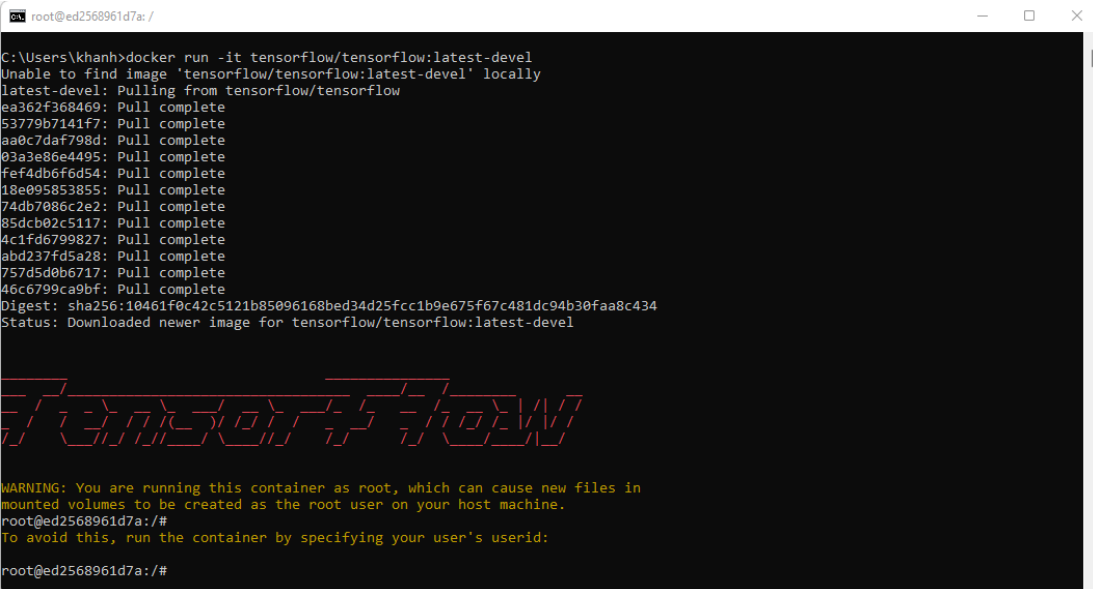
1.1 Cài đặt Docker

- Cài đặt Docker : <https://hub.docker.com/editions/community/docker-ce-desktop-windows>

1.2 Cài đặt Tensorflow trên Docker và training mô hình

- **Bước 1:** Đăng nhập vào docker bằng lệnh Docker login
- **Bước 2:** Download Tensorflow container

Docker run -it tensorflow/tensorflow:latest-devel



```
root@ed2568961d7a: /
C:\Users\khanh>docker run -it tensorflow/tensorflow:latest-devel
Unable to find image 'tensorflow/tensorflow:latest-devel' locally
latest-devel: Pulling from tensorflow/tensorflow
ea362f368469: Pull complete
53779b7141f7: Pull complete
aa0c7daf798d: Pull complete
03a3e86e4495: Pull complete
fef4db6f6d54: Pull complete
18e095853855: Pull complete
74db7086c2e2: Pull complete
85dcb02c5117: Pull complete
4c1fd6799827: Pull complete
abd237fd5a28: Pull complete
757d5d0b6717: Pull complete
46c6799ca9bf: Pull complete
Digest: sha256:10461f0c42c5121b85096168bed34d25fcc1b9e675f67c481dc94b30faa8c434
Status: Downloaded newer image for tensorflow/tensorflow:latest-devel

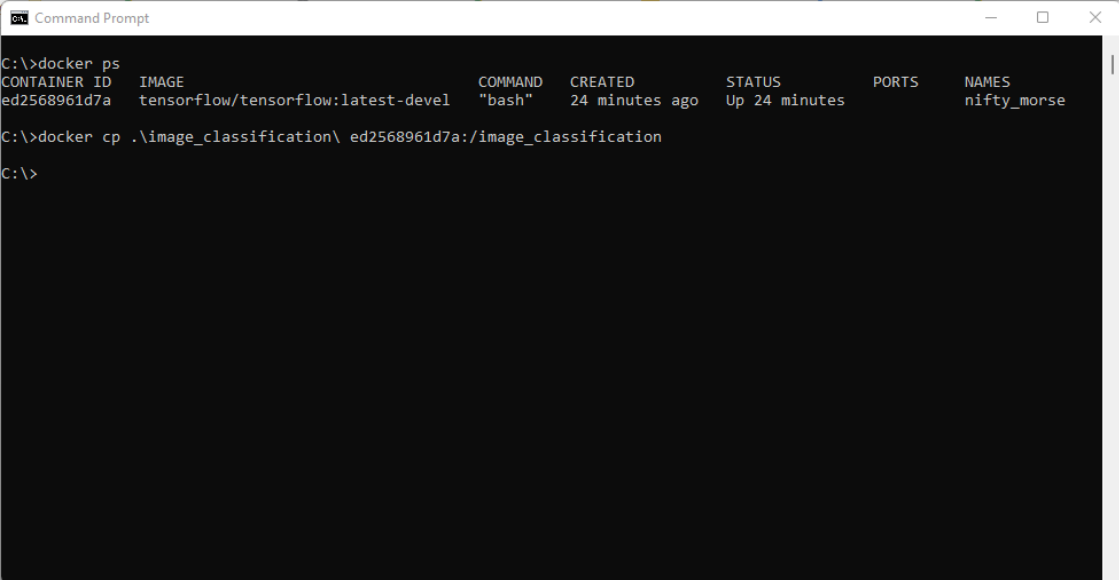
TensorFlow

WARNING: You are running this container as root, which can cause new files in
mounted volumes to be created as the root user on your host machine.
root@ed2568961d7a:/#
To avoid this, run the container by specifying your user's userid:
root@ed2568961d7a:/#
```

