

# VINZIP shop business enhancing project

CNN & YOLO를 활용한 제품 분류 및 로고 식별

# Agenda

1

## 주제선정

주제소개, 주제선정 이유

2

## 기술소개

CNN, YOLO

3

## 모델구현

작업 프로세스, 데이터 분석/수집, 모델 설계/구현

4

## 결론

프로젝트 결과, 프로젝트 의의

# 1. 주제선정

## 01 주제설명

### VINZIP project란,

CNN과 YOLO 기술을 활용하여 VINZIP 온라인 쇼핑몰(구제샵)의 업무절차를 간편화하는 것을 목표로 합니다. 학습시킨 모델을 통해 **제품의 종류를 분류**하고, 브랜드의 logo를 식별하여 **브랜드를 자동 분류**합니다.

#### 특징

- CNN 알고리즘을 활용해 제품을 outer, top, bottom 으로 분류합니다.
- YOLO Object Detection을 사용해 학습시킨 로고를 찾아 내고 이를 통해 제품의 브랜드를 알아냅니다.
- 웹스크롤링과 Label\_img 프로그램을 사용해 학습데이터를 확보하고 tensorflow를 이용해 학습했습니다.
- predict 결과에 파이썬 코드를 이용해서 제품을 자동으로 적절한 디렉토리로 분류합니다.



outer  
top  
bottom

adidas  
fila  
nike  
polo  
⋮

# 1. 주제선정

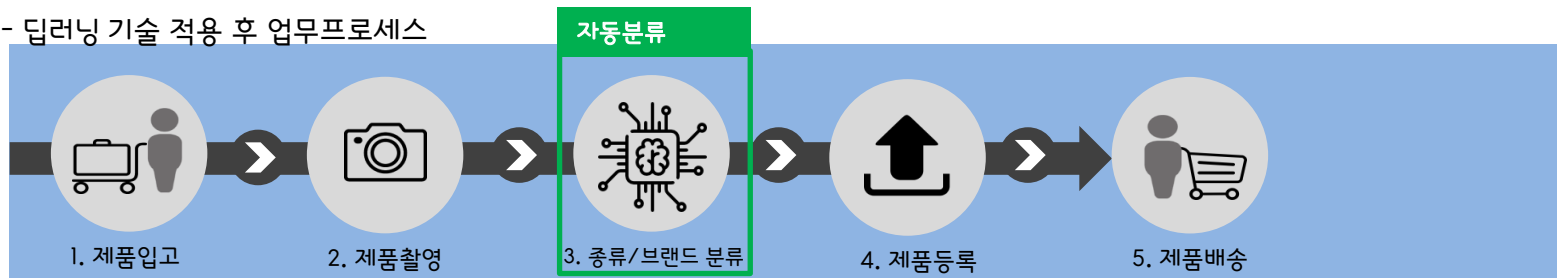
## 02 주제선정 이유

딥러닝 기술을 쇼핑몰의 업무 프로세스에 접목하여 도움을 줄 수 있을지 궁금해서 이 프로젝트를 수행하게 되었습니다. Vinzip 쇼핑몰의 업무 프로세스를 파악하여 **시간이 많이 소요되는 작업을 컴퓨터가 대신**할 수 있도록 구현했습니다.

