TensorFlow

하 석 재
sjha72@gmail.com

왜 텐서플로우인가?

- 작년(2016)의 가장 힛트 단어?
  - 알파고(AlphaGo) !!!
- 딥마인드(DeepMind)
  - Demis Hassabis(CEO)
  - 영국회사 2010년에 영국에서 설립
  - 구글에 2014에 인수
  - **기계학습(machine learning)**과 **신경과학(neuroscience)**을 **기반**으로
  - 인간 지능을 분석, 구현
- DQN(Deep Q Network)
  - 딥마인드가 개발한 기계학습 알고리즘
  - Human-level control through Deep Reinforcement Learning(2015)

텐서플로우란?

- 구글이 만든 오픈소스 머신러닝/딥러닝 라이브러리
  - 하둡에코시스템의 머하웃(Mahout)에 대응
  - An open-source software library for Machine Intelligence
- 2015년 11월에 발표
- 현재 버전 1.4
- 내부적으로 구글 포토 외에 다양한 구글 서비스에 적용되어 있슴
- 소스는 공개되었으나 학습데이터는 공개하지 않음
- 파이썬, C++, 자바지원
- CPU/GPU기반(NVIDIA CUDA지원)
  - GPGPU(General Purpose Computing on GPU)
  - Distributed 버전 발표함
  - OpenCL은 1.2까지 지원
- 사실상 리눅스 환경을 요구함
  - 윈도우에서 가상환경을 통해 사용 가능

텐서플로우는

- General purpose Machine Learning Library
  - 다양한 용도에 사용 가능하지만 특정 기술(예:Neural Network)에 특화(최적화)되지 않음
  - 특화된 라이브러리에 비해 느림
  - 코딩이 상대적으로 복잡함
  - 느린 성능을 Scalability로 해결
  - 전용 CPU(TPU)를 설계

인공지능(AI:Artificial Intelligence)이란?

- 위키
  - 인공지능(**人工知能**, artificial intelligence, AI)은 기계로부터 만들어진 지능을 말한다. 컴퓨터 공학에서 이상적인 지능을 갖춘 존재, 혹은 시스템에 의해 만들어진 지능,즉 인공적인 지능을 뜻한다.
  - CAD(Computer Aided Design)
  - CAM(Computer Aided Manufacturing)
  - CASE(Computer Aided Software Engineering)
    - 인공지능 프로그래밍
      - 자동 프로그래밍(Automatic Programming)
      - cf. 자율주차, 자율주행
    - UML(Unified Modeling Language), …
    - 프레임워크, …
  - cf. 튜링테스트(Turing Test)
    - 기계가 인간과 얼마나 비슷하게 대화할 수 있는지를 기준으로 기계에 지능이 있는지를 판별하고자 하는 **테스트**로, 앨런 **튜링**이 1950년에 제안했다.

인공지능의 흐름

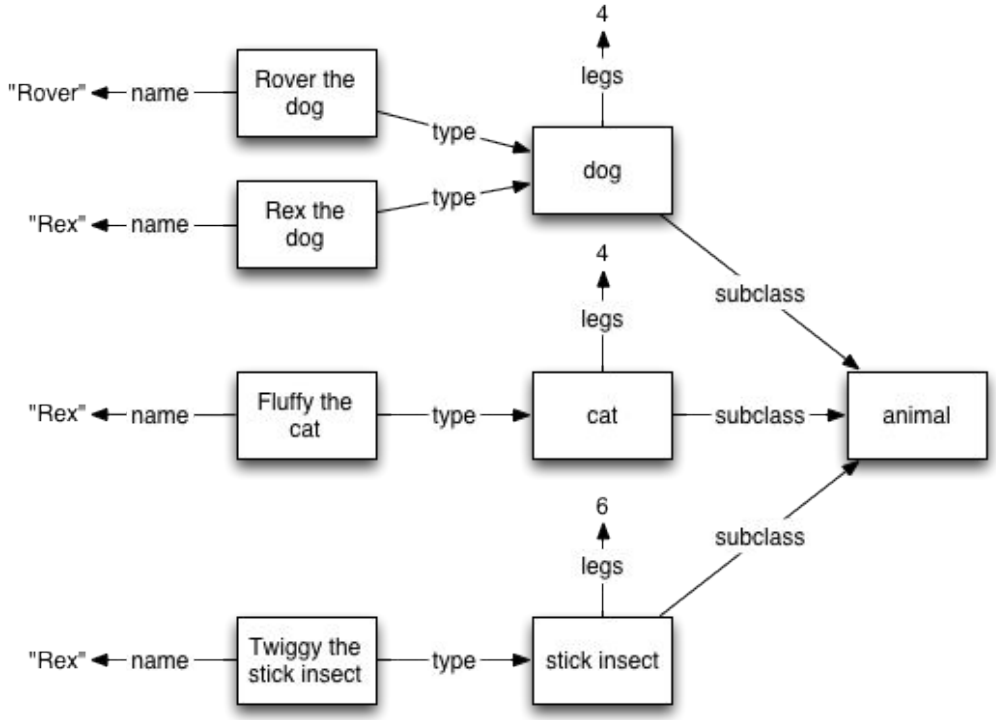
1. Language(80년대)