Hình 10: Kết quả sau khi pull image và chạy thành công container Tensorflow

- **Bước 3:** Chuẩn bị dữ liệu
 - Dữ liệu là bộ tf_flower được xây dựng bởi Tensorflow team
 - Dữ liệu gồm 5 loại hoa khác nhau:
 - Daisy: 633 ảnh
 - Dandelion: 898 ảnh
 - Roses: 641 ảnh
 - Sunflower: 699 ảnh
 - Tulips: 799 ảnh

- Chuẩn bị 2 file code python gồm :
 - Retrain.py: để train model
 - Label_image.py: để gán nhãn cho ảnh
- **Bước 4:** Copy thư mục dữ liệu đã chuẩn bị ở trên từ Windows sang container Tensorflow đã cài đặt
 - Sử dụng lệnh: `docker cp .\image_classification container_ID:/image_classification`

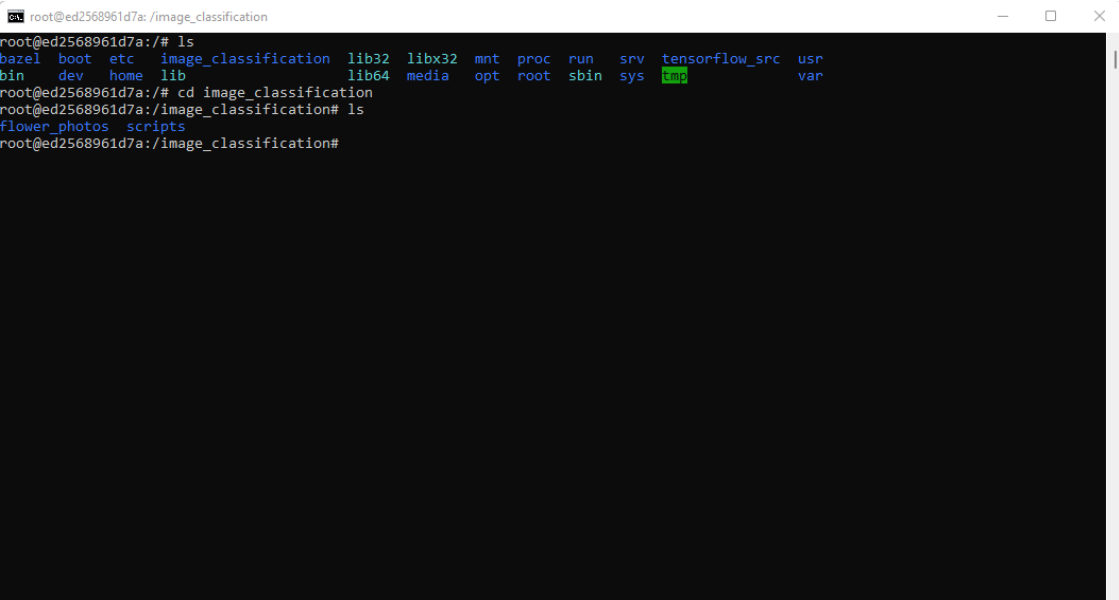


```
C:\>docker ps
CONTAINER ID   IMAGE                                COMMAND                  CREATED        STATUS        PORTS   NAMES
ed2568961d7a   tensorflow/tensorflow:latest-devel   "bash"                  24 minutes ago Up 24 minutes           nifty_morse

C:\>docker cp .\image_classification\ ed2568961d7a:/image_classification

C:\>
```

Hình 11: Copy data từ Windows sang docker container



```
root@ed2568961d7a: /image_classification

root@ed2568961d7a: /# ls
bazel  boot  etc  image_classification  lib32  libx32  mnt  proc  run  srv  tensorflow_src  usr
bin    dev   home  lib                    lib64  media   opt  root  sbin  sys  tmp         var

root@ed2568961d7a: /# cd image_classification
root@ed2568961d7a: /image_classification# ls
flower_photos  scripts

root@ed2568961d7a: /image_classification#
```

- **Bước 5:** Train lại model với hoa hồng(roses) và hoa hướng dương(sunflower)

```

root@2feb000304e9:/# cd image_classification/
root@2feb000304e9:/image_classification# ls
flowerpics  scripts
root@2feb000304e9:/image_classification# python scripts/retrain.py --bottleneck_dir=/bottleneck/ --model_dir=/inception --output_labels=/retrained_labels.txt --output_graph=/retrained_graph.pb --image_dir=/flowerpics/
INFO:tensorflow:Looking for images in 'roses'
INFO:tensorflow:Looking for images in 'sunflowers'
INFO:tensorflow:Using /tmp/tfhub_modules to cache modules.
INFO:tensorflow:Downloading TF-Hub Module 'https://tfhub.dev/google/imagenet/inception_v3/feature_vector/1'.