- 기존의 업무프로세스



- 딥러닝 기술 적용 후 업무프로세스



# Agenda

1

## 주제선정

주제소개, 주제선정 이유

2

## 기술소개

CNN, YOLO

3

## 모델구현

작업 프로세스, 데이터 분석/수집, 모델 설계/구현

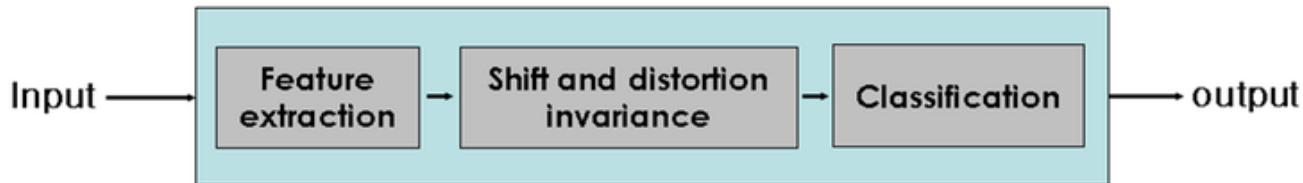
4

## 결론

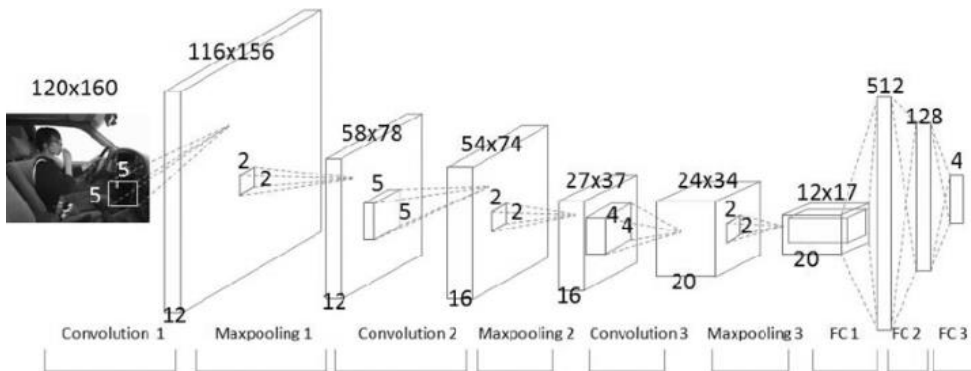
프로젝트 결과, 프로젝트 의의

## 2. 기술소개

### CNN이란?



CNN(Convolutional Neural Network)은 이미지의 **공간 정보를 유지**하면서 인접 이미지와의 특징을 효과적으로 인식하고 강조하는 방식으로 이미지의 **특징을 추출하는 부분(Conv)**과 **이미지를 분류하는 부분(Fc)**으로 구성되는 구조입니다. 특징 추출 영역은 Filter를 사용하여 공유 파라미터 수를 최소화하면서 이미지의 특징을 찾는 Convolution 레이어와 특징을 강화하고 모으는 Pooling 레이어로 구성됩니다.



## 2. 기술소개

### 01 CNN

CNN(Convolutional Neural Network)은 기존 Fully Connected Neural Network와 비교하여 다음과 같은 차별성을 갖습니다. Cnn은 제품의 종류(outer, top, bottom)를 **분류**할 때 사용했습니다.

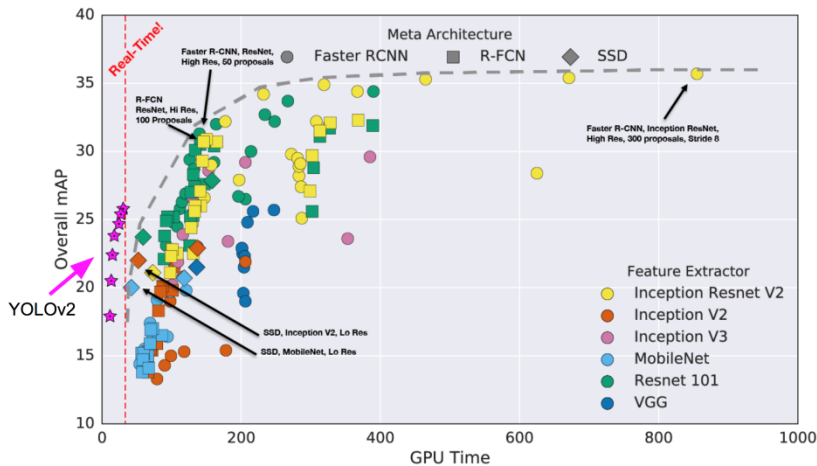
- 각 레이어의 입출력 데이터의 형상 유지
- 이미지의 공간 정보를 유지하면서 인접 이미지와의 특징을 효과적으로 인식
- 복수의 필터로 이미지의 특징 추출 및 학습
- 추출한 이미지의 특징을 모으고 강화하는 Pooling 레이어
- 필터를 공유 파라미터로 사용하기 때문에, 일반 인공 신경망과 비교하여 학습 파라미터가 매우 적음

## 2. 기술소개

02 YOLO

### YOLO란?

YOLO는 'You Only Look Once'의 약자로, Deep Learning Real Time **Object Detection Algorithm** 중 하나입니다. 기존의 기술들보다 빠른 속도를 가지지만 정확도는 더 낮습니다.



Huang, Jonathan, et al. "Speed/accuracy trade-offs for modern convolutional object detectors." *arXiv preprint arXiv:1611.10012* (2016).

속도	정확도
YOLO > Fast/Faster R-CNN, SSD	Fast/Faster R-CNN, SSD > YOLO

이 프로젝트는 darknet의 tensorflow 버전인 darkflow를 사용했습니다.  
(darknet은 YOLO를 C로 짠 오픈소스입니다.)



## 2. 기술소개

02 YOLO

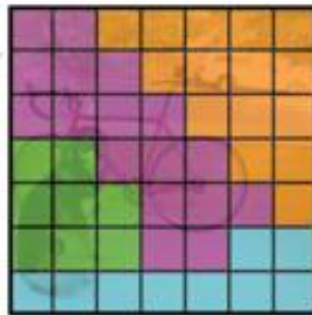
YOLO는 다음과 같이 동작합니다.



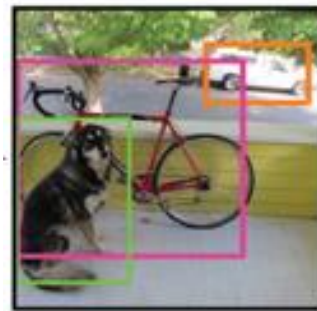
Object detecting을 위해 입력한 이미지를  $S \times S$  개의 그리드로 분할하고, 그리드의 신뢰도를 계산한다.



신뢰도는 그리드 내 객체 인식 시 정확성을 반영하는데, 초기에는 객체 인식과는 동떨어진 경계 상자가 설정되지만, 신뢰도를 계산하여 경계 상자의 위치를 조정함으로써, 가장 높은 정확성을 가지는 경계 상자를 얻을 수 있다.



