Week 1. Introduction

Siyeong Lee
DSDL, Sogang University

Outline

1. Introduction
   a. 이번 세미나의 목표
2. 전체적인 Process
3. 부가적인 도구
   a. Library
   b. Git
4. 해야 할 일

Introduction

- 이번 세미나의 목표
  - Main: 간단한 학습까지 가능한 Deep Neural network Library 구축
- 세부목표
  - 일단은 많은 코드 작성 기회!
  - Github와 같이 다같이 코드 공유할 수 있는 기회!
  - Deep learning의 처음부터 끝까지..
    - Backpropagation을 짜보지 않았으면 Deep Learning을 피상적으로 이해하는 것!
  - Convolution layer의 성격 분석
- Issue: 잦은 토론/질문

전체적인 프로세스

- 1월: 부분 / 2월: 전체
- 1월
  - 1월에는 가장 기본이 되는 부분 부분에 대한 구현
  - Convolution layer와 Transposed Convolution layer에 대한 구현
  - Merge layer를 통한 Residual 구조 만들기
- 2월
  - Classification에서의 혁명: Inception v3, v4/ Xception
  - Object Detection으로는 어떻게 바꿔야 할까?
  - Visualization은 어떻게 하지?

진행 시 주의사항

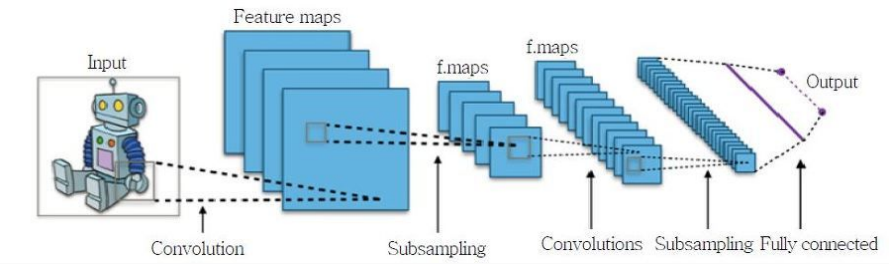
- 다같이 진행할 때 가장 중요한 점
  - 끊임없는 의사소통
  - 주석달기
  - 토론
- 문서화 작업
  - 단순하게 구글 Docs로 이루어진…
  - 양식 없이 내용만 타이핑 되어도 좋음

Convolutional Neural Network

- End-to-End / Black box
- 가장 기본적인 문제: Image Classification
  - Conv. Layer 들의 조합로 해결해보자
  - Conv. Layer를 더욱 돋보이게 하는 pooling 기법 (영상 내에서의 invarient)
    - Hinton이 언급했다시피, 이것은 영상 이해 관점에서 방해 요소로 하지만 계산량도 줄이고 성능은 향상되고!
  - Loss function의 진화 (Cross entropy: KL divergence)

- 이산확률변수의 경우: $D_{\mathrm{KL}}(P\|Q) = \sum_i P(i) \log \frac{P(i)}{Q(i)}$

CNN



추론할 때

f(x;w,b)

학습할 때

f(x;w,b)

원하는 w,b를 찾는게 목표

: Backpropation

Deep Learning은 진화 중!

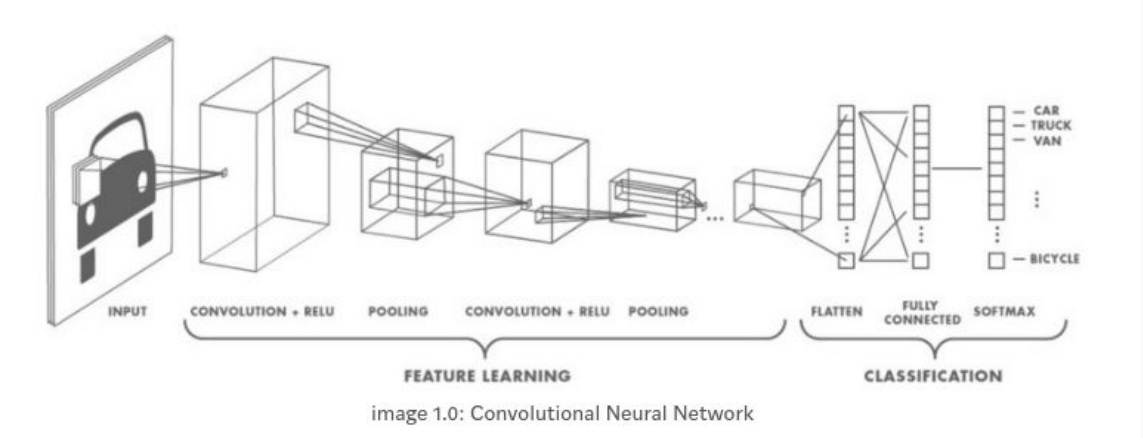
- CNN 모델과는 다른 CapsNet
  - 

image 1.0: Convolutional Neural Network

하지만 아직은....