```

Hình 12: Train mô hình với hai tập dữ liệu hoa

- Tập lệnh sẽ bắt đầu train trên hai thư mục hình ảnh. Quá trình này mất khoảng 15 phút trên máy tính của em

```

INFO:tensorflow:Creating bottleneck at /bottleneck/roses/0060338380_eb6c806624_n.jpg_https-tfhub.dev-google-imagenet-inception_v3-feature_vector-1.txt
INFO:tensorflow:Creating bottleneck at /bottleneck/roses/12395698413_c080278f77.jpg_https-tfhub.dev-google-imagenet-inception_v3-feature_vector-1.txt
INFO:tensorflow:Creating bottleneck at /bottleneck/roses/6158504080_b844a9ae05.jpg_https-tfhub.dev-google-imagenet-inception_v3-feature_vector-1.txt
INFO:tensorflow:Creating bottleneck at /bottleneck/roses/6363951285_a802238d4e.jpg_https-tfhub.dev-google-imagenet-inception_v3-feature_vector-1.txt
INFO:tensorflow:Creating bottleneck at /bottleneck/roses/2892056920_918c52889b_m.jpg_https-tfhub.dev-google-imagenet-inception_v3-feature_vector-1.txt
INFO:tensorflow:Creating bottleneck at /bottleneck/roses/5060519573_c628547e20_n.jpg_https-tfhub.dev-google-imagenet-inception_v3-feature_vector-1.txt
INFO:tensorflow:Creating bottleneck at /bottleneck/roses/5529341024_0c35f2657d.jpg_https-tfhub.dev-google-imagenet-inception_v3-feature_vector-1.txt
INFO:tensorflow:Creating bottleneck at /bottleneck/roses/2501207526_cbd66a3f7e_m.jpg_https-tfhub.dev-google-imagenet-inception_v3-feature_vector-1.txt
INFO:tensorflow:Creating bottleneck at /bottleneck/roses/2535405431_e6f950443.jpg_https-tfhub.dev-google-imagenet-inception_v3-feature_vector-1.txt
INFO:tensorflow:Creating bottleneck at /bottleneck/roses/1838936880_01c24a2087_2.jpg_https-tfhub.dev-google-imagenet-inception_v3-feature_vector-1.txt
INFO:tensorflow:Creating bottleneck at /bottleneck/roses/3921704817_276eb4386b.jpg_https-tfhub.dev-google-imagenet-inception_v3-feature_vector-1.txt
INFO:tensorflow:Creating bottleneck at /bottleneck/roses/2408236801_f43c6bcff2.jpg_https-tfhub.dev-google-imagenet-inception_v3-feature_vector-1.txt
INFO:tensorflow:Creating bottleneck at /bottleneck/roses/5410629292_2f06e4b295.jpg_https-tfhub.dev-google-imagenet-inception_v3-feature_vector-1.txt
INFO:tensorflow:Creating bottleneck at /bottleneck/roses/17953368844_b3d18cf30_m.jpg_https-tfhub.dev-google-imagenet-inception_v3-feature_vector-1.txt
INFO:tensorflow:Creating bottleneck at /bottleneck/roses/8524505546_b24b2d4928_n.jpg_https-tfhub.dev-google-imagenet-inception_v3-feature_vector-1.txt
INFO:tensorflow:Creating bottleneck at /bottleneck/roses/12323085443_8ac0c0b713_n.jpg_https-tfhub.dev-google-imagenet-inception_v3-feature_vector-1.txt
INFO:tensorflow:Creating bottleneck at /bottleneck/roses/15537825851_a80b6321d7_n.jpg_https-tfhub.dev-google-imagenet-inception_v3-feature_vector-1.txt
INFO:tensorflow:Creating bottleneck at /bottleneck/roses/11757822526_fe30b93ca_m.jpg_https-tfhub.dev-google-imagenet-inception_v3-feature_vector-1.txt
INFO:tensorflow:Creating bottleneck at /bottleneck/roses/118974357_0faa23c0e9_n.jpg_https-tfhub.dev-google-imagenet-inception_v3-feature_vector-1.txt
INFO:tensorflow:Creating bottleneck at /bottleneck/roses/15509799653_0562d4a4fa.jpg_https-tfhub.dev-google-imagenet-inception_v3-feature_vector-1.txt
INFO:tensorflow:Creating bottleneck at /bottleneck/roses/16484108863_979beac08.jpg_https-tfhub.dev-google-imagenet-inception_v3-feature_vector-1.txt
INFO:tensorflow:Creating bottleneck at /bottleneck/roses/6864417932_36fa4ceecf_n.jpg_https-tfhub.dev-google-imagenet-inception_v3-feature_vector-1.txt
INFO:tensorflow:Creating bottleneck at /bottleneck/roses/3265902338_d8b1e4d545.jpg_https-tfhub.dev-google-imagenet-inception_v3-feature_vector-1.txt
INFO:tensorflow:Creating bottleneck at /bottleneck/roses/323872063_7264e7e018_m.jpg_https-tfhub.dev-google-imagenet-inception_v3-feature_vector-1.txt
INFO:tensorflow:Creating bottleneck at /bottleneck/roses/16334786972_1b9e71cab0_m.jpg_https-tfhub.dev-google-imagenet-inception_v3-feature_vector-1.txt