그리드에 객체 포함 여부를 계산하기 위해, 객체 클래스 점수를 계산한다. 이 결과로  $S \times S \times N$  객체가 예측된다. 이 그리드의 대부분은 낮은 신뢰도를 가지는데, 신뢰도를 높이기 위해 주변의 그리드를 합칠 수 있다. 이후, 임계값을 30%로 설정하면, 많은 그리드는 제외된다.



최종적으로 각 객체에 대해 정확히 분류하는 것을 확인할 수 있다. YOLO는 단순한 처리로 속도가 매우 빠르다. 기존 다른 real-time 비전 기술과 비교할 때, 2배 정도 높은 성능을 보이는데, 이미지 전체를 한 번에 바라보는 방식으로 클래스를 분별하기 때문이다.

# Agenda

1

## 주제선정

주제소개, 주제선정 이유

2

## 기술소개

CNN, YOLO

3

## 모델구현

작업 프로세스, 데이터 분석/수집, 모델 설계/구현

4

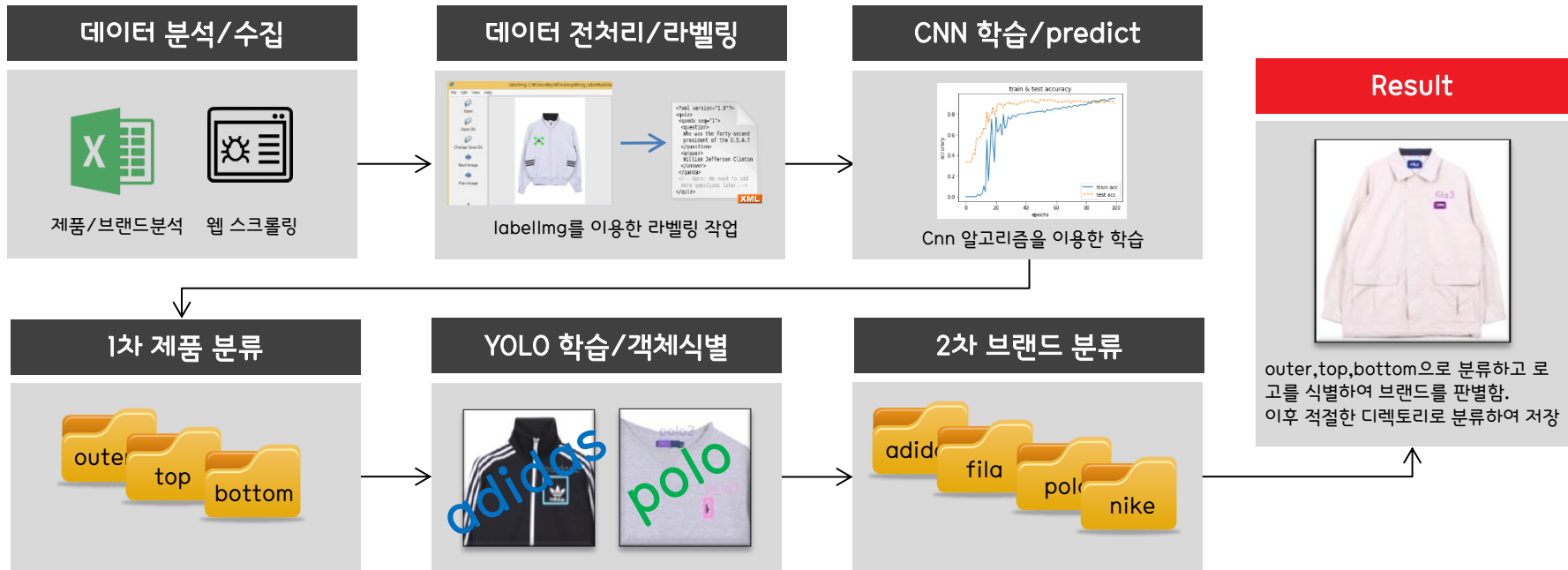
## 결론

프로젝트 결과, 프로젝트 의의

### 3. 모델구현

#### 01 작업 프로세스

이번 프로젝트의 작업 프로세스의 흐름을 요약하면 다음과 같습니다.



### 3. 모델구현

#### 02 데이터 분석/수집

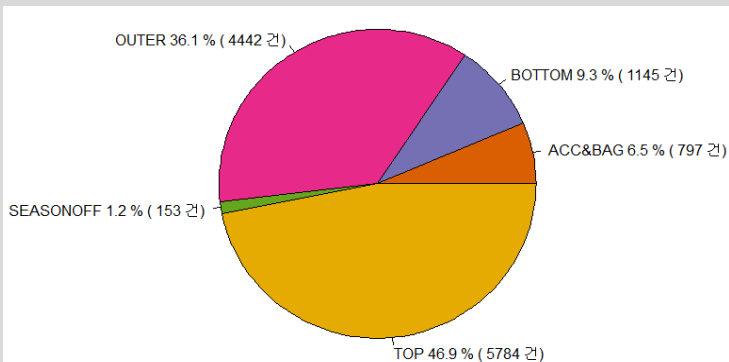
R 프로그래밍을 통해 Vinzip 온라인 쇼핑몰 제품 데이터를 분석 및 시각화하여 해당 쇼핑몰에 알맞은 데이터를 선별적으로 수집했습니다.