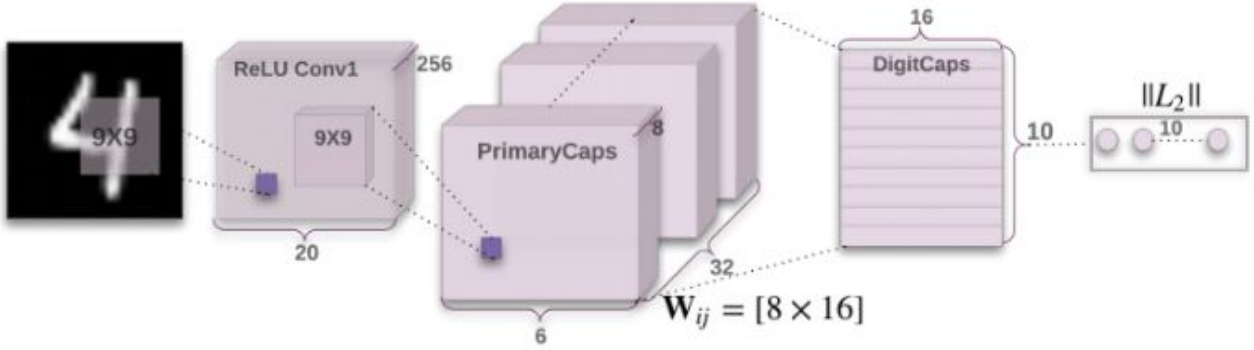
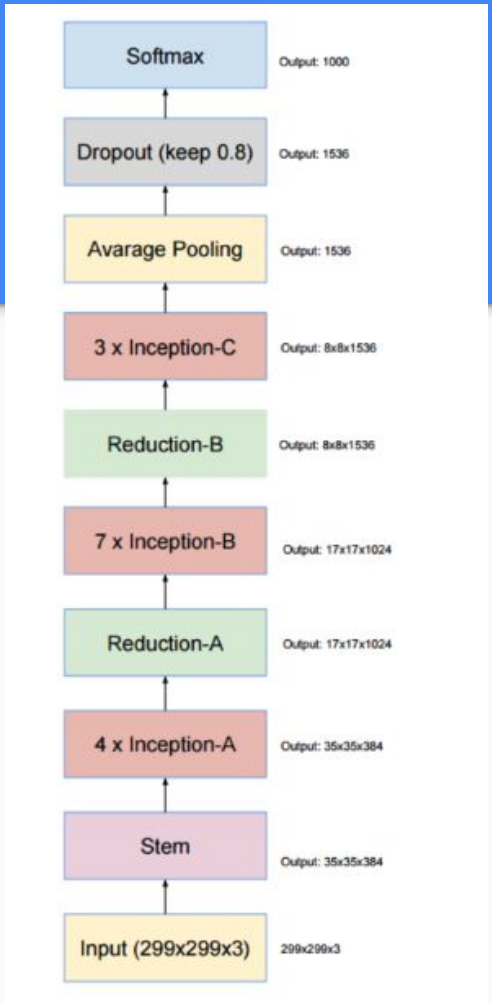


