

Gopher's AI – research & develop

"핵심은 기존 서비스 대비 AI기술을 기반한 서비스라면 AI관련 내용과 데이터에 대한 내용이 구체적으로 표현했으면 하며, 또한 시장에 대한 수치화된 데이터는 있으나, 고객이 가지고 있는 문제점에 대한 객관적인 근거된 자료가 있었으면 함. 고객적인 측면에서 문제를 해결해나간다면 좋은 아이템이 될 것으로 보임"

이 문제의 해결책을 2가지로 분류하여 설명하자면

- 1. AI관련 구체적인 내용과 계획 (CTO 파트)
- 2. 시장의 수치화된 데이터 뿐만 아니라 고객이 현재 가진 문제점을 파악해야한다. (CMO)

1. 예시를 하나 들자면

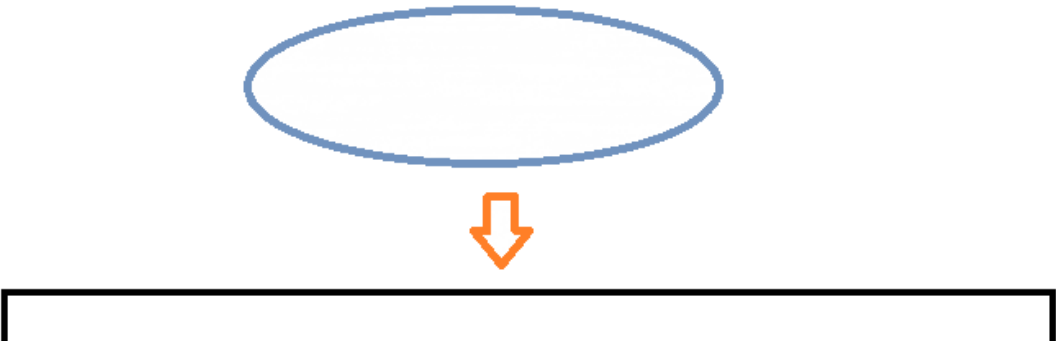
현재 open AI api는 혹은 chat gpt플러그인으로 생각 이상으로 많은 데이터를 불러오고 판단할 수 있다. 그러나 그 정보를 한번 걸러서 정확하고 본인이 사용하고 싶은 데이터로 분류하는 것이 관건이다. 이는 ai를 활용하는 데에 있어서 가장 중요한 정확도의 문제다.