#### 제품 분류



- 웹 스크롤링을 통해 해당 쇼핑몰의 제품 이미지를 수집합니다. 총 8000장의 제품 이미지를 수집했습니다.
- 다양한 사이즈의 사진을 파이썬 코드를 통해 64x64 사이즈로 정제합니다.

#### 제품 분석 시각화



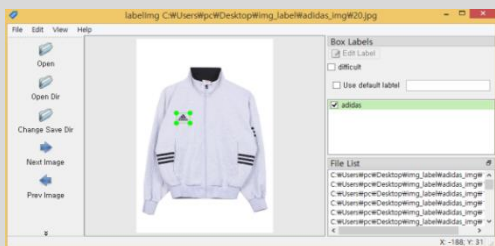
- 제품의 종류는 top, outer, bottom이 주력이기 때문에 3가지 종류의 데이터에 대해서 학습을 진행했습니다.

### 3. 모델구현

## 02 데이터 분석/수집

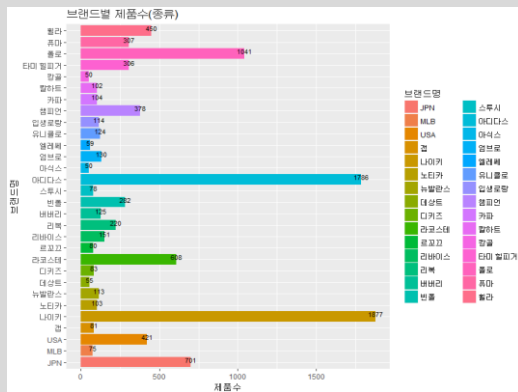
브랜드 식별 학습을 위해 vinzip 온라인 쇼핑몰에서 각각의 브랜드 검색 결과를 웹 스크롤링으로 수집했습니다. 그 후 labellmg 프로그램을 이용한 라벨링 작업을 수행했습니다.

## 브랜드 로고 식별



- **LabelImg**이란 객체의 사각형 **경계(bounding box)**를 따서 labeling 및 다양한 포맷으로 저장할 수 있는 유틸리티 프로그램입니다.
- LabelImg를 통해 **xml**이 생성되고 xml 파일은 이미지에서 로고 각각의 좌표와 라벨명을 가지고 있습니다.

## 브랜드 분석 시각화



- 브랜드의 종류가 다양해서 상위 브랜드를 선정하여 **브랜드 로고 식별**을 하기 위한 라벨링 작업을 진행했습니다.

### 3. 모델구현

#### 02 데이터 분석/수집

브랜드당 여러 개의 로고가 존재하기 때문에, **각각의 모양에 대한 라벨을 부여**하는 것이 더 높은 인식률을 보일 것으로 판단했습니다. 라벨링 작업 시 로고마다 고유한 라벨을 부여했습니다.

하나의 라벨로 통일 (=모양이 다양해서 학습이 잘 안됨)

모양이 각각 다르기 때문에 라벨의 특성을 판단하기 어려울 것이라 판단



adidas®



= adidas



모양마다 각각의 라벨 부여



+

결과 출력 전에 전처리

~~adidas~~

adidas



### 3. 모델구현

02 데이터 분석/수집





데이터 정제 작업으로 **사이즈 재조정**과 **손상된 파일제거**를 일괄적으로 진행했습니다.



### 3. 모델구현

#### 03 모델 설계/구현

웹 스크롤링과 labellmg 프로그램은 window 환경에서 진행했습니다. 윈도우에서 수집한 데이터를 [google drive](#) 공유문서함에 업로드하고 [colab 환경](#)에서 이 데이터를 이용해 학습을 진행했습니다.