image 3.0: CapsNet Architecture

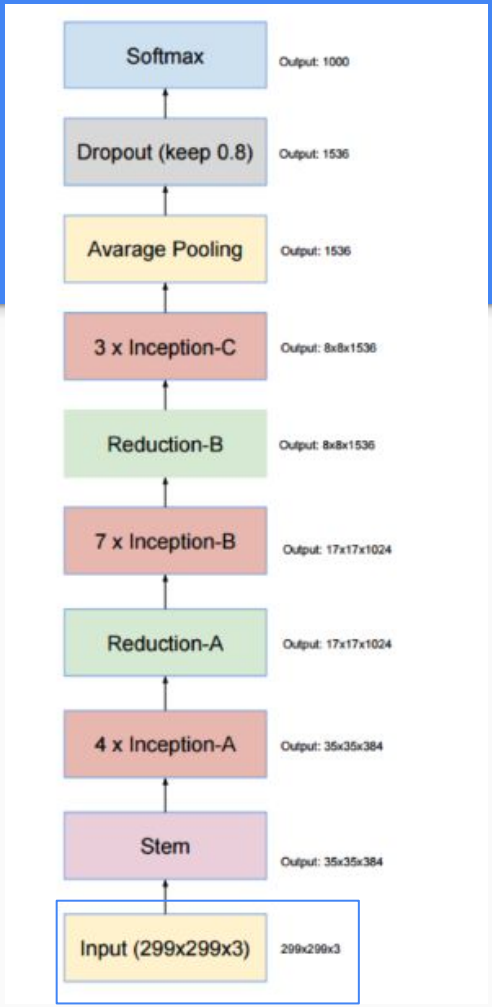
Neural Network와 Process

- 구현하려고 하는 model
  - Google Inception version 1, 2, 3 중 하나
  - 옆의 그림은 Version 4



Neural Network와 Process

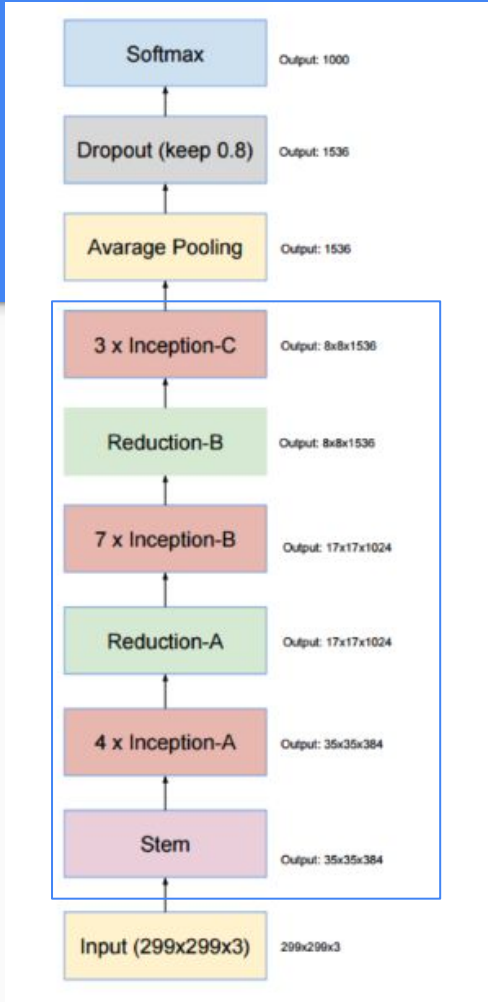
1. Input/output
    a. 이미지 데이터를 C로 받기 위해서는?
        i. 직접 짜자!
            1. Darknet-lib
            2. 아니면 특정 타입의 영상만 PPM
        ii. 아니다!
            1. OpenCV



Neural Network와 Process

1. Convolution layer
   a. 당연히 짜야지....

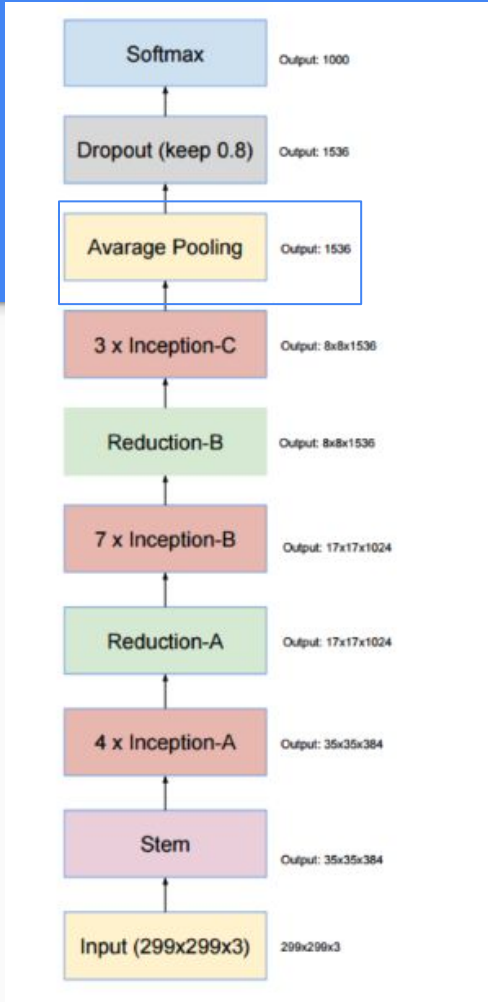
2. Transposed Convolution layer
   a. 학습은 잘 되고 있는지 볼 수 있으면 좋을 텐데