```

Hình 13: Quá trình training

- Sau khi train thành công

```

INFO:tensorflow:2019-01-17 20:19:40.011619: Step 3980: Train accuracy = 100.0%
INFO:tensorflow:2019-01-17 20:19:40.011761: Step 3980: Cross entropy = 0.018658
INFO:tensorflow:2019-01-17 20:19:40.054510: Step 3980: Validation accuracy = 96.0% (N=100)
INFO:tensorflow:2019-01-17 20:19:40.488059: Step 3990: Train accuracy = 100.0%
INFO:tensorflow:2019-01-17 20:19:40.488238: Step 3990: Cross entropy = 0.022904
INFO:tensorflow:2019-01-17 20:19:40.531031: Step 3990: Validation accuracy = 98.0% (N=100)
INFO:tensorflow:2019-01-17 20:19:40.922979: Step 3999: Train accuracy = 100.0%
INFO:tensorflow:2019-01-17 20:19:40.923208: Step 3999: Cross entropy = 0.011318
INFO:tensorflow:2019-01-17 20:19:40.982767: Step 3999: Validation accuracy = 94.0% (N=100)
2019-01-17 20:19:42.206509: W tensorflow/core/graph/graph_constructor.cc:1265] Importing a graph with a lower producer version 26 into an existing graph with producer version 27. Shape inference will have run different parts of the graph with different producer versions.
INFO:tensorflow:Saver not created because there are no variables in the graph to restore
INFO:tensorflow:Restoring parameters from /tmp/_retrain_checkpoint
INFO:tensorflow:Final test accuracy = 97.7% (N=131)
INFO:tensorflow:Save final result to : /retrained_graph.pb
2019-01-17 20:19:47.277709: W tensorflow/core/graph/graph_constructor.cc:1265] Importing a graph with a lower producer version 26 into an existing graph with producer version 27. Shape inference will have run different parts of the graph with different producer versions.
INFO:tensorflow:Saver not created because there are no variables in the graph to restore
INFO:tensorflow:Restoring parameters from /tmp/_retrain_checkpoint
INFO:tensorflow:Froze 378 variables.
INFO:tensorflow:Converted 378 variables to const ops.
WARNING:tensorflow:From scripts/retrain.py:909: __init__ (from tensorflow.python.platform.gfile) is deprecated and will be removed in a future version.
Instructions for updating:
Use tf.gfile.GFile.
root@2feb000304e9:/image_classification#

```

Hình 14: Sau khi train thành công

- Bước 6: Test model

- Sử dụng ảnh hoa hồng:



Hình 15: Ảnh hoa hồng được sử dụng

```
root@2feb000304e9: /image_classification (Admin)
<1> root@2feb000304e9: <2> Windows PowerShell
INFO:tensorflow:Saver not created because there are no variables in the graph to restore
INFO:tensorflow:Restoring parameters from /tmp/_retrain_checkpoint
INFO:tensorflow:Froze 378 variables.
INFO:tensorflow:Converted 378 variables to const ops.
WARNING:tensorflow:From scripts/retrain.py:989: __init__ (from tensorflow.python.platform.gfile) is deprecated and will be removed in a future version.
Instructions for updating:
Use tf.gfile.GFile.
root@2feb000304e9: /image_classification# ls -al
total 24
drwxr-xr-x 6 root root 4096 Jan 17 20:15 .
drwxr-xr-x 1 root root 4096 Jan 17 20:13 ..
drwxr-xr-x 2 root root 4096 Jan 17 20:15 bottleneck
drwxr-xr-x 4 root root 4096 Jan 17 19:30 flowerpics
drwxr-xr-x 2 root root 4096 Jan 17 20:15 inception
drwxr-xr-x 2 root root 4096 Jan 17 19:40 scripts
root@2feb000304e9: /image_classification# ls -al
total 284
drwxr-xr-x 6 root root 4096 Jan 17 20:28 .
drwxr-xr-x 1 root root 4096 Jan 17 20:13 ..
drwxr-xr-x 2 root root 4096 Jan 17 20:15 bottleneck
-rwxr-xr-x 1 root root 263347 Jan 17 20:28 dublin-rose.jpg
drwxr-xr-x 4 root root 4096 Jan 17 19:30 flowerpics
drwxr-xr-x 2 root root 4096 Jan 17 20:15 inception
drwxr-xr-x 2 root root 4096 Jan 17 19:40 scripts
root@2feb000304e9: /image_classification#
```

Hình 16: Sử dụng ảnh để test

- Tiếp theo sử dụng đoạn mã để test:

```
python scripts/label_image.py --graph=/retrained_graph.pb --
labels=/retrained_labels.txt --input_layer=Placeholder --
output_layer=final_result --image=dublin-rose.jpg
```