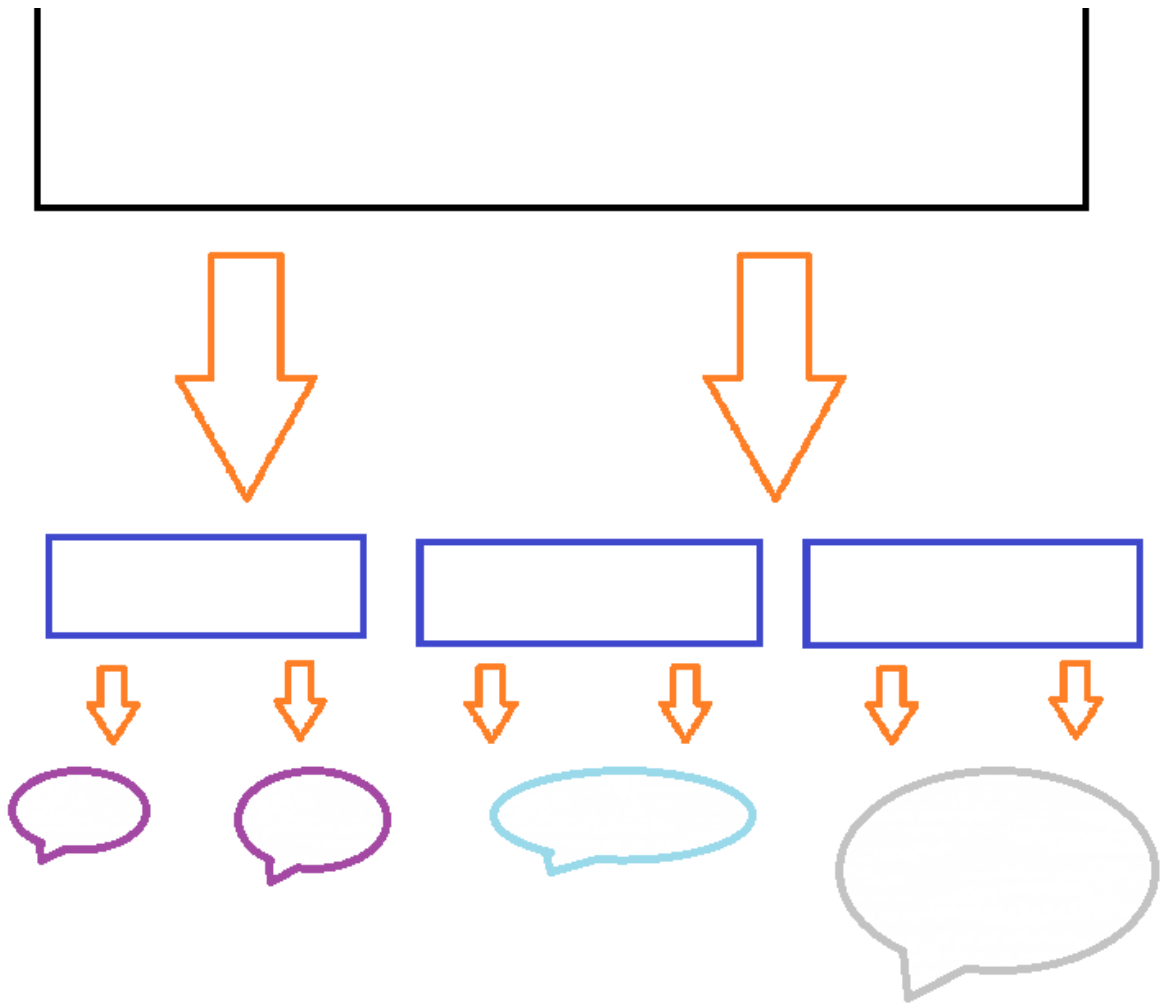
정보의 바다에서 ai가 긁어온 정보를 한번 정류하는 과정 즉 finding을 통해 본인이 원하는 데이터를 사용한 다. 이는 두가지 방법이 있다. 하나는 해당 분야의 전문가가 받아온 데이터를 직접 분류하는 방식이고

다른 하나는 다음 <그림 1> 그림과 같이

동그라미에는 고퍼
여기 큰 검은 박스에는 AI
파란 박스에 고퍼제작 AI 모델 1 2 3
말풍선에는 "나이키 신발 찾아줘" "~~ 상품 찾아줘" 대표님 맘대로 예시 넣기

<그림 1>





우리 고퍼에서 사용하게될 다른 자체 개발 ai이다. Pytoech를 사용하여 분류 ai에 관한 새로운 모델을 제작하여 학습 시켜 AI 아래 작은 AI가 여러개 작동되어 각 부서에 배치한다.그렇다면 더 정확한 정보들과 필터링 파인딩 된 정보들만 유저에게 제공한다. 현재 AI 시장에서 가장 큰 문제점이라 할 수 있는 정확도와 명확성 관련한 기술을 고퍼가 제공하므로 타사의 프로그램이나 제품출시자나 상품판매자의 입장에서 보다 효율적이다. 이해를 쉽게하기 위하여 예시 모델을 만들었습니다.

운동화 분류 AI 모델 예시입니다.