Neural Network와 Process

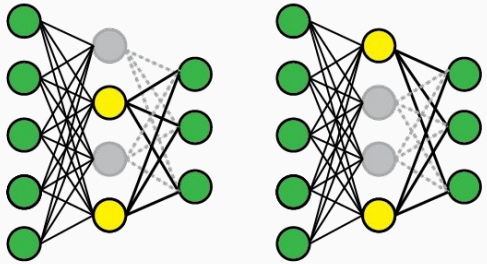
1. Merge Layer
   a. 그 유명한 Residual model을 위하여

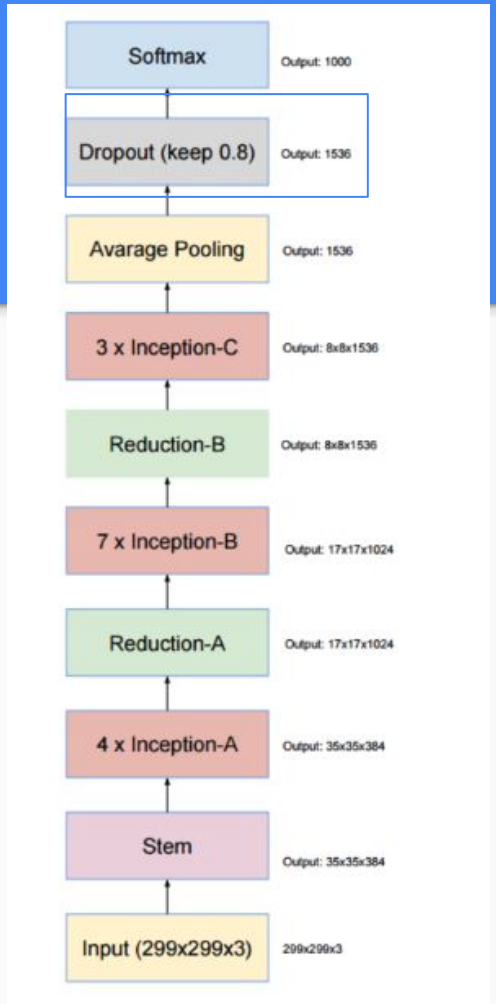
2. Pooling Layer
   a. CNN을 Master-piece로 만들어 준
   b. 하지만 약점이라고 하더라..



Neural Network와 Process

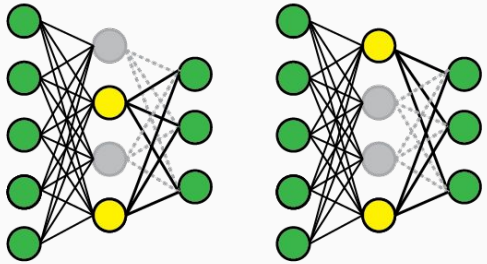
1. Dropout layer
   a. 이건 생각 좀 다 같이 해보자…
   b. Batch Normalization과는 철학이 다른 알고리즘
      i. Background에는 다양한 모델이 숨겨져 있도록
      ii. 모든 weight가 sparse 하도록
         1. 활성화되면 업데이트에 참여하니까!