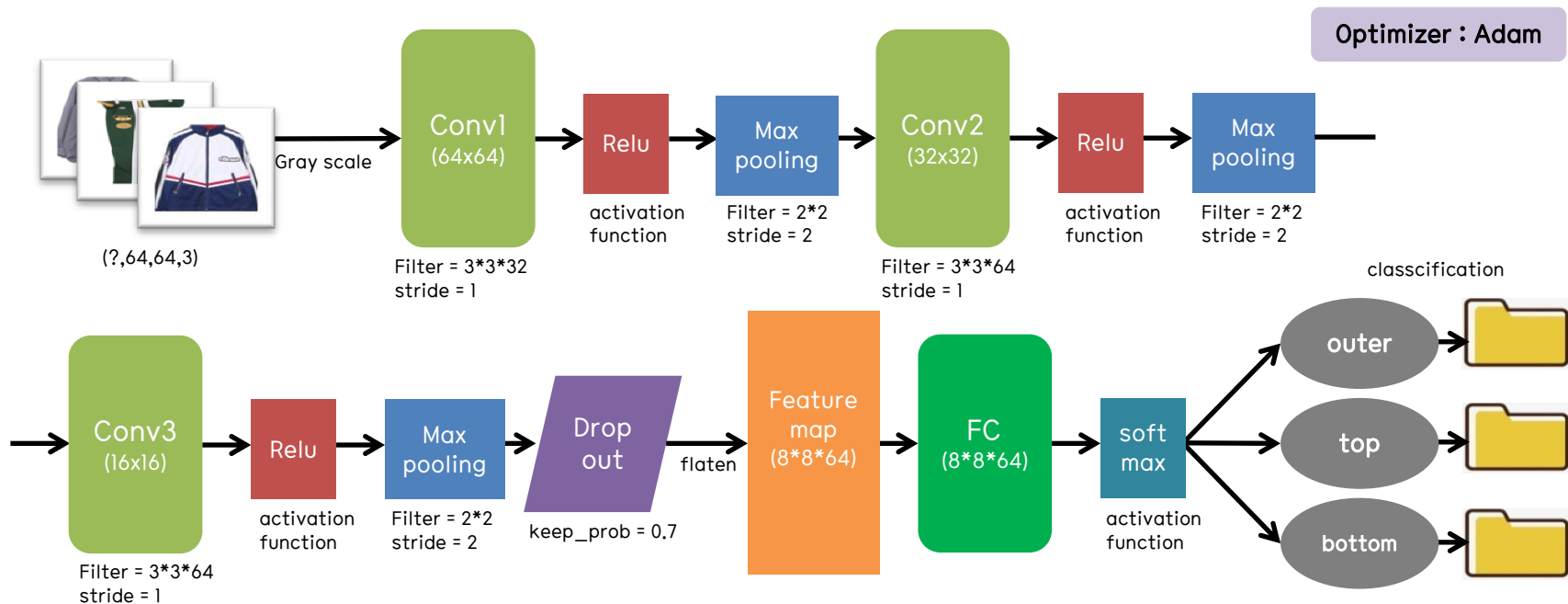
운영체제	GPU 환경	언어 환경	플랫폼 / 소프트웨어
 Window7  Ubuntu17.10	  	 python Python 3.5.6	 Anaconda Python 3.5 ver.  OpenCV 3.43.18  Google drive + colab



### 3. 모델구현

#### 03 모델 설계/구현

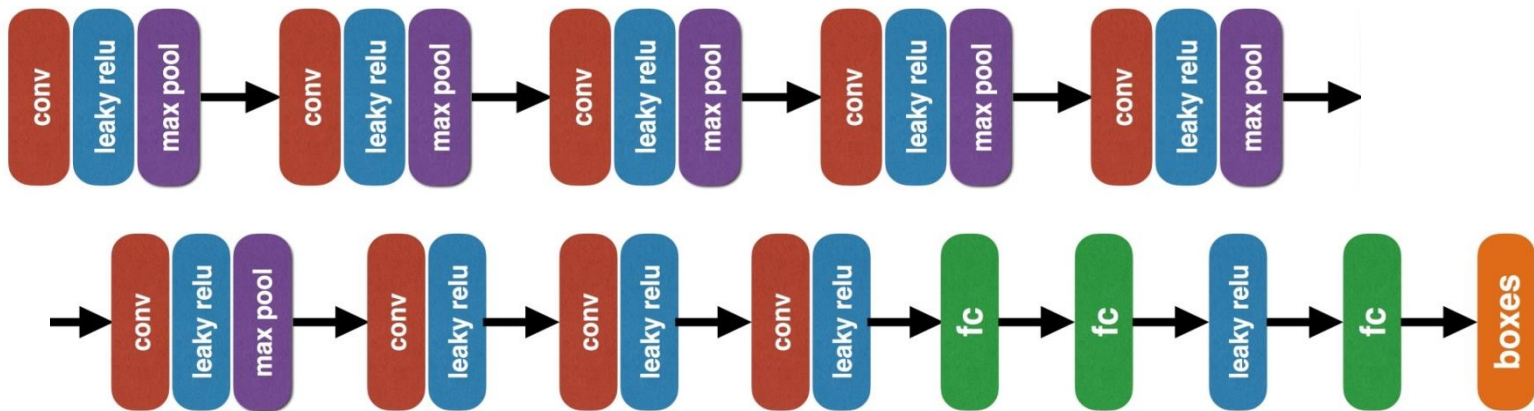
제품 종류(outer, top, bottom) 분류는 **cnn 알고리즘**을 사용했습니다. Google colab를 이용해 google drive에 올려둔 데이터로 학습을 진행했습니다.



### 3. 모델구현

#### 03 모델 설계/구현

제품에서 브랜드 로고를 식별하는 모델의 학습은 **tiny-yolo 모델**을 이용하여 학습했습니다.  
일반 Yolo 모델은 층이 깊고 무거워서 batch size를 줄였음에도 불구하고 1060GTX 3g에서 학습 시키지 못했기 때문에 이보다 가볍고 빠르게 학습할 수 있다고 알려진 tiny-yolo를 선택했습니다.  
tiny-yolo의 아키텍처는 다음과 같습니다.