    - 논리(Logic) 모델링 언어
    - 술어논리(Predicate Logic)에 기반
        - cf. 삼단논법
            - 모든 사람은 죽는다
            - 철수는 사람이다
            - 즉, 철수는 죽는다
    - LISP(미국) / Prolog(일본,유럽)
        - Map/Reduce -> 빅데이터

인공지능의 흐름

2. Onthology(2000년대 초반)

- 시맨틱 웹을 구현할 수 있는 도구로서, 지식개념을 의미적으로 연결할수 있는 도구로서 RDF, OWL, SWRL 등의 언어를 이용해 표현한다.
- RDF(Resource Description Framework)
- cf.
  - 하늘에서 눈이 내린다.
  - 눈이 아프다.



https://www.ebi.ac.uk/rdf/sites/ebi.ac.uk.rdf/files/images/example3.png

인공지능의 흐름

3. 머신러닝/딥러닝(~ 현재)

- 신경망이론의 문제
  - 다른 기법으로 풀지 못하였던 복잡한 문제 해결가능하지만 계층을 깊게 하면 계산이 복잡하여 연산이 불가능
- ”A fast learning algorithm for deep belifef nets” , 2006
  - 캐나다의 CIFAR (Canadian Institute for Advanced Research) 연구소의 Hinton 교수의 논문
  - 뉴럴네트워크에 입력하는 초기값을 제대로 입력하면 여러 계층의 레이어에서도 연산이 가능하다는 것을 증명
- ”Greedy Layer-Wise training of deep network” , 2007
  - Yosua Bengio 의 논문에서 다계층의 신경망을 구축하면 복잡한 문제를 풀 수 있다는 것을 증명

인공지능의 흐름

- 딥러닝 = 머신러닝 +신경망이론
  - 리브랜딩(Re-branding)
- 현재는 머신러닝의 주류가 됨

머신러닝이란?

- 데이타를 기반으로 학습을 시켜서 예측하게 만드는 기법
- 통계학적으로는 추측 통계학(Inferential statistics)에 해당
- 지도학습(Supervised Learning)
  - 학습데이터 집합(Training Set), 원하는 결과 값(Desired Output)
    - 레이블된 데이터(Labeled Data)
  - 비용함수(Cost Function)
    - 표준편차 계산
    - 주로 **회귀(Regression)**
    - 학습에 사용하지 않는 검증 데이터 셋에 대해 오류 발생

머신러닝이란?

- 비지도학습(Unsupervised Learning)
  - 레이블 없는 데이터를 이용해 학습하는 방법
  - 데이터의 상호 유사성을 판단해 공통된 특징을 찾아내는 과정
  - 몇 개의 클래스로 분류될지, 클래스 안에는 몇 개의 데이터가 들어갈지 알 수 없슴
  - **군집화(Aggregation) / 분류(Classification)**

머신러닝의 예

- 이메일 스팸 필터링
- 편지봉투 우편번호 글자 인식
- 쇼핑몰이나 케이블 TV의 추천 시스템
  - 아마존/넷플릭스
- 자연어 인식
- 자동차 자율 주행

신경망(Neural Network)이란?