- Kết quả cuối cùng:

```
root@2feb000304e9: /image_classification# ls -a
. .. bottleneck dublin-rose.jpg flowerpics inception scripts
root@2feb000304e9: /image_classification#
root@2feb000304e9: /image_classification#
root@2feb000304e9: /image_classification# python scripts/label_image.py --graph=/retrained_graph.pb --labels=/retrained_labels.txt --input_layer=Placeholder --output_layer=final_result --image=dublin-rose.jpg
2019-01-17 20:33:14.687246: I tensorflow/core/platform/cpu_feature_guard.cc:141] Your CPU supports instructions that this TensorFlow binary was not compiled to use: AVX2 FMA
roses 0.9997321
sunflowers 0.00026787148
root@2feb000304e9: /image_classification#
```

Hình 17: Kết quả sau khi test

CHƯƠNG 4. SỬ DỤNG TENSORFLOW XÂY DỰNG MÔ HÌNH HỌC MÁY TRÊN GOOGLE COLAB

Ngày nay, với các bài toán học máy được xây dựng với TensorFlow, người ta sử dụng Keras là một API cấp cao của TensorFlow. Keras giúp người dùng xây dựng mô hình học máy một cách đơn giản, dễ sử dụng, dễ mở rộng.

Nhóm chúng em sử dụng GPU của Google Colab để xây dựng và huấn luyện một mô hình học máy trên Cloud.

4.1. Xây dựng bài toán

- Đề tài của bài toán là “Nhận diện cảm xúc gương mặt – Facial Expression”.
- Đầu vào: Gương mặt một người bất kỳ
- Đầu ra: Dự đoán về cảm xúc của gương mặt

4.2. Chuẩn bị dữ liệu

- Dữ liệu được sử dụng là bộ FER2013.csv, được thu thập từ Kaggle.
- Đây là bộ ảnh gồm 35.887 bức ảnh đen trắng, tỉ lệ 48 x 48
- Dữ liệu bao gồm 7 loại cảm xúc khác nhau, bao gồm:
 - Túc giận: 4953 ảnh
 - Ghê tởm: 547 ảnh
 - Sợ hãi: 5121 ảnh
 - Vui vẻ: 8989 ảnh
 - Buồn: 6077 ảnh
 - Ngạc nhiên: 4002 ảnh
 - Bình thường: 6198 ảnh

4.3. Mã nguồn của chương trình

- Kết nối Google Colab với Google Drive
- Import các thư viện cần thiết
- Kiểm tra phiên bản của TensorFlow, phiên bản TensorFlow đang được sử dụng là 2.7.0

```
[ ] from google.colab import drive
drive.mount('/content/gdrive')
```

Mounted at /content/gdrive

```
import os
import numpy as np
import pandas as pd
import tensorflow as tf
from matplotlib import pyplot

from sklearn.model_selection import train_test_split
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from keras.utils import np_utils
```

```
[ ] tf.__version__
```

'2.7.0'

- Tiến hành Load Dataset từ Kaggle vào Google Colab

▼ Load Dataset from Kaggle

```
!pip install -q kaggle
```

```
from google.colab import files
```

```
files.upload()
```

Chọn tệp. Không có tệp nào được chọn. Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.
Saving kaggle.json to kaggle.json
{'kaggle.json': b'{"username": "hoanghaimy", "key": "428e41cea6dd3c88e59c119ea5664930"}'}

```
[ ] !mkdir ~/.kaggle
```

```
!cp kaggle.json ~/.kaggle/
```

```
!chmod 600 ~/.kaggle/kaggle.json
```

```
[ ] !kaggle datasets download -d deaddskull17/fer2013
```

```
[ ] Downloading fer2013.zip to /content
83% 80.0M/96.6M [00:02<00:00, 29.8MB/s]
100% 96.6M/96.6M [00:02<00:00, 47.7MB/s]
```

```
!unzip fer2013.zip
```

Archive: fer2013.zip
inflating: fer2013.csv

```
[ ] df = pd.read_csv('fer2013.csv')
df.head()
```

	emotion	pixels	Usage
0	0	70 80 82 72 58 58 60 63 54 58 60 48 89 115 121...	Training
1	0	151 150 147 155 148 133 111 140 170 174 182 15...	Training
2	2	231 212 156 164 174 138 161 173 182 200 106 38...	Training
3	4	24 32 36 30 32 23 19 20 30 41 21 22 32 34 21 1...	Training
4	6	4 0 0 0 0 0 0 0 0 0 0 3 15 23 28 48 50 58 84...	Training

- Kiểm tra số lượng ảnh của tập Training, Private Test và Public Test

- Kiểm tra số lượng ảnh của từng class

```
df.Usage.value_counts()

Training      28709
PrivateTest   3589
PublicTest    3589
Name: Usage, dtype: int64

[ ] df.emotion.value_counts()

3      8989
6      6198
4      6077
2      5121
0      4953
5      4002
1       547
Name: emotion, dtype: int64

[ ] df.emotion.unique()

array([0, 2, 4, 6, 3, 5, 1])
```

- Xử lý dữ liệu:

- + Phân chia và đặt tên các class cảm xúc
- + Load data vào tập train, validation và tập test
- + Tăng cường dữ liệu