현재 사업자가 아니어 쿠팡 지마켓등의 쇼핑몰 api key를 발급받지 못하여 아쉽게도 google api를 무료로 사용하여 구글에 달린 이미지를 긁어 오도록 한다.

```
from google_images_search import GoogleImagesSearch
from PIL import Image
import os
import requests
from io import BytesIO

# Google API 키 및 사용자 고유 식별자(CX)를 입력하세요.
```

```

api_key = "AIzaSy8LZj6H7iMsn4xA8g1HX1hwktffEsDw8jc"
cx = "503bfa1d4b2b84f3c"

# 검색할 키워드 및 저장할 디렉토리 설정
search_term = "마디다스 신발"
save_directory = "C:\\Users\\halo0\\Desktop\\AI_test폴더"

# GoogleImagesSearch 객체 생성
gis = GoogleImagesSearch(api_key, cx)

# 검색 조건 설정
_search_params = {
    "q": search_term,
    "num": 50, # 한 번에 다운로드 받을 이미지 개수
    "imgSize": "large",
    "fileType": "jpg"
}

# 검색 실행
gis.search(search_params=_search_params)

# 디렉토리가 없으면 생성
if not os.path.exists(save_directory):
    os.makedirs(save_directory)

# 이미지 다운로드
for i, image in enumerate(gis.results()):
    try:
        # 이미지 데이터를 가져옵니다.
        response = requests.get(image.url)

        # 이미지 파일을 엽니다.
        img = Image.open(BytesIO(response.content))

        # 이미지 크기를 조정합니다 (선택 사항).
        img = img.resize((500, 500))

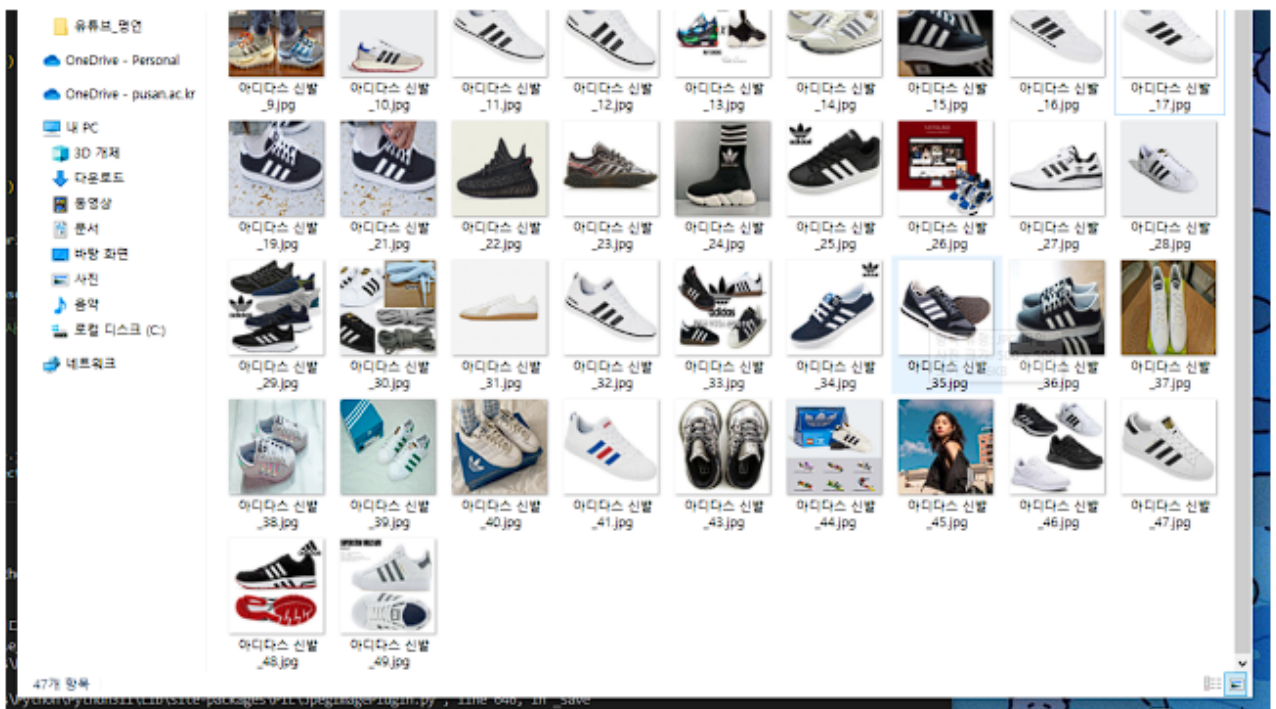
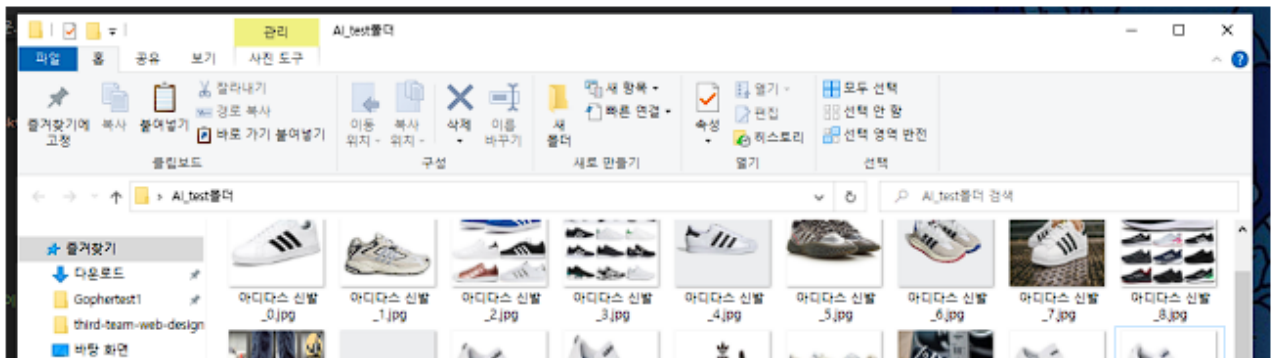
        # 이미지를 RGB 모드로 변환합니다.
        img = img.convert('RGB')

        # 이미지를 저장합니다.
        file_name = f"{search_term}_{i}.jpg"
        img.save(os.path.join(save_directory, file_name))
    except Exception as e:
        print(f"이미지 {i}를 저장하는 동안 오류 발생: {e}")