### 3. 모델구현

#### 03 모델 설계/구현

옵티마이저는 Adam을 사용하여 학습했습니다. 비교적 가벼운 tiny-yolo 모델을 사용했지만 꽤 많은 학습 시간이 소요됐습니다.

```
Parsing ./cfg/tiny-yolo_brand.cfg
Loading None ...
Finished in 0.0s
```

```
Building net ...
```

Source	Train?	Layer description	Output size
		input	(?, 416, 416, 3)
Init	Yep!	conv 3x3p1_1 +bnorm leaky	(?, 416, 416, 16)
Load	Yep!	maxp 2x2p0_2	(?, 208, 208, 16)
Init	Yep!	conv 3x3p1_1 +bnorm leaky	(?, 208, 208, 32)
Load	Yep!	maxp 2x2p0_2	(?, 104, 104, 32)
Init	Yep!	conv 3x3p1_1 +bnorm leaky	(?, 104, 104, 64)
Load	Yep!	maxp 2x2p0_2	(?, 52, 52, 64)
Init	Yep!	conv 3x3p1_1 +bnorm leaky	(?, 52, 52, 128)
Load	Yep!	maxp 2x2p0_2	(?, 26, 26, 128)
Init	Yep!	conv 3x3p1_1 +bnorm leaky	(?, 26, 26, 256)
Load	Yep!	maxp 2x2p0_2	(?, 13, 13, 256)
Init	Yep!	conv 3x3p1_1 +bnorm leaky	(?, 13, 13, 512)
Load	Yep!	maxp 2x2p0_1	(?, 13, 13, 512)
Init	Yep!	conv 3x3p1_1 +bnorm leaky	(?, 13, 13, 1024)
Init	Yep!	conv 3x3p1_1 +bnorm leaky	(?, 13, 13, 1024)
Init	Yep!	conv 1x1p0_1 linear	(?, 13, 13, 85)

```
step 59988 - loss 0.7478644251823425 - moving ave loss 0.9198392423780448
Finish 4999 epoch(es)
step 59989 - loss 0.6737640500068665 - moving ave loss 0.895231723140927
step 59990 - loss 1.0117241144180298 - moving ave loss 0.9068809622686373
step 59991 - loss 1.2440237998962402 - moving ave loss 0.9405952460313977
step 59992 - loss 0.8834490776062012 - moving ave loss 0.934880629188878
step 59993 - loss 1.8876367807388306 - moving ave loss 1.0301562443438734
step 59994 - loss 0.7047627568244934 - moving ave loss 0.9976168955919354
step 59995 - loss 1.2095715999603271 - moving ave loss 1.0188123660287747
step 59996 - loss 1.1561477184295654 - moving ave loss 1.0325459012688538
step 59997 - loss 1.0389589071273804 - moving ave loss 1.0331872018547066
step 59998 - loss 1.015170693397522 - moving ave loss 1.0313855510089882
step 59999 - loss 0.5847681164741516 - moving ave loss 0.9867238075555046
step 60000 - loss 1.6674118041992188 - moving ave loss 1.054792607219876
Checkpoint at step 60000
Finish 5000 epoch(es)
```

An exception has occurred, use %tb to see the full traceback.

**SystemExit:** Training finished, exit.

Wall time: 9h 28min 29s

### 3. 모델구현

#### 03 모델 설계/구현

Cnn과 YOLO 모델의 주요 하이퍼파라미터는 다음과 같습니다.

Model	CNN
Activate function	relu
Learning rate	0.001
Batch	600
Optimizer	Adam
가중치 초기값	정규분포 난수, xavier
Batch normalization	use
Epoch	300
Pool	Max-pool

Model	Tiny-yolo
Activate function	Leaky-relu
Learning rate	1e-04
Batch	10
Optimizer	Adam
가중치 초기값	He
Batch normalization	use
Epoch	5000
Pool	Max-pool