- 위키
  - 생물학의 신경망(동물의 중추신경계, 특히 뇌)에서 영감을 얻은 통계학적 학습 알고리즘
  - 인공신경망은 시냅스의 결합으로 네트워크를 형성한 인공 뉴런(노드)이 학습을 통해 시냅스의 결합 세기를 변화시켜, 문제 해결 능력을 가지는 모델 전반을 가리킴
  - 신경망은 일반적으로 규칙기반 프로그래밍으로 풀기 어려운 컴퓨터 비전 또는 음성 인식과 같은 다양한 범위의 문제를 푸는데 이용한다
- 정의
  - 통계학적 모델들의 집합이 다음과 같은 특징들을 가진다면 해당 집합을 신경(neural)이라고 부른다
  1. 조정이 가능한 가중치들의 집합 즉, 학습 알고리즘에 의해 조정이 가능한 숫자로 표현된 매개변수로 구성되어있다.
  2. 입력의 비선형 함수를 유추할 수 있다.

딥러닝(Deep Learning)이란?

- 머신러닝(ML)의 한 분야
- 위키피디아
  - 여러 비선형 변환기법의 조합을 통해 높은 수준의 추상화(abstractions, 다량의 데이터나 복잡한 자료들 속에서 핵심적인 내용 또는 기능을 요약하는 작업)를 시도하는 기계학습(machine learning) 알고리즘의 집합
  - 큰 틀에서 사람의 사고방식을 컴퓨터에게 가르치는 기계학습의 한 분야
- **머신러닝과 신경망(Neural Network)이 결합**
- 세부적으로는
  - deep neural networks
  - convolutional deep neural networks
  - deep belief networks, DQN 등이 있슴

텐서플로우 관련 자료

- 텐서플로우 공식사이트
  - https://www.tensorflow.org/
- 텐서플로우 코리아
  - https://github.com/tensorflowkorea/tensorflow-kr
  - https://www.gitbook.com/book/tensorflowkorea/tensorflow-kr/details
  - 잔카를로 자코네(김창엽 번역), “텐서플로 입문:예제로 배우는 텐서플로”, 에이콘출판사
- 각종 블로그자료
  - http://bcho.tistory.com/category/빅데이터/머신러닝
  - http://yujuwon.tistory.com/entry/TENSOR-FLOW-MNIST-인식하기
  - ...