```

```
print("이미지 다운로드 완료")
```

이때 결과는 다음과 같다.



이때 우리는 mvp에서 구글에서가 아니라 쇼핑몰 사이트에서 링크와 해당링크에 있는 이미지 이렇게 두가지를 긁어 올 수 있도록 할 예정이다. (배열 사용 편하게 하려고 우선적으로 구글이미지로 가져왔습니다.)

동일한 방식으로 다른 브랜드의 신발 데이터도 가져오도록 합니다. (다음 AI 모델에서는 아디다스 나이키 뉴발란스 데이터를 활용)

현재 우리는 아디다스 운동화라는 특정 제품을 검색하여 쇼핑몰 상품들의 데이터를 가져왔지만

chatGPT 플러그인과 타사의 openAI api들을 고퍼페이지에 연동하여 단순히" 신발을 사려고 하는데 어디서 사지?" 라고 말하게 된다면 운동화 상품정보들과 구매처들의 데이터를 전부 고퍼의 db서버에 보내면 알고리즘이 큰 분류할 것이다.

- 부츠
- 구두
- 하이힐
- 나막신
- 모카신
- 샌들
- 운동화
- 실내화
- 슬리퍼
- 안전화],

이중 운동화를 선택하게 된다면 (농구화 등산화 보드화 골프화 배구화 트래킹화 런닝화 축구화 배드민턴화 캔버스화 컨버스 스니커즈) ...등의 정보들을 나열하는데 클릭 10대 20대 나이별 성별 색상 계절별 트렌드등의 분석을 통해 인기 상품 순서로 제품을 나열 할 것이다. 제품 분석에 있어서는 기존에 나온 분석AI및 알고리즘이 있지만 현재 우리 회사는 CMO가 상주 중이며 CMO와 트렌드 분석 알고리즘연구에 있다.

따라서 분석AI를 자체개발 하여 시장흐름에 맞는 어찌구 저찌구 할 예정 //부탁드립니다.,

현재 chatgpt는 자연어 처리에 있어 최고의 성능을 자랑하지만 구글 검색을 비롯한 웹 검색이 불가능 한 bot 이므로 이 소프트웨어의 기능을 확장 추가를 위해 고퍼의 플러그인을 제작에 chatgpt를 비롯한 다른 open ai api들을 특정 쇼핑몰의 데이터를 이용하여 "고퍼봇" pandarank에 판다봇과 유사한 시스템을 갖지만 상품추천및 소싱관련한 정보들을 추가로 판매자에게 줄 수 있도록 챗봇mvp를 제작중이다.

본론으로 돌아가 다음 운동화를 선택하여 등산화 골프화 스니커즈 등등의 제품중에서 스니커즈를 선택하여 쇼핑몰 data를 받아오게 된다면 이때 브랜드 마다 또한 분류가 나뉠 것이며 이런 쇼핑몰 data를 open AI 가 가져오는 과정에서 정확하지 않은 정보들을 가져 올 경우가 많다. 이때 고퍼가 개발한 AI 모델들을 사용하여 정보전달의 정확도를 높이려고 한다. 고퍼 분석에서 가장 중요한 것은 트렌드 분석에 있어 AI가 정확한 정보를 전달하기 위해 연구중이다.