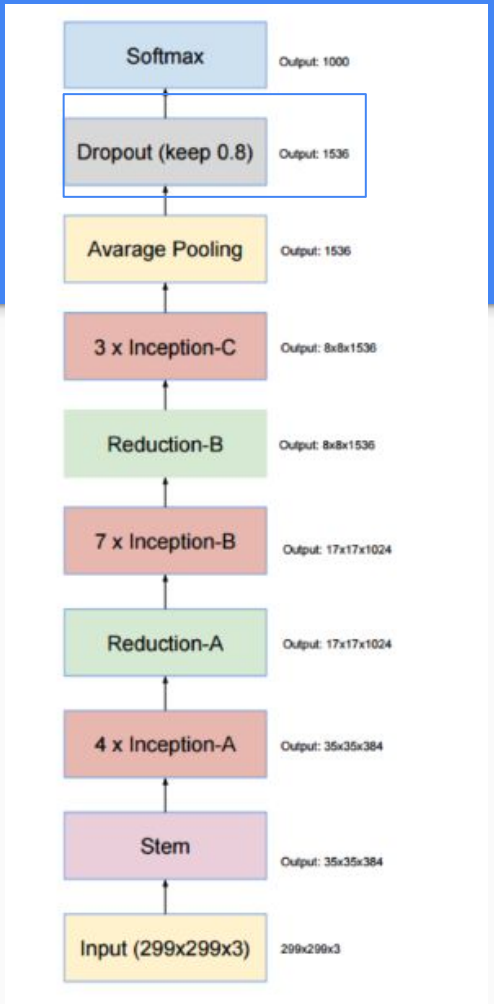




Neural Network와 Process

1. Dropout layer
    a. 해당 layer는 BN에 의해 묻혔다가....
        i. SELU activation layer로 다시 또?
        ii. 역시 아직 기본이 되는 Regularization 기법
            1. Pruning의 시작

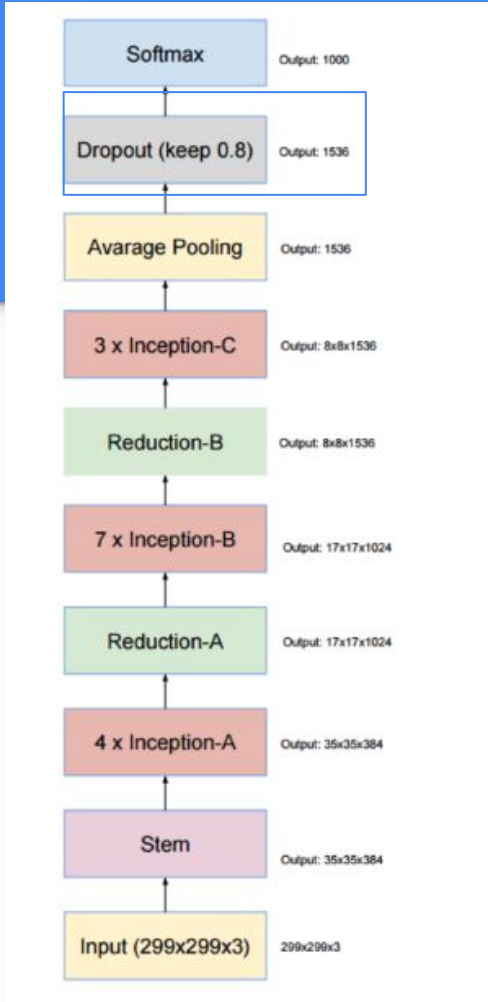




- 이산확률변수의 경우: $D_{\mathrm{KL}}(P\|Q) = \sum_{i} P(i) \log \frac{P(i)}{Q(i)}$

Neural Network와 Process

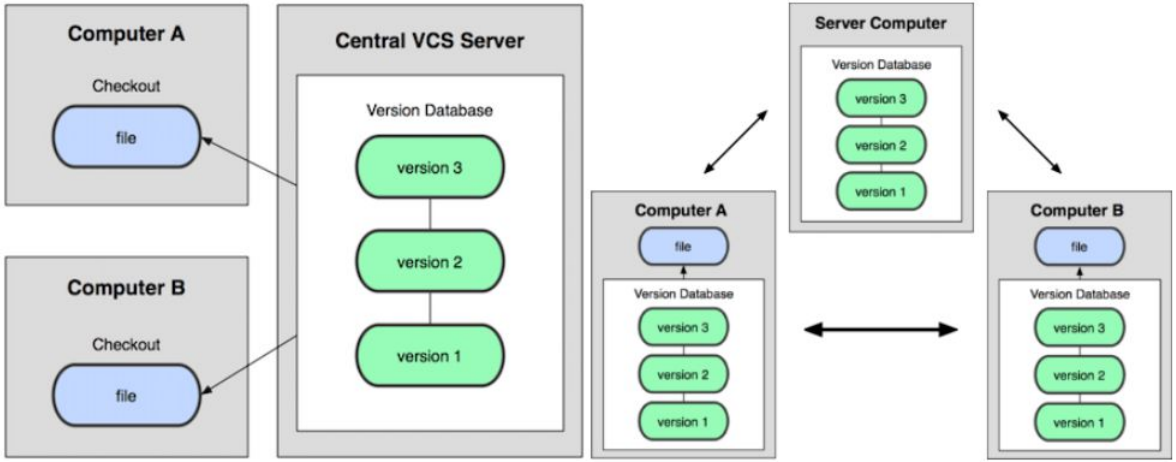
1. Loss function
   a. 가장 기본적인 Entropy function
2. 이를 학습시기키 위한 Back-propagation



Git이란?