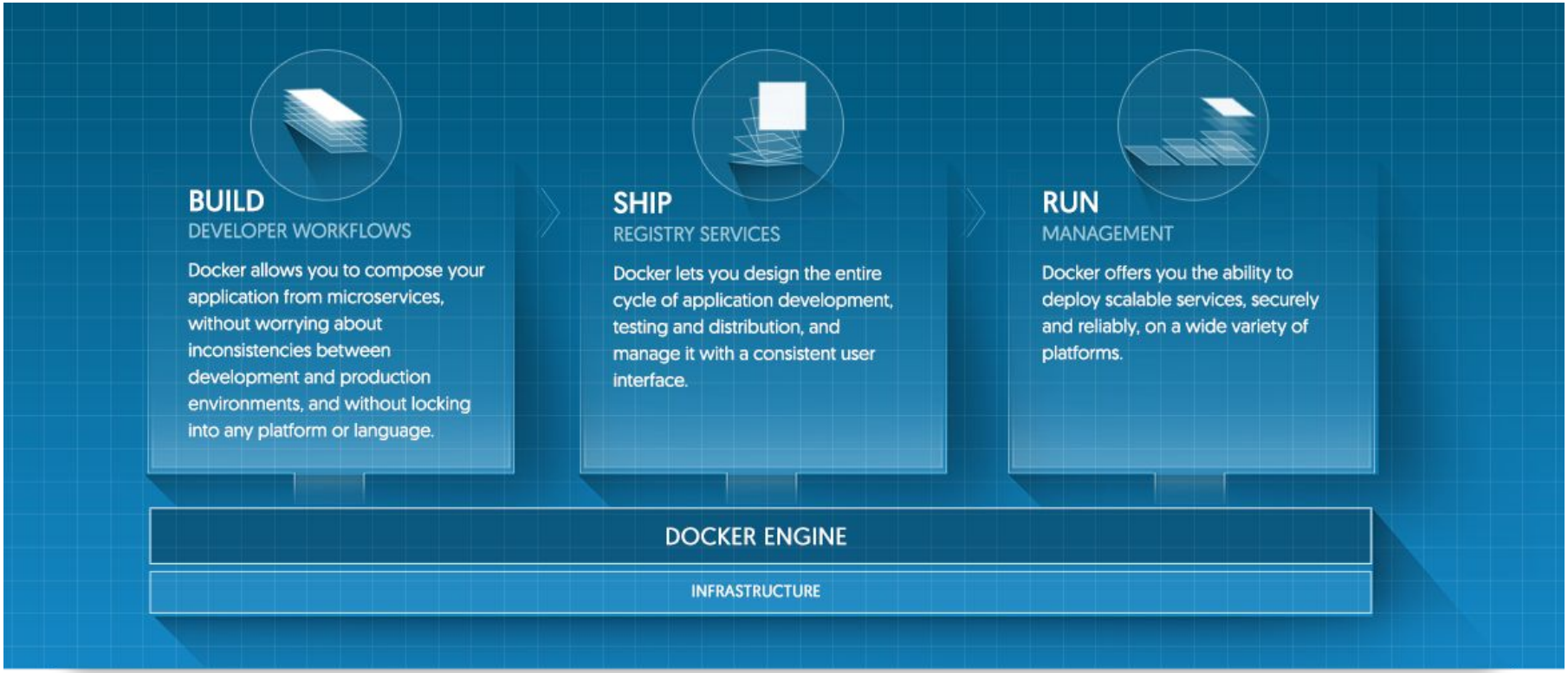
Tensorflow를 설치하기 위해서는

- 필수(Required)
  - Python 설치
  - Tensorflow 설치
- 선택(Optional)
  - IDE
  - IPython/Jupyter Notebook
- 가상화 툴VMWare/VirtualBox
  - 도커(Docker)
- 리눅스
  - 우분투(Ubuntu)