고퍼의 핵심 기술중 하나인 'AI상품등록'

ai 상품등록 은상품을 사진을 찍어서 올리게 되면 자동으로 로그인된 판매자의 정보와 해당 제품이 인식이 되어 쿠팡 gmarket 네이버 쇼핑등의 판매 입고에 등록이 되도록 판매자들에게 판매 과정에 있어 시간과 인건비를 단축시킬 수 있다.이때 사용되는 기초적인 기술이 ai 제품분류 기술이다

예시로 스니커즈를 아디다스, 뉴발란스, 나이키 세가지 운동화중 아디다스 운동화를 판매하려고 한다.

```
1 import torch
2 import torch.nn as nn
3 import torch.optim as optim
4
5 import torchvision
6 from torchvision import datasets, models, transforms
```

```

7
8 import numpy as np
9 import time
10
11
12 device = torch.device("cuda:0" if torch.cuda.is_available() else "cpu") # device 객체

1 # 데이터셋을 불러올 때 사용할 변형(transformation) 객체 정의
2 transforms_train = transforms.Compose([
3     transforms.Resize((224, 224)),
4     transforms.RandomHorizontalFlip(), # 데이터 증진(augmentation)
5     transforms.ToTensor(),
6     transforms.Normalize([0.485, 0.456, 0.406], [0.229, 0.224, 0.225]) # 정규화(normalization)
7 ])
8
9 transforms_test = transforms.Compose([
10     transforms.Resize((224, 224)),
11     transforms.ToTensor(),
12     transforms.Normalize([0.485, 0.456, 0.406], [0.229, 0.224, 0.225])
13 ])
14
15 data_dir = './custom_dataset'
16 train_datasets = datasets.ImageFolder(os.path.join(data_dir, 'train'), transforms_train)
17 test_datasets = datasets.ImageFolder(os.path.join(data_dir, 'test'), transforms_test)
18
19 train_dataloader = torch.utils.data.DataLoader(train_datasets, batch_size=4, shuffle=True, num_workers=4)
20 test_dataloader = torch.utils.data.DataLoader(test_datasets, batch_size=4, shuffle=True, num_workers=4)
21
22 print('size of learned dataset:', len(train_datasets))
23 print('size of test dataset:', len(test_datasets))
24
25 class_names = train_datasets.classes
26 print('class:', class_names)
27

```

#파이토치를 이용하여 모듈생성

```

size of learned dataset: 112
size of test dataset: 37
class: ['addidas sneakers', 'newbalance sneakers', 'nike sneakers']

```

#데이터 클래스들을 세 종류로만 주었기 때문에 3개의 클래스생성됨

```

1 def imshow(input, title):
2     # torch.Tensor를 numpy 객체로 변환
3     input = input.numpy().transpose((1, 2, 0))
4     # 이미지 정규화 해제하기
5     mean = np.array([0.485, 0.456, 0.406])
6     std = np.array([0.229, 0.224, 0.225])
7     input = std * input + mean
8     input = np.clip(input, 0, 1)
9     # 이미지 출력
10    plt.imshow(input)
11    plt.title(title)
12    plt.show()
13
14
15 # 학습 데이터를 배치 단위로 불러오기
16 iterator = iter(train_dataloader)
17
18 # 현재 배치를 이용해 격자 형태의 이미지를 만들어 시각화
19 inputs, classes = next(iterator)
20 out = torchvision.utils.make_grid(inputs)

```

[illegible]