# Agenda

1

## 주제선정

주제소개, 주제선정 이유

2

## 기술소개

CNN, YOLO

3

## 모델구현

작업 프로세스, 데이터 분석/수집, 모델 설계/구현

4

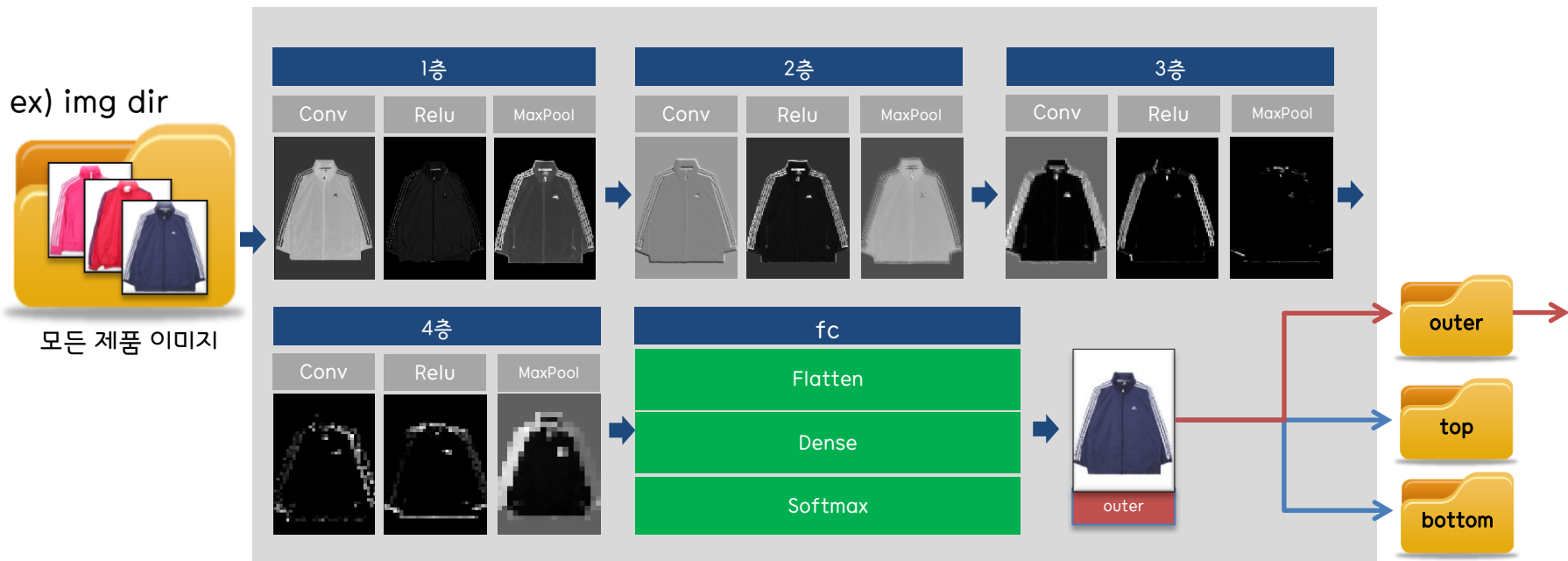
## 결론

프로젝트 결과, 프로젝트 의의

## 4. 결론

### 01 프로젝트 결과

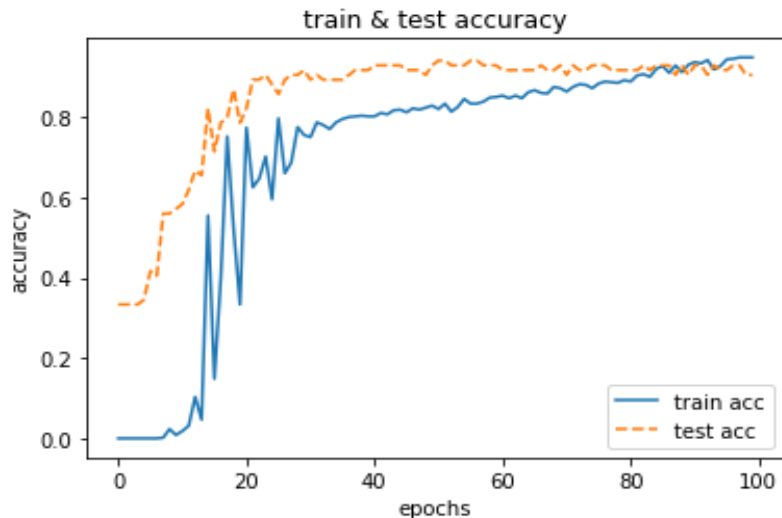
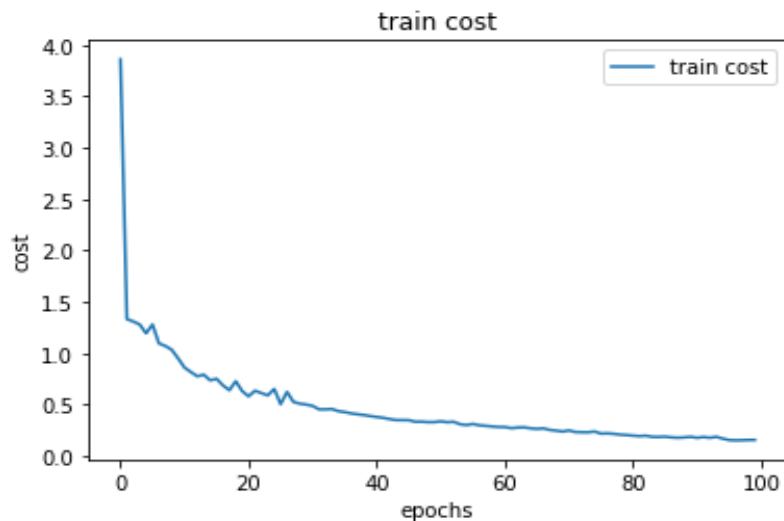
제품의 종류(outer, top, bottom)를 분류하기 위해 학습한 cnn 모델을 거쳐 적절한 폴더에 제품 이미지를 저장합니다.



## 4. 결론

### 01 프로젝트 결과

학습한 cnn 모델은 90%에 가까운 정확도를 보였습니다. (더 높은 정확도를 위해 신경망을 수정 중입니다)  
Cost function은 Classification 문제에서 자주 사용되는 **cross entropy 함수**를 사용했습니다.