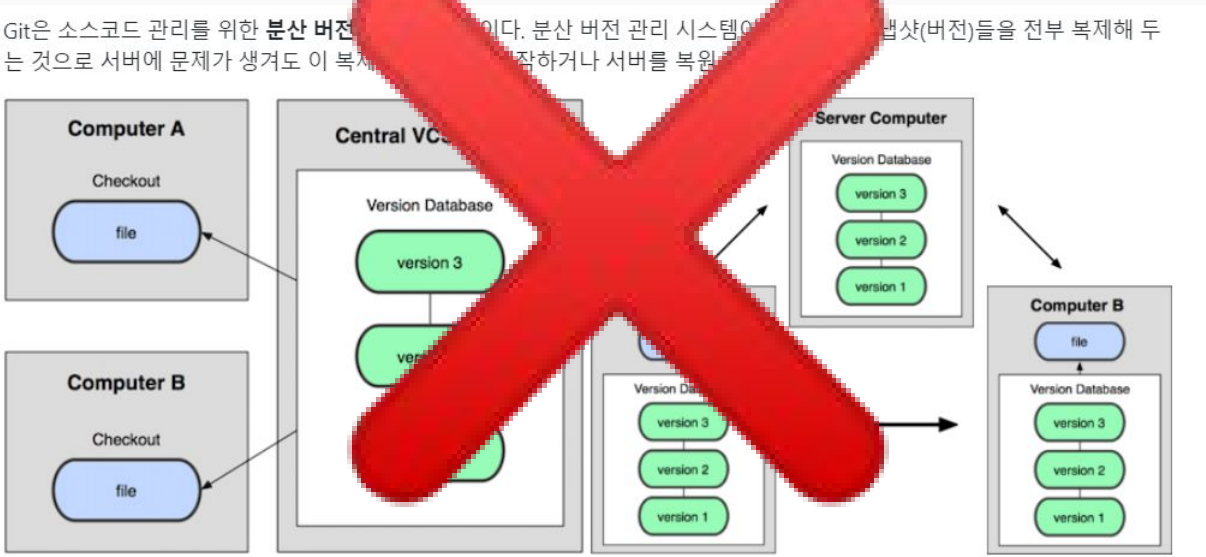
- http://rogerdudler.github.io/git-guide/index.ko.html

Git은 소스코드 관리를 위한 **분산 버전 관리 시스템** 이다. 분산 버전 관리 시스템이란 파일의 스냅샷(버전)들을 전부 복제해 두는 것으로 서버에 문제가 생겨도 이 복제물로 작업을 시작하거나 서버를 복원 할 수 있다.



Git이란?