#WARNING:matplotlib.font_manager:findfont: Font family 'NanumBarunGothic' not found. 이 예러 메시지는 기능에서 오류가 아니라 addidas sneakers를 아디다스 스니커즈 라고 한글로 출력하고 싶어서 나눔바름고딕체를 사용하여 한글폰트를 이용하려는데 버전 문제로 나눔바름고딕체를 plt 할 수 없는 워닝 메시지입니다. ai구현에 전혀 문제 없음.

```
1 num_epochs = 50
2 model.train()
3 start_time = time.time()
4
5 # 전체 반복(epoch) 수 만큼 반복하며
6 for epoch in range(num_epochs):
7     running_loss = 0.
8     running_corrects = 0
9
10    # 배치 단위로 학습 데이터 불러오기
```



```

10 # 매시 단계도 적당 데이터 출력하기
11 for inputs, labels in train_dataloader:
12     inputs = inputs.to(device)
13     labels = labels.to(device)
14
15     # 모델에 입력(forward)하고 결과 계산
16     optimizer.zero_grad()
17     outputs = model(inputs)
18     _, preds = torch.max(outputs, 1)
19     loss = criterion(outputs, labels)
20
21     # 역전파를 통해 기울기(gradient) 계산 및 학습 진행
22     loss.backward()
23     optimizer.step()
24
25     running_loss += loss.item() * inputs.size(0)
26     running_corrects += torch.sum(preds == labels.data)
27
28 epoch_loss = running_loss / len(train_datasets)
29 epoch_acc = running_corrects / len(train_datasets) * 100.
30
31 # 학습 과정 중에 결과 출력
32 print('#{} Loss: {:.4f} Acc: {:.4f}% Time: {:.4f}s'.format(epoch, epoch_loss, epoch_acc, time.time() - start_time))

```

현재는 시간 문제로 50epoch 만큼 데이터 셋을 반복학습 시켰으나 추후 많은 량의 데이터를 다루게 되면 더 늘릴것.
이렇게 loss발생이 적어지고 정확도가 높아질 때 까지 반복학습을 시킵니다.

```

#0 Loss: 1.0441 Acc: 51.7857% Time: 8.3508s
#1 Loss: 0.7361 Acc: 65.1786% Time: 14.4687s
#2 Loss: 0.3988 Acc: 83.9286% Time: 22.8668s
#3 Loss: 0.3274 Acc: 89.2857% Time: 29.8827s
#4 Loss: 0.1863 Acc: 91.9643% Time: 38.1288s
#5 Loss: 0.1595 Acc: 94.6429% Time: 44.3320s
#6 Loss: 0.3313 Acc: 89.2857% Time: 52.2106s
#7 Loss: 0.4008 Acc: 87.5000% Time: 58.6621s
#8 Loss: 0.2026 Acc: 93.7500% Time: 66.2144s
#9 Loss: 0.4859 Acc: 83.9286% Time: 72.9593s
#10 Loss: 0.2969 Acc: 92.8571% Time: 80.1885s
#11 Loss: 0.1639 Acc: 94.6429% Time: 87.2651s
#12 Loss: 0.1567 Acc: 95.5357% Time: 94.5730s
#13 Loss: 0.1262 Acc: 97.3214% Time: 101.7126s
#14 Loss: 0.2037 Acc: 94.6429% Time: 108.7163s
#15 Loss: 0.0318 Acc: 100.0000% Time: 116.0509s
#16 Loss: 0.0740 Acc: 96.4286% Time: 122.8180s
#17 Loss: 0.1806 Acc: 92.8571% Time: 131.5704s
#18 Loss: 0.2804 Acc: 94.6429% Time: 139.1507s
#19 Loss: 0.0729 Acc: 98.2143% Time: 146.1386s
#20 Loss: 0.2068 Acc: 92.8571% Time: 153.3806s
#21 Loss: 0.1256 Acc: 95.5357% Time: 160.7485s
#22 Loss: 0.0668 Acc: 98.2143% Time: 167.9018s
#23 Loss: 0.0943 Acc: 95.5357% Time: 175.4975s
#24 Loss: 0.0331 Acc: 100.0000% Time: 182.4910s
#25 Loss: 0.0599 Acc: 98.2143% Time: 189.9121s
#26 Loss: 0.1132 Acc: 95.5357% Time: 196.6696s
#27 Loss: 0.1627 Acc: 94.6429% Time: 204.2341s
#28 Loss: 0.3285 Acc: 92.8571% Time: 210.5701s
#29 Loss: 0.1508 Acc: 96.4286% Time: 218.4380s
#30 Loss: 0.1112 Acc: 96.4286% Time: 224.6710s
#31 Loss: 0.0311 Acc: 100.0000% Time: 232.8900s
#32 Loss: 0.0584 Acc: 98.2143% Time: 239.0775s
#33 Loss: 0.0823 Acc: 96.4286% Time: 247.3835s
#34 Loss: 0.0630 Acc: 98.2143% Time: 253.5613s
#35 Loss: 0.1931 Acc: 94.6429% Time: 261.9337s
#36 Loss: 0.0838 Acc: 97.3214% Time: 268.2580s
#37 Loss: 0.0251 Acc: 99.1071% Time: 276.5997s
#38 Loss: 0.1645 Acc: 95.5357% Time: 282.7421s
#39 Loss: 0.2476 Acc: 92.8571% Time: 290.9785s
#40 Loss: 0.0792 Acc: 98.2143% Time: 297.2159s