본 수업에서는 **도커(Docker) 사용**

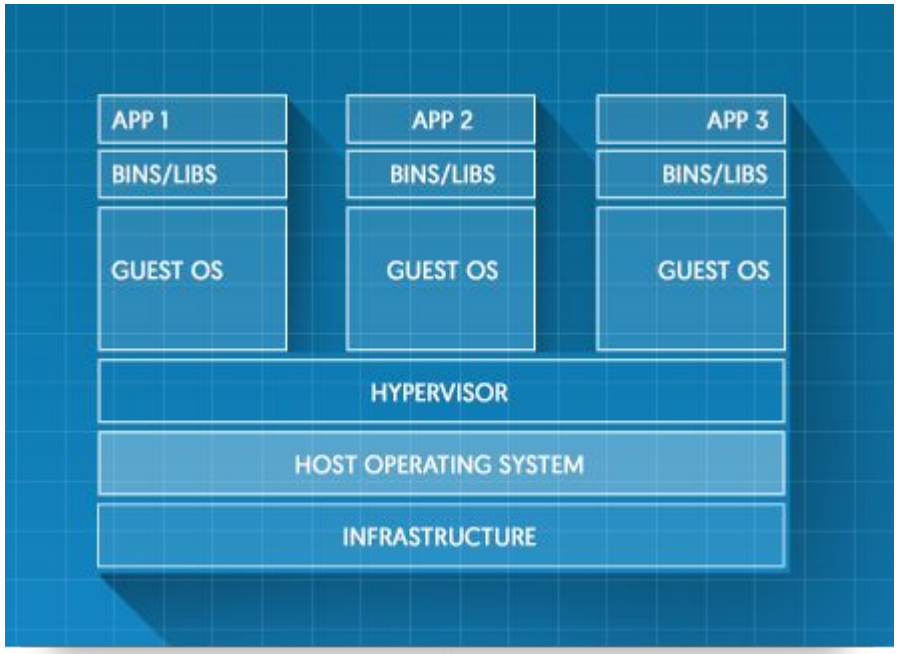
도커란

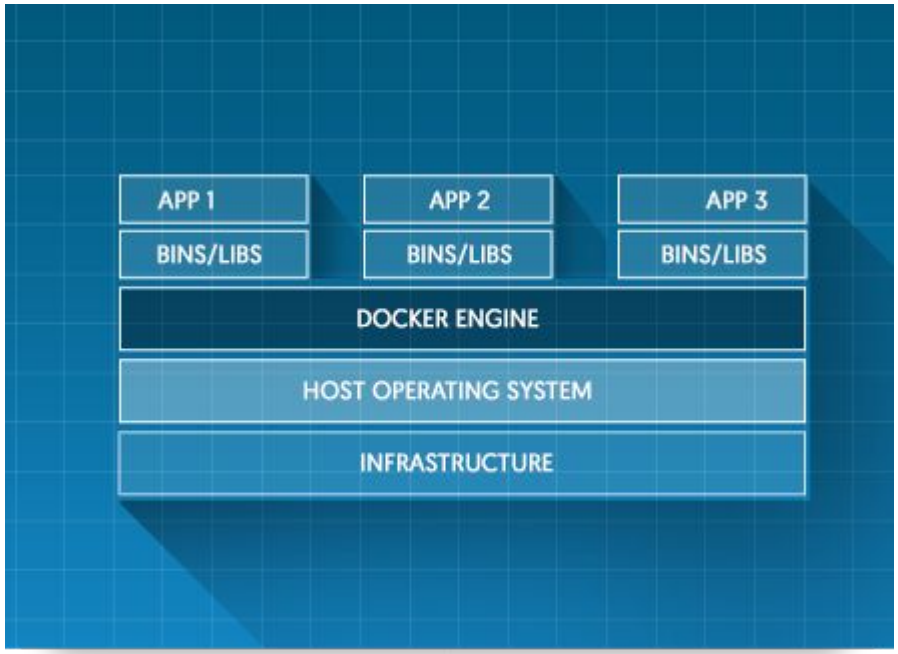
Build-Ship-Run



컨테이너 기반 가상화

- 기존의 가상화와 다른 개념
  - 하드웨어 가상화가 아닌 실행환경의 분리(isolation)
  - 각 컨테이너간 영향을 분리





하이퍼바이저 / 도커

도커의 성능

- 오버헤드가 5%이내

items	method	host	docker
CPU	sysbench	1	0.9931
memory	sysbench seq	1 (r)	0.9999
		1 (w)	0.9759
	sysbench rnd	1 (r)	1.0056
		1 (w)	0.9807
disk	dd	1	0.9716
network	iperf	1	0.7889

도커의 특징

- 모든 컨테이너들이 **동일 OS 커널 공유**
  - 독립적인 스케줄링이나
    CPU/메모리/디스크/네트워크를 가상화하지 않음
- 리눅스의 특수 기능(LXC)을 사용한 실행환경 격리를 응용
  - **리눅스에서만 사용가능**
    - 처음에는 우분투에서 현재 리눅스 배포판(fedora, RHEL, centos, …) 에서 사용가능
  - 다른 OS(윈도우/OSX)에서는
    - 일반 하이퍼바이저(경량)가 있어야 함
  - 현재는 LXC-> Libcontainer를 사용해 리눅스 의존도를 줄이려하고 있슴

도커 설치

- http://docker.com
- 요구사항
  - 윈도우 64비트 버전이상
  - 도커 툴박스(윈도우 8.1 이하)/도커 머신(윈도우 10이상)
  - Boot2Docker vs. Docker Machine
- 다운로드 & 설치

도커툴박스 vs. 도커머신

- Boot2docker(deprecated)
  - Tiny Core linux 기반의 경량 리눅스배포판 사용
  - 내부적으로 버추얼박스 지원
- Docker machine(new)
  - 가상호스트에 도커엔진 설치하는 툴
  - 버추얼박스, vmware 지원

도커기반 우분투 설치

- 도커 이미지 검색(기본이 최신버전)
  - **docker search ubuntu**
- 우분투 이미지 다운로드
  - **docker pull ubuntu**
- 이미지 리스트 출력
  - **docker images**
- 컨테이너 생성
  - **docker run --name=ubuntu ubuntu**
- 컨테이너 접속
  - docker attach ubuntu
  - **docker exec -it ubuntu bash**
- 컨테이너 탈출
  - **exit** or **컨트롤-P-Q**(컨테이너 정지하지 않고 나옴)

도커기본 명령어

- 도커 컨테이너 리스트
  - **docker ps -a**
- 도커 컨테이너 정지
  - **docker stop ubuntu**
- 도커 컨테이너 재시작
  - **docker restart ubuntu**
- 도커 컨테이너 삭제
  - **docker rm ubuntu**
  - **docker rm -f ubuntu**
  - **docker kill ubuntu**
- 도커 이미지 삭제
  - **docker rmi ubuntu**
  - **docker images**

도커 명령어(***)