- http://rogerdudler.github.io/git-guide/index.ko.html



Git

- 새로운 저장소 만들기
  - 폴더를 만들고, 폴더 안에서 아래 명령을 실행!

```
git init
```

  - 새로운 git 저장소가 만들어짐!
  - 해당 폴더를 git을 통해 관리하겠다는 뜻

Git

- 저장소 받아오기
  - 로컬 저장소를 clone

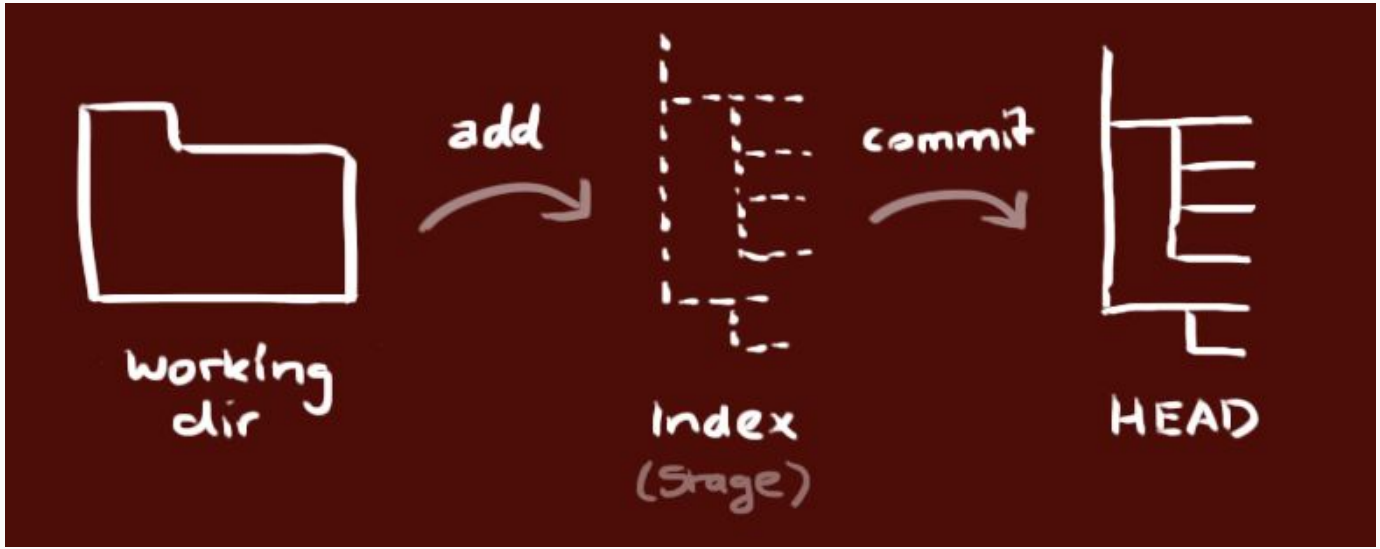
```
git init /로컬/저장소/경로
```

  - 원격 서버의 저장소를 복제하려면 아래 명령을 실행

```
git clone 사용자명@호스트:/원격/저장소/경로
```

Git

- 작업의 흐름



Git

- 일단 Working dir에서 Index로 추가
  - 변경된 사항이 뭔지 알려주기

```
git add <파일 이름>
```

```
git add *
```

```
git add .
```

Git

- 그럼 일단 local git에 올리기 위해서는

```
git commit -m “처음이당”
```

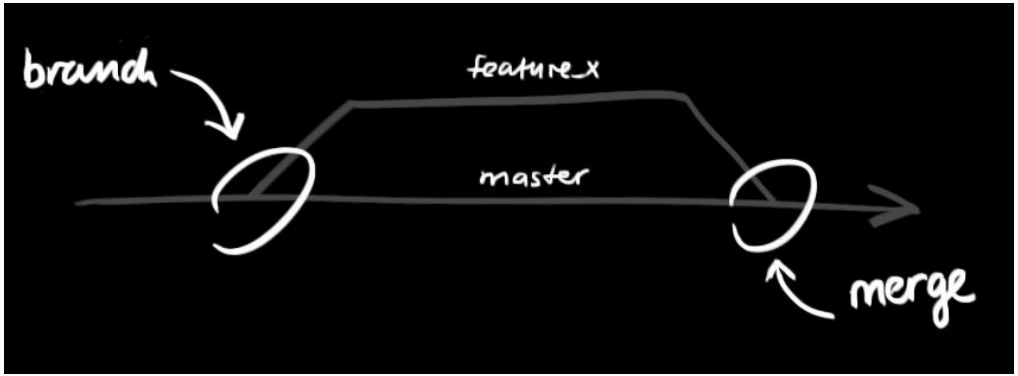
- 이걸 github로 보내자! (clone 했다면 알고 있지만 모르면!)

```
git push origin master
```

```
git remote add origin <서버 주소>
```

Git

- 가지(branch) 치기
  - 안전하게 격리된 상태에서 무언가 만들 때 사용!
  - 맨 처음 가지를 master 가지
  - 다른 가지를 만들어서 개발을 진행하고, 이후 master로 돌아와 병합하는 과정



Git

git push origin <가지 이름>

- 가지 만들기

```
git checkout -b feature_x
```

- 가지 바꾸기

```
git checkout master
```

- 가지 지우기

```
git branch -d feature_x
```

Git

- merge
  - 로컬 저장소를 원격 저장소에 맞춰 갱신하려면

```
git pull
```

- 다른 가지에 있는 변경 내용을 현재 가지에 병합하려면

```
git merge <가지 이름>
```

Git

이렇게 충돌이 발생하면, git이 알려주는 파일의 충돌 부분을

여러분이 직접 수정해서 병합이 가능하도록 해야 하죠.