Preprocessing Dataset

```
[ ] num_classes = 7
    label_to_text = {0: 'Angry', 1: 'Disgust', 2: 'Fear', 3: 'Happy', 4: 'Sad', 5: 'Surprise', 6: 'Neutral'}

[ ] train_set = df[(df.Usage == 'Training')]
    val_set = df[(df.Usage == 'PublicTest')]
    test_set = df[(df.Usage == 'PrivateTest')]

    X_train = np.array(list(map(str.split, train_set.pixels)), np.float32)
    X_val = np.array(list(map(str.split, val_set.pixels)), np.float32)
    X_test = np.array(list(map(str.split, test_set.pixels)), np.float32)

    X_train = X_train.reshape(X_train.shape[0], 48, 48, 1)
    X_val = X_val.reshape(X_val.shape[0], 48, 48, 1)
    X_test = X_test.reshape(X_test.shape[0], 48, 48, 1)

[ ] num_train = X_train.shape[0]
    num_val = X_val.shape[0]

num_train
#num_val
#num_test

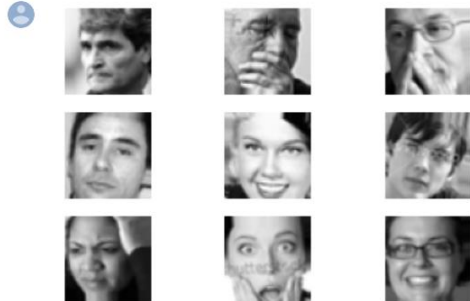
28709

[ ] y_train = train_set.emotion
    y_train = np_utils.to_categorical(y_train, num_classes) # convert to one-hot vector
    y_val = val_set.emotion
    y_val = np_utils.to_categorical(y_val, num_classes)
    y_test = test_set.emotion
    y_test = np_utils.to_categorical(y_test, num_classes)

[ ] datagen = ImageDataGenerator(
    rescale=1./255,
    rotation_range = 10,
    horizontal_flip = True,
    width_shift_range=0.1,
    height_shift_range=0.1,
    fill_mode = 'nearest')

testgen = ImageDataGenerator(
```

```
for X_batch, y_batch in datagen.flow(X_train, y_train, batch_size=9):  
    for i in range(0, 9):  
        pyplot.axis('off')  
        pyplot.subplot(330 + 1 + i)  
        pyplot.imshow(X_batch[i].reshape(48, 48), cmap=pyplot.get_cmap('gray'))  
    pyplot.axis('off')  
    pyplot.show()  
    break
```



- Xây dựng model với Tensorflow và Keras

▼ Buiding Model

```
basemodel = tf.keras.models.Sequential([tf.keras.layers.Conv2D(32,(3,3),activation='relu',input_shape = (48,48,1)),  
    tf.keras.layers.MaxPool2D(2,2),  
    tf.keras.layers.BatchNormalization(),  
    #  
    tf.keras.layers.Conv2D(64,(3,3),activation='relu',input_shape = (48,48,1)),  
    tf.keras.layers.MaxPool2D(2,2),  
    tf.keras.layers.BatchNormalization(),  
    #  
    tf.keras.layers.Conv2D(128,(3,3),activation='relu',input_shape = (48,48,1)),  
    tf.keras.layers.MaxPool2D(2,2),  
    tf.keras.layers.BatchNormalization(),  
    #  
    tf.keras.layers.Conv2D(256,(3,3),activation='relu',input_shape = (48,48,1)),  
    tf.keras.layers.MaxPool2D(2,2),  
    tf.keras.layers.Flatten(),  
    tf.keras.layers.Dense(1000,activation = 'relu'),  
    tf.keras.layers.Dense(7,activation = 'softmax')  
])
```

- Định nghĩa optimizer được sử dụng, learning rate, hàm loss và metrics
- Thiết lập Checkpoint để lưu lại weight tốt nhất


```
[ ] basemodel.compile(optimizer = tf.keras.optimizers.RMSprop(learning_rate=0.001),
                      loss = 'categorical_crossentropy',
                      metrics = ['accuracy'])

file_name = 'best_model.h5'
checkpoint_path = os.path.join('/content/gdrive/MyDrive/Facial_Expression_TF/checkpoint/', file_name)

call_back = tf.keras.callbacks.ModelCheckpoint(filepath = checkpoint_path,
                                              monitor = 'val_accuracy',
                                              verbose=1,
                                              save_freq = 'epoch',
                                              save_best_only = True,
                                              save_weights_only = False,
                                              mode = 'max'
                                              )

[ ] checkpoint_path

'/content/gdrive/MyDrive/Facial_Expression_TF/checkpoint/best_model.h5'
```