- 이미지 파일 생성
  - **docker save -o ubuntu_img.tar ubuntu**
- 이미지 압축/해제
  - **gzip ubuntu_img.tar / bzip2 ubuntu_img.tar**
  - **gzip -d ubuntu_img.tar.gz / bzip2 -d ubuntu_img.tar.bz2**
- 이미지 삭제
  - **docker rmi ubuntu**
- 파일에서 이미지 로드
  - **docker load -i ubuntu_img.tar**
  - **docker images**
    - **이미지ID 확인**
- 이미지 태그 지정
  - **docker tag 이미지ID ubuntu**

컨테이너의 IP주소 알아내기

- 일반 우분투에서는
  - **ifconfig**
- 도커의 컨테이너에서는
  - **docker inspect ubuntu | grep “IPAddress”**
  -
  - **docker cp ./packt.jpg tensor:/root/**

도커 run 명령어 옵션

- **p**(publish) : 포트 노출
- **d**(detach)/**--detach** : 서버형 실행
- **e**(env)/**--env** : 환경변수 설정
- **i**(interactive) : 표준입력 열어두기
- **t**(tty) : 터미널 인터페이스
- **v**(volume) : 호스트 디렉토리 연결
- **w**(workdir) : 작업디렉토리 설정
- **l**(link)/**--link** : 컨테이너 연결

도커 명령어

- **search**(검색) ubuntu
- **pull**(다운로드) ubuntu:latest
- **run**(이미지->컨테이너생성), **exec**(컨테이너 쉘명령어 수행)
- **ps**(프로세스 리스트)/attach(컨테이너 접속)
- **stop/restart/kill/rm**(컨테이너 삭제)/**pause/unpause**
- **images**(이미지리스트)/**rmi**(이미지삭제)
- **commit**(컨테이너->이미지생성)
- **history**(이미지 변경사항내역)
- **diff**(이미지와 컨테이너사이의 변경내역조회)
- **inspect**(컨테이너/이미지의 세부정보 조회)
- **tag**(이미지 새로운 태그지정)

apt-get update & install

- 컨테이너에 접속한 상태에서
  - **apt-get update**
  - 이 명령을 수행해야 apt-get install 명령어를 사용할 수 있다
  - 바로 apt-get install을 사용하면 에러 발생
- 일반 우분투에서는 기본 설치되어 있는 nano(기본 에디터)가 도커기반 이미지에서는 깔려 있지 않음
  - **apt-get install nano**
- 설치 후에는 커맨드라인에서
  - **nano**라고 입력하면 나도 실행

이미지 생성

- 도커 이미지 생성방법
  - 컨테이너에 새로운 내용을 추가/변경한 후 commit해 이미지 생성
    - **docker diff ubuntu**(컨테이너 변경사항 확인)
    - **docker commit -m “test” -a “sjha” ubuntu ubuntu_nano**
    - **docker images**

이미지 생성

- 도커 이미지 생성방법
  - Dockerfile을 수행시켜 새로운 이미지 생성
    - 우분투 이미지에 Dockerfile 입력 후 수행
    - **docker build --tag=ubuntu_nano .**
    - **docker images**
    - **docker history ubuntu_nano**(이미지 변경사항 확인)

```
FROM ubuntu:latest
MAINTAINER Seokjae Ha <sjha72@gmail.com>

RUN apt-get update
RUN apt-get install nano
ENV TERM=xterm
```

Dockerfile 기초

- **FROM**
  - 도커 이미지 생성할 때 사용할 기본 이미지를 지정
  - 만약 해당 이미지가 없으면 서버 저장소(repository)에서 다운로드 받는다.
- **MAINTAINER**
  - 이미지를 생성한 사람에 대한 기본 정보표시

명령어 수행

- **RUN**
  - FROM에서 지정한 기본 이미지 위에 명령 수행해 새로운 이미지 생성
    - 우분투 최신 패치 수행
      - **RUN apt-get update**
    - 우분투 wget 패키지 설치
      - **RUN apt-get install wget**
- **CMD**
  - 컨테이너가 수행될 때 지정된 명령어/명령/스크립트 파일 실행
  - Dockerfile에서 한 번만 가능
    - **CMD [“echo $PATH”]**
- **ENTRYPOINT**
  - CMD와 거의 같으나 컨테이너 생성(run)이나 시작(start)될 때 실행
    - **ENTRYPOINT [“/sample.sh”]**