```


#41 Loss: 0.0515 Acc: 98.2143% Time: 305.3641s
#42 Loss: 0.0163 Acc: 99.1071% Time: 311.5656s
#43 Loss: 0.0749 Acc: 98.2143% Time: 319.3575s
#44 Loss: 0.0395 Acc: 99.1071% Time: 326.1552s
#45 Loss: 0.1306 Acc: 95.5357% Time: 333.8457s
#46 Loss: 0.1033 Acc: 96.4286% Time: 340.5879s
#47 Loss: 0.0121 Acc: 100.0000% Time: 347.7327s
#48 Loss: 0.0056 Acc: 100.0000% Time: 354.7745s
#49 Loss: 0.0113 Acc: 100.0000% Time: 361.3723s

accuracy가 100% 아래로 돌때즈음 학습은 종료시키도록 합니다

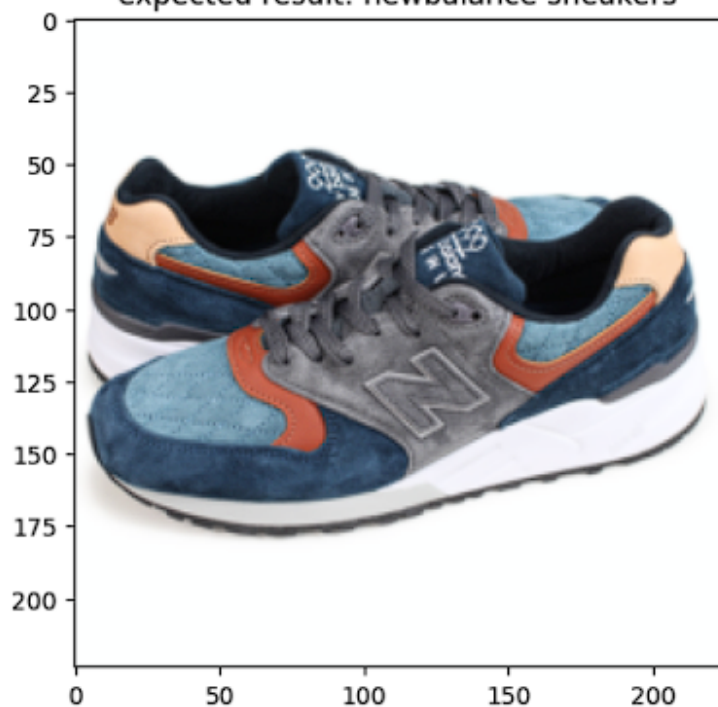
촬영하여 파일에 올린 신발이미지를 종류별로 분류하여 파일에 넣습니다. 그리고 다음 그림처럼 출력하도록 함.



WARNING: not suitable for use as a reference for the Foot classification (Mean: 0.451)



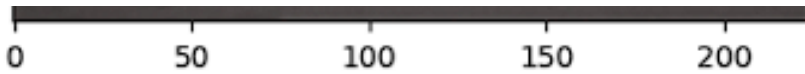
expected result: newbalance sneakers



다음은 아디다스 신발을 예시로 들겠습니다.

expected result: addidas sneakers





[Test Phase] Loss: 0.0940 Acc: 97.2973% Time: 6.9497s

이렇게 판매자가 제품의 사진을 찍어 고퍼 ai상품등록 페이지 파일넣기에 올리게 되면 판매하려는 제품을 ai가 판단하게 됩니다.