- Tiến hành training model với 50 epochs, batch size là 64

▼ Training Model

```
[ ] history = basemodel.fit(train_flow, epochs = 50, validation_data=val_flow, callbacks = call_back)

Epoch 1/50
449/449 [=====] - ETA: 0s - loss: 1.6425 - accuracy: 0.3502
Epoch 0001: val_accuracy improved from -inf to 0.28643, saving model to /content/gdrive/MyDrive/Facial_Expression_TF/checkpoint/best_model.h5
449/449 [=====] - 36s 51ms/step - loss: 1.6425 - accuracy: 0.3502 - val_loss: 1.7411 - val_accuracy: 0.2864
Epoch 2/50
448/449 [=====>.] - ETA: 0s - loss: 1.4785 - accuracy: 0.4326
Epoch 0002: val_accuracy improved from 0.28643 to 0.45890, saving model to /content/gdrive/MyDrive/Facial_Expression_TF/checkpoint/best_model.h5
449/449 [=====] - 21s 48ms/step - loss: 1.4786 - accuracy: 0.4324 - val_loss: 1.3985 - val_accuracy: 0.4589
Epoch 3/50
448/449 [=====>.] - ETA: 0s - loss: 1.3937 - accuracy: 0.4696
Epoch 0003: val_accuracy improved from 0.45890 to 0.48119, saving model to /content/gdrive/MyDrive/Facial_Expression_TF/checkpoint/best_model.h5
449/449 [=====] - 22s 50ms/step - loss: 1.3930 - accuracy: 0.4698 - val_loss: 1.3848 - val_accuracy: 0.4812
Epoch 4/50
448/449 [=====>.] - ETA: 0s - loss: 1.3439 - accuracy: 0.4898
Epoch 0004: val_accuracy improved from 0.48119 to 0.49568, saving model to /content/gdrive/MyDrive/Facial_Expression_TF/checkpoint/best_model.h5
449/449 [=====] - 22s 49ms/step - loss: 1.3440 - accuracy: 0.4898 - val_loss: 1.3326 - val_accuracy: 0.4957
Epoch 5/50
449/449 [=====] - 1. ETA: 0s - loss: 1.3003 - accuracy: 0.5040
```

- Kết quả thu thập được là độ chính xác 62.6% trên tập test

```
449/449 [=====] - ETA: 0s - loss: 0.8967 - accuracy: 0.6670
Epoch 0040: val_accuracy did not improve from 0.61410
449/449 [=====] - 22s 50ms/step - loss: 0.8967 - accuracy: 0.6670 - val_loss: 1.1259 - val_accuracy: 0.5893
Epoch 41/50
448/449 [=====>.] - ETA: 0s - loss: 0.8943 - accuracy: 0.6652
Epoch 0041: val_accuracy did not improve from 0.61410
449/449 [=====] - 22s 50ms/step - loss: 0.8944 - accuracy: 0.6651 - val_loss: 1.1308 - val_accuracy: 0.5874
Epoch 42/50
449/449 [=====] - ETA: 0s - loss: 0.8918 - accuracy: 0.6664
Epoch 0042: val_accuracy did not improve from 0.61410
449/449 [=====] - 23s 50ms/step - loss: 0.8918 - accuracy: 0.6664 - val_loss: 1.0914 - val_accuracy: 0.6080
Epoch 43/50
449/449 [=====] - ETA: 0s - loss: 0.8813 - accuracy: 0.6706
Epoch 0043: val_accuracy did not improve from 0.61410
449/449 [=====] - 24s 53ms/step - loss: 0.8813 - accuracy: 0.6706 - val_loss: 1.0891 - val_accuracy: 0.6060

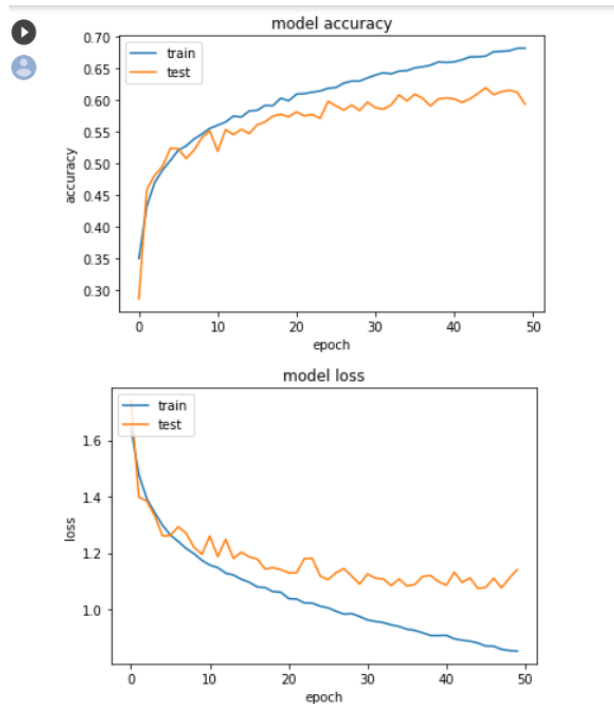
test_loss, test_acc = basemodel.evaluate( test_flow, verbose=2)

print('\nTest accuracy:', test_acc)

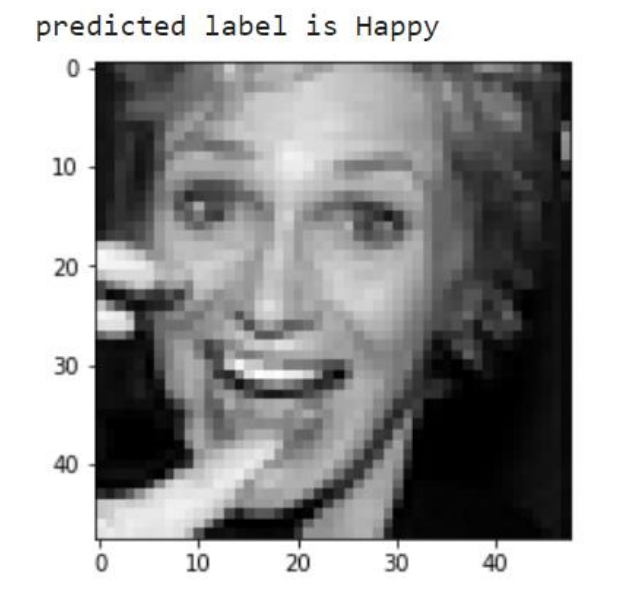
57/57 - 1s - loss: 1.0184 - accuracy: 0.6261 - 540ms/epoch - 9ms/step

Test accuracy: 0.6260796785354614
```