환경변수 설정

- 일반적인 우분투의 경우
  - 홈디렉토리의 ~/.bashrc나 ~/.profile에
    - export sample=/sample 과 같이 지정한 후
    - source ~/bashrc 나 source ~/.profile로 환경변수에 반영한다
- 도커에서는
  - 환경변수를 지정하려면
    - **docker run --env sample=/sample --name=ubuntu ubuntu**
  - Dockefile
    - **ENV sample=/sample**
- 환경변수 확인
  - **echo $sample**

포트 노출

- 컨테이너의 포트와 호스트의 포트를 연결
- 컨테이너의 80번포트와 호스트의 80번 포트를 연결
  - 외부에서 80번 포트로 접근하면 컨테이너의 80번 포트로 연결 (포트포워딩)
  - **docker run -p 80:80--name=ubuntu ubuntu**
- Dockerfile
  - **expose 80**
  - 컨테이너의 80번 포트를 외부에 노출한다
  - -p 옵션과 같이 사용

파일을 이미지에 추가

- 호스트의 파일을 이미지 생성시 추가(복사)
- Dockerfile
  - **ADD ~/sample.txt /sample.txt**
    - 호스트의 ~/sample.txt 파일을 컨테이너의 /에 추가(복사)
  - 압축파일을 지정할 경우 압축을 풀어 추가
  - URL을 지정할 경우에는 압축해제 없이 추가됨

명령 수행할 사용자/폴더 지정

- RUN/CMD/ENTRYPOINT 수행하기 전 사용자계정 지정
  - **USER sample**
    - sample 사용자로 변경
- RUN/CMD/ENTRYPOINT 수행하기 전 폴더 지정
  - **WORKDIR ~/sample**
    - ~/sample폴더로 변경해 아래 명령을 수행하라

볼륨연결

- 컨테이너의 폴더와 호스트의 물리폴더 간의 연결
- 물리 폴더 ~[홈디렉토리]/Downloads를 컨테이너의 /download 폴더로 연결
  - **docker run --name=ubuntu ubuntu -v ~/Downloads:/sample**
- Dockerfile
  - **VOLUME /sample**
  - **VOLUME [“/data”, “/sample”]**
  - 해당 디렉토리는 컨테이너 폴더가 아닌 호스트의 물리폴더로 저장
  - **-v** 옵션과 같이 사용

도커 컨테이너간 연결

- 컨테이너간 상호연결 설정
- mysql 다운로드
  - **docker pull mysql**
- mysql 컨테이너 실행(서버모드)
  - **docker run -d -e MYSQL_ROOT_PASSWORD=kitri**
    --name=mysql mysql
- 우분투 컨테이너 실행(연결)
  - **docker run --name ubuntu -d --link mysql:mysql ubuntu**

파이썬(Python) 언어는

- 파이썬 튜토리얼
  https://wikidocs.net/52

- 귀도 반 로섬(Guido van Rossum), 1991
- Strong-typed vs. Weak-typed
- Compiler vs. Interpreter
- Object-oriented

- 플랫폼 독립적이며
  - **인터프리터식, 객체지향적, 동적 타이핑(dynamically typed)** 대화형 언어
- 구현체(Implementation)
  - **CPython**, Jython, IronPython, PyPy

파이썬 라이브러리

- Ipython: 인터렉티브 python kernel
- numpy: 수치배열 연산
- Scipy: 과학 공학 계산에 필요한 툴박스
- Matplotlib: 매트랩 스타일의 plot 생성
- pandas: R과 같은 statistical package (통계)
- sympy: 심볼릭연산. 즉 도함수를 쉽게 그릴 수 있다.
- scikit-learn: machine learning package (deep learning은 아직없음)

- http://goodtogreate.tistory.com/entry/Data-Science를-위한-Python-package-꾸러미들

IPython & Jupyter Notebook

- IPython
  - https://ipython.org/
  - A powerful interactive shell.
  - A kernel for Jupyter.
- Jupyter Notebook
  - http://jupyter.org/
  - a web application that allows you to create and share documents that contain live code, equations, visualizations and explanatory text