충돌을 해결했다면, 아래 명령으로 git에게

아까의 파일을 병합하라고 알려주세요.

```
git add <파일 이름>
```

변경 내용을 병합하기 전에, 어떻게 바뀌었는지 비교해볼 수도 있어요.

```
git diff <원래 가지> <비교 대상 가지>
```

Git

- 로컬 변경 내용 되돌리기
  - 로컬의 변경 내용을 되돌리기

```
git checkout -- <파일 이름>
```

만약, 로컬에 있는 모든 변경 내용과 확정본을 포기하려면,

아래 명령으로 원격 저장소의 최신 이력을 가져오고,

로컬 master 가지가 저 이력을 가리키도록 할 수 있어요.

```
git fetch origin
```

```
git reset --hard origin/master
```

Git - Tutorial

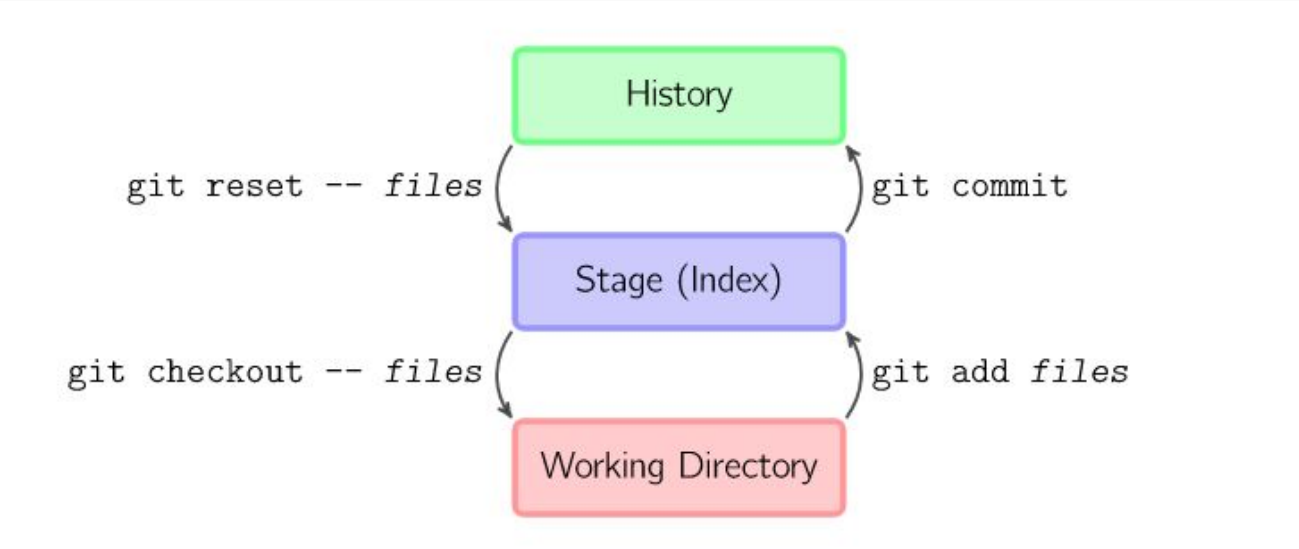
- 아 몰랑~!
  - 무슨 소리인지....

- Tutorial

  http://onlywei.github.io/explain-git-with-d3/#

Git - Tutorial

- 일단 명심해야 할 것!



Git - Tutorial

- Fetch
  - 중앙 저장소의 소스를 로컬 저장소로 가져온다! 그러나 현재 작업중인 소스들을 변경하는 Merge 작업을 하지는 않는다
- Pull
  - 중앙 저장소의 소스를 로컬 저장소로 가져온다! 또한 현재 작업중인 소스들의 Merge 작업까지 통합하여 수행한다

Git - Tutorial

- Local git에서의 시각화
  - gitk!!

해야 할 일

- 세미나 몇 시간 동안 진행될지?
- 앞으로의 진행 방향
  - 환경 구축은 어디서?
    - Ubuntu… : Server ID 발급

해야 할 일

Git 주소: https://github.com/orgs/DSDL-DeepNetwork

관련 Blog: https://dsdl-deepnetwork.github.io/

- Git 가입 후
  - bbiyackbbiyack@gmail.com 으로 아이디 보내기

우리의 Github 구조(Open 되어 있어요)

- Deep_Lib: 실질적인 코드가 구성되는 Repo.
- 2018_Winter_Storage: 서로 참고가 될 만한 자료를 공유하는 Repo.
  - 발표자료도 이리로 공유!!
- DSDL-DeepNetwork.github.io: 진행 관련된 사항 공유 및 전체 프로세스 공유 blog.