[제품인식]--->[신발]--->[운동화]--->[스니커즈]--->[아디다스]--->[아디다스 운동화 신발로 인식]

다음 예정된 AI2는 해당 상품의 사용 연령층 분석, 계절 분류, 제품 목적등의 기능또한 자연어 처리를 할 수 있도록 만든다

후에 → 쇼핑물 SEO 규격에 맞도록

EX)

[네이버 쇼핑물]

제목 : 아디다스 운동화 신발 아디다스

키워드: @10대 @청소년 @신발 @운동화 @아디다스 @여름

판매자정보 : 신발팔이 소년

id: SADFASDF11

해당 제품 브랜드 평균가 : 79000₩ (가격 설정은 알림 무시하면 브랜드 평균가로 아니면 어플에서 제품가격설정알림 받도록)

이런 규격을 따라 AI를 거쳐 고퍼 가 자동으로 판매자의 상품을 쇼핑물에 등록하여 줍니다. 이때 판매자에게 해당 상품의 트렌드와 키워드 분석 또한 충분히 구현가능합니다. 위의 결과를 참고하여



이때 자연어 처리는 기존에 나온 GPT 플러그인을 사용할 것이다. 실제로 우리 아디다스 운동화 신발과 현재 계절 연령층을 종합한 키워드를 만들어보자. 우리가 세부 AI를 거쳐서 나온 키워드 들이다 이때 GPT플러그인에게 요즘트렌드인 키워드들을 분석해 입력하게 되면 다음의 결과를 제공합니다. 우리는 다시 '슈퍼스타' '울트라 부스트' 'NMD'의 데이터를 받아

아디다스 슈퍼스타, 아디다스 울트라 부스트 아디다스 NMD를 쇼핑물에서 크롤링하여 비슷한 방식으로 AI에게 트렌드 분석을 맡기며 정확도 분석또한 위의 CNN모델처럼 동작하도록 합니다.

KA

10대, 청소년, 여름, 운동화, 아디다스, 신발 종류 알려줘

Share Prompt



아디다스는 전 세계적으로 유명한 스포츠 브랜드로, 여러 가지 신발 종류를 생산하고 있습니다. 특히 10대와 청소년들이 선호하는 여름 운동화를 소개해 드리겠습니다.

1. 슈퍼스타 (Superstar)

아디다스의 대표적인 스니커즈로, 클래식한 디자인과 편안한 착화감으로 10대와 청소년들에게 인기가 많습니다. 특히 흰색과 검은색 스

트라이프가 돋보이는 슈퍼스타는 여름에도 가볍게 신을 수 있는 아이템입니다.

2. 울트라부스트 (Ultra Boost)

아디다스의 울트라부스트는 편안한 착화감과 뛰어난 쿠션감으로 인기를 얻은 신발입니다. 여름에 가볍게 운동할 때에도 뛰어난 통기성과 지지력을 제공해 줍니다.

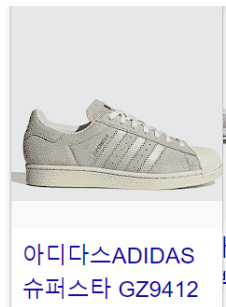
3. NMD (NMD R1, NMD R2)

NMD는 아디다스의 아이코닉한 스트리트 스니커즈로, 혁신적인 디자인과 뛰어난 착용감으로 10대와 청소년들 사이에서 큰 인기를 끌고 있습니다. 특히 NMD R1과 NMD R2는 다양한 색상과 스타일로 출시되어 개성 있는 스타일링을 완성할 수 있습니다.

4. 컨티넨탈 80 (Continental 80)

아디다스의 컨티넨탈 80은 1980년대의 레트로 감성을 느낄 수 있는 스타일로, 10대와 청소년들에게 인기가 많습니다. 여름에도 가볍게 신을 수 있으며, 다양한 색상으로 개성을 표현할 수 있습니다.

이 외에도 아디다스는 다양한 신발 종류와 스타일을 제공하고 있으므로, 여름에 어울리는 운동화를 찾기 위해 공식 온라인 스토어나 매장에서 다양한 제품들을 확인해 보시기 바랍니다.



이렇게 쇼핑몰 운영 및 판매자에게 효율적인 판매및 소싱활동을 할 수 있도록 트렌드 분석 알고리즘을 연구하고 관련 AI 를 개발하는 회사가 GOPHER 입니다.