## 4. 결론

### 01 프로젝트 결과

1차로 종류를 분류한 뒤, 2차로 브랜드 로고를 식별하여 제품의 브랜드를 분류하여 각각의 폴더에 제품 이미지를 저장합니다.





## 4. 결론

### 01 프로젝트 결과

tiny-yolo를 사용해 학습시킨 모델은 상표에 있는 로고도 식별할 수 있을 정도로 높은 식별력을 보였지만 간혹 잘못 식별하는 경우가 있었습니다. 더 높은 정확도를 위해선 더 많은 라벨링된 학습 데이터(xml)가 추가되어야 합니다.

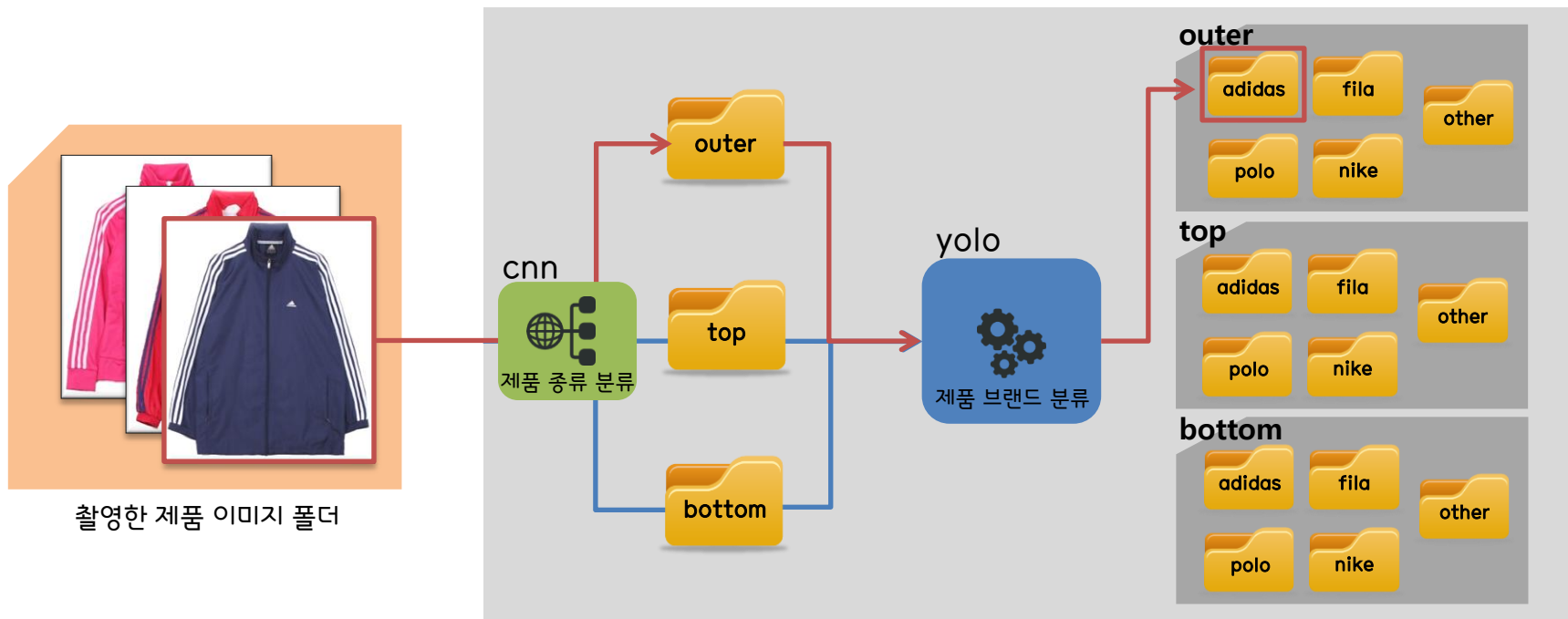


#### 실패



## 4. 결론

프로젝트의 전체 흐름은 다음과 같습니다.



## 4. 결론

### 02 프로젝트 의의



01

#### 실제 쇼핑몰 데이터 분석 및 시각화

실제로 서비스 중인 쇼핑몰의 데이터를 R, python으로 분석/시각화하여 필요한 데이터만 웹 스크롤링을 통해 수집하였음.

02

#### YOLO를 이용한 Object Detection

이미지 또는 영상에서 특정 객체를 식별하는 YOLO를 사용한 경험을 통해 앞으로 이미지 및 영상에서 분류와 식별 문제를 좀더 쉽게 해결할 수 있게 됨.

03

#### 다양한 환경에서 프로젝트 수행

Cnn 및 tiny-yolo 모델 학습을 google colab / drive를 이용해서 수행하였음. GPU 성능 한계를 느끼게 되어 더 좋은 GPU의 필요성을 느낌. 이 경험은 앞으로 다양한 환경에서 프로젝트를 원활히 수행할 수 있는 바탕이 될 것임.

**THANK YOU**