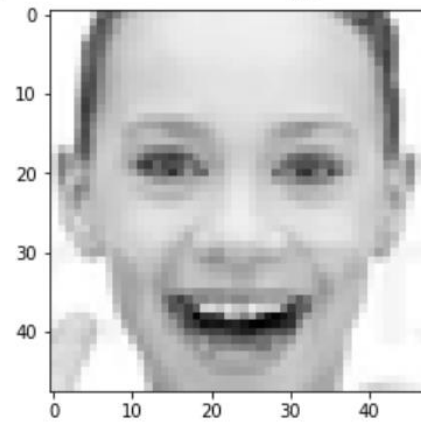
- Biểu đồ Accuracy và Loss



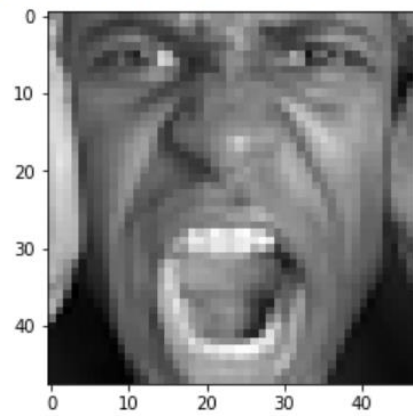
- Một số kết quả dự đoán của mô hình



```
... predicted label is Happy
```



```
... predicted label is Angry
```



KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

Thông qua bài tập lớn với đề tài “**Tìm hiểu thư viện mã nguồn mở TensorFlow**”, nhóm chúng em đã tìm hiểu về bộ thư viện mã nguồn mở TensorFlow, nắm được lịch sử ra đời, kiến trúc, các thức hoạt động và các ưu điểm mà bộ thư viện mã nguồn mở TensorFlow mang lại. Đồng thời, nhóm chúng em đã cài đặt là làm quen với bộ thư viện, sử dụng TensorFlow để training hai mô hình học máy. Mô hình đầu tiên là mô hình phân loại hoa, được huấn luyện trên Docker, mô hình thứ hai là mô hình nhận diện cảm xúc gương mặt, được huấn luyện trên Google Colab.

Từ đó, nhóm chúng em đã nắm được kiến thức và cách sử dụng thư viện TensorFlow, nắm được các xây dựng mô hình học máy với bộ thư viện mã nguồn mở này. Đối với từng thành viên trong nhóm, với sự phân chia công việc hợp lý cũng như sự nhiệt tình của các thành viên, chúng em đã giúp đỡ nhau, chia sẻ kiến thức, qua đó mỗi thành viên trong nhóm đều đã nắm được nội dung và thực hành tốt bài tập lớn.

Về hướng phát triển, sau khi hiểu rõ được cách thức hoạt động, cách sử dụng bộ thư viện TensorFlow, mỗi thành viên trong nhóm chúng em sẽ sử dụng bộ thư viện để xây dựng các mô hình học máy, học sâu, phục vụ cho việc nghiên cứu và học tập của mỗi người.

Tuy nhiên, bài tập lớn của chúng em không thể tránh khỏi những thiếu sót, chúng em mong muốn được nhận những lời góp ý và nhận xét từ thầy và cả lớp để chúng em có thể rút kinh nghiệm và hoàn thiện bài tập lớn được tốt hơn. Nhóm chúng em xin gửi lời cảm ơn chân thành tới thầy Trần Hải Anh vì những kiến thức thầy giảng dạy, cũng như những lời đánh giá, góp ý, chia sẻ tận tình của thầy đã giúp chúng em thực hiện tốt đề tài bài tập lớn.

TÀI LIỆU THAM KHẢO

- [1] Tensorflow là gì? 10 tài liệu học tensorflow đầy đủ nhất | TopDev
- [2] Tensorflow cho người mới bắt đầu (viblo.asia)
- [3] Bắt đầu với Machine Learning thông qua Tensorflow (Phần I.2)
(kipalog.com)
- [4] Tensorflow là gì? Lợi ích của nó tuyệt vời ra sao? (timviec365.com)
- [5] Nguồn download dataset: <https://www.kaggle.com/deadskull7/fer2013>