IPython 사용법

- 자동완성기능
  - 탭<tabl>키를 친다
- 오브젝트 뒤에 ?를 입력하고 실행(Run)하면
  - 오브젝트에 대한 정보출력
- 매직명령어
  - %run
    - 외부 스크립트 파일을 실행
  - %hist
    - 명령어 히스토리를 출력
  - %time
    - 실행 시간을 출력

http://yujuwon.tistory.com/entry/ipython-사용하기

텐서(Tensor)란

- 텐서플로우의 기본자료구조
- 데이터 플로우 그래프에서 간선(edge)
- 다차원 배열이나 리스트로 구성
- Tensor
  - rank(차원)
  - shape(행과 열의 길이)
  - type(데이터 형식)

Tensorflow 프로그램 구조

1. 텐서플로우 라이브러리 로딩
2. 데이터를 텐서로 변환
3. 세션생성
4. run() 호출

Jupyter Notebook 실행 및 도커보드 실행하기

- 도커 컨테이너 실행하기

  docker run -it -p 8888:8888 -p 6006:6006 --name test_tensor b.gcr.io/tensorflow/tensorflow-full /run_jupyter.sh

- 로그 폴더 생성

  mkdir /tmp/tensorflowlogs

- 도커보드 실행하기

  tensorboard --logdir=./logs --host 0.0.0.0

회귀, 군집, 분류

- 회귀(Regression)
  - 지도학습
  - http://bcho.tistory.com/1141
- KNN(K-Nearnest Neighbor)
  - 회귀와 분류에 사용
- 군집화(Aggregation)
  - 비지도학습
  - K-Means 알고리즘 사용

MNIST 데이터로 학습하기

- Input_data.py 다운로드

https://tensorflow.googlesource.com/tensorflow/+/master/tensorflow/examples/tutorials/mnist/input_data.py

- 저장한 후 컨테이너에 복사

  docker cp ./input_data.py tensor:/root/

```
import sys
sys.path.append("/root/")
import input_data
mnistData = input_data.read_data_sets("MNIST_data/", one_hot=True)
```

http://yujuwon.tistory.com/entry/TENSOR-FLOW-MNIST-인식하기

Gradient Descent(경사하강법) 방법

- 비용함수를 최소화하는 값을 찾을 때 사용
- 볼록한(convex)타입의 경우에만 적용이 가능

Activation Function

- tanh
  - sigmoid 함수를 재활용하기 위한 함수. sigmoid의 범위를 -1에서 1로 넓혔다.
- ReLU
  - max(0, x)처럼 음수에 대해서만 0으로 처리하는 함수
- Leaky ReLU
  - ReLU 함수의 변형으로 음수에 대해 1/10로 값을 줄여서 사용하는 함수
- ELU
  - ReLU를 0이 아닌 다른 값을 기준으로 사용하는 함수
- maxout
  - 두 개의 W와 b 중에서 큰 값이 나온 것을 사용하는 함수

Binary Classification

- Logistic Regression 의 다른 이름
- Softmax Regression
  - 2개 이상의 그룹을 나눌 수 있도록 한 확장
  - cf. HardMax(최대값-통계용어)
  - 여러 개의 decision boundary를 사용
- One-Hot 인코딩
  - 소프트 맥스에서 제일 중요한 요소만 남기고 모두 제거
  - argmax() 함수
    - Tensorflow

분류(Classification)

- 회귀곡선을 기준으로 두 개의 영역으로 구분가능
- 예외적인 값들이 많이 나타난다면?
  - 로지스틱 회귀(Logistic Regression) 사용
  - 시그모이드(Sigmoid)

Deep NN(Neural Network)의 특징

- 임의의 모양의 패턴을 분류할 수 있슴
- 노드가 많을 수록,은닉계층이 많을 수록
- 활성화함수는 Sigmoid를 ReLu로 대체
- Overfitting의 경우 조심
  - DropOut을 사용
  - 한 epoch에 일부를 제외하고 학습

텐서플로 딥러닝

- CNN(Convolutional Neural Network)
  - 합성곱 신경망
  - 이미지/음성 인식에 강점
  - 지역수용영역(local receptive fields) - 합성곱(convolution) - 풀링(pooling)
  - Local connectivity
  - 공유 가중치- 공유 편향
  - 압축 특징 지도
  - 경사 하강법/역전파 알고리즘으로 학습
  - MNIST 이미지 분류

텐서플로 딥러닝

- RNN(Recurrent Neural Network)
  - 순환 신경망
  - 모두 같은 매개변수를 사용
  - LSTM
  - 자연어처리/음성인식에 강점
    - 다음 문자 예측, 문장에서 다음 단어 예측
  - 학습시간이 많이 걸림