# 세종대 인근 랜드마크 분류기


김준우
Sejong University
bababrll@naver.com

김지원 ○
Sejong University
socome@sju.ac.kr

남현호
Sejong University
skagusg@gmail.com

이동훈
Sejong University
glee1228@naver.com

이수민
Sejong University
sumin416@naver.com



## Abstract

GPS(Global Positioning System)의 발전은 우리 생활에 큰 변화를 가져왔다. 그리고 GPS 의 중요성은 앞으로도 중요시 되는데 그 중 무인이동체 분야에서는 GPS 를 절대적으로 의존하고 있다. 하지만 GPS 에 오차는 존재할 수 밖에 없고, RTK(Real Time Kinematic) GPS 등 정확도를 높인 다양한 기기들이 존재하지만 가격이 비싼 단점이 있다. 이에 컴퓨터 비전 분야에서는 딥러닝을 통해 카메라로 촬영된 풍경 이미지를 보유하고 있는 DB 와 유사도를 검색해 위치에 대한 추가적인 정보를 얻어 기존 GPS 정확도를 향상시키려는 연구들이 이어졌다. 본 프로젝트에서는 그 중 대표적인 NetVLAD 를 이번 텀프로젝트를 통해서 구현하고, 직접 촬영한 데이터셋에 적용시켜 보았다.


## Introduction

Image retrieval 은 컴퓨터 비전의 근본적인 문제이다. 쿼리 이미지가 주어지면 데이터베이스에서 비슷한 이미지를 찾을 수 있어야 한다.

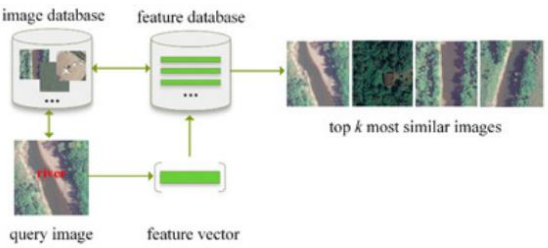


그림 1. Image retrieval

최근 딥러닝 기반 연구들은 학습을 통해 획득된 feature 를 이용하여 Image retrieval 을 수행한다. 대표적으로 NetVLAD[1]가 있다. 본 프로젝트에서도 Resnet18[2], resnet50, vgg16[3], vgg19, mobilenet[4], densenet121[5] 등과 같은 Convolutional Network 를 통해 feature 를 추출하고, NetVLAD 를 주 모델로 사용하고 end-to-end scheme 으로 성능을 높이도록 했다. 직접 촬영한 데이터셋을 사용했기 때문에 유사도가 높고 갯수가 적고 데이터들을 보완하기 위해 Argumentation 기법을 사용하여 적은 데이터셋을 대량의 데이터로 증식하여 효과적인 학습 결과를 낼 수 있었다. 다양한 Argumentation 을 통해서 1 개의 데이터를 7 개 이상의 데이터로 증식시킬 수 있었다. 본 논문에서는 세종대학교 인근 랜드마크 분류 및 검색을 쿼리 이미지를 통해서 수행하고, 쿼리 이미지와의 유사도 순으로 대표 이미지들을 출력해보고자 한다.

## 1. Related Work

본 프로젝트는 2016 년 CVPR 에서 발표된 'NetVLAD : CNN architecture for weakly supervised place recognition' 를 주 모델로 설정하였고, Loss 는 person re-identification 이나 face recognition 분야에 널리 쓰이는 triplet loss [6]를 사용하였다. 또한 Feature 를 추출하는 backbone network 로는 ResNet18, VGG16 등이 있다.

### 1.1. Model

본 프로젝트에서는 앞서 제기한 문제를 해결하기 위해서 2016 년 CVPR 에서 발표된 NetVLAD 를 사용했다. 본 모델은 과거 해당 문제를 해결하기 위해서 이미지에 대해서 SIFT(Scale-Invariant Feature Transform) 와 같은 Extract local features 를 이용해 이미지의 특징을 기술하고 이렇게 기술된 이미지들의 벡터들을 집합화 하는 Bow[7], VLAD[8] 등을 사용하였다. 그 중 VLAD 는 이미지의 특징을 기술한 벡터들을 군집화를 진행하고 군집화를 통해서 중앙값과 새로 들어온 이미지의 벡터의 차이를 계산하여 이를 최종적인 특징으로 기술하는 방법이다. 이번 프로젝트를 진행하기 위해서 사용한 모델은 앞서 설명한 이미지 분류의 방법들을 end to end 방식으로 학습이 가능하도록 설계한 NetVLAD 이다.

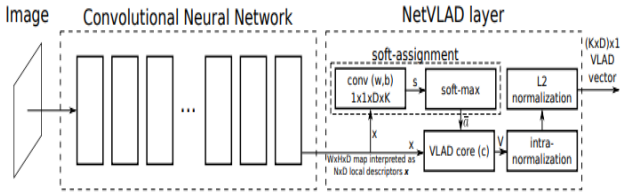


그림 2. CNN architecture with the NetVLAD layer

NetVLAD 를 앞서 설명한 방법과 비교해서 이야기하면 가장먼지 이미지에 대한 특징은 CNN 을 이용해 추출한다. 기존에 있는 다양한 CNN 을 이용할 수 있으며 본 프로젝트에서는 ResNet18 을 사용하였으며, 다양한 네트워크로 변경해 성능을 비교하였다. 이를 통해서 기술된 이미지 특징 벡터는 새롭게 만들어진 NetVLAD 네트워크를 통과한다. 이를 통해서 특징 벡터를 만들고, 이 특징벡터는 다음 수식과 같을 통해서 다시 계산한다.

$$V(j,k) = \sum_{i=1}^{N} \frac{e^{\mathbf{w}_k^T \mathbf{x}_i + b_k}}{\sum_{k'} e^{\mathbf{w}_{k'}^T \mathbf{x}_i + b_{k'}}} (x_i(j) - c_k(j))$$

수식 1. NetVLAD layer 의 수식

여기서 $\frac{e^{w_k^T x_i + b_k}}{\sum_{k'} e^{w_{k'}^T x_i + b_{k'}}}$는 soft assignment 를 위한 수식이며 실제 $x_i(j) - c_k(j)$ 부분이 기존 VLAD 와 같다.

## 1.2. Classification and Loss

Online Triplet Loss

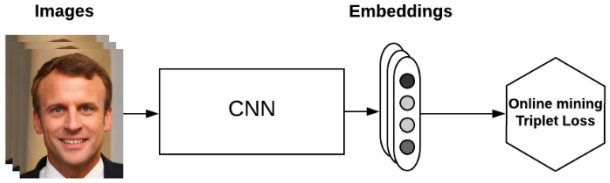


그림 3. Online Triplet Loss

본 프로젝트에서 Loss 를 계산하기 위해 Online Triplet Loss 를 사용하였다. 먼저 Triplet Loss 란 negative 는 멀리, postive 는 멀리 떨어트리는 기법이다. 수식은 다음과 같다.

$$\mathcal{L} = max(d(a,p) - d(a,n) + margin, 0)$$

postive 와의 유클리드 거리에서 negative 와의 유클리드 거리를 빼고 거기에 마진을 더한 것이 Loss 가 되는 방식으로 Loss 를 계산해 학습을 진행한다.

이와 같은 Triplet Loss 에서 Online Triplet Loss 는 기존 Triplet Loss 계산을 위해서 2 개의 positive 이미지와 1 개의 negative 이미지만 고정적으로 입력해야 한다는 점이 개선됐다. 이를 통해서 임이의 배치 사이즈만큼의 이미지가 들어와도 이중 postive 와 negative 의 개수가 각각 2 개 1 씩이 만들어진다면 자동으로 이를 찾아내 triplet loss 가 계산한다.

Classification

이번 텀프로젝트는 '분류'를 진행하는 것을 목표로 한다. 따라서 본 프로젝트에서도 1 차적으로 SVM 을 통한 분류를 진행하였다. NetVLAD 를 학습하고 이를 통해서 만들어진 특징 벡터들을 N 차원 공간에 뿌린다. 그리고 이를 우리가 처음 설정한 N 개의 클래스로 분류하기 위한 초평면을 찾아 분류를 진행하였다.

다음으로는 분류에서 나아가 쿼리 이미지에 대해서 DB 간의 유사도를 계산하고 이를 통해서 유사도가 높은 순으로 출력하는 방식의 분류도 진행하였다. 이는 SVM 을 사용한 것이 아닌, 모든 DB 와 비교해서 가장 유사도가 높은 5 개의 이미지를 출력하고 출력된 5 개의 이미지의 클래스(라벨)정보를 가지고 분류를 진행하였다

## 1.3. Additional Features

본 프로젝트의 CNN 아키텍처 중 베이스로 사용된 Resnet 은 2015 년 ILSVRC 에 출전하여 classification, localization 및 detection 분야에서 모두 우승을 차지한 아키텍처이다. 가장 큰 특징으로는 복잡도를 줄이기

위해 hidden fully connected layer, dropout 를 사용하지 않은 점과 shortcut connection 을 사용한 점이다.

## 2. Dataset

Sejong_datasets 는 저희가 개설한 github : NetVLAD-Example-on-Colab 에서 Colab 을 이용하여 Kaggle 과 Google drive 를 통해 온라인으로 제공한다. 모든 데이터셋은 공개되어 있으며 직접 따라 할 수 있다.

Sejong_datasets 은 총 1332 개의 사진으로 구성되어있고 이미지의 사이즈나 파일 형식은 일정하지 않다. 세종대 인근의 6 개의 지형물에 대한 이미지 정보를 담고 있으며, 저희 AI_Leader 에서 촬영한 사진들과 웹 또는 로드뷰를 통해 직접 얻은 datasets 이다. 따라서 환경적인 제약이나 촬영 방식이 미숙하여 이미지의 수가 다른 전문가들이 보기에 부족할 수 있다. 이러한 점을 보완하기 위해 imgaug 를 통해 다양한 날씨 다양한 색상 다양한 noise 를 추가하여 학습과정 중 데이터의 수를 증폭시키는 방법으로 보완하였다.

Test set 은 우리가 촬영한 이미지들이 아닌 웹 또는 로드뷰를 통해 얻은 자료들로만 구성하였다. 이를통해 Test set 의 신뢰성을 조금 높이고자 하였다. 또한 Test 에 Imaug 를 적용한 Test_diff 라는 폴더를 만들어 노이즈가 포함된 Test set 에서도 성능을 보이는지 확인하였다.

## 3. Training

학습에 사용한 데이터 셋은 Newtrain(656개)이고, 각각 클래스 이미지는 AI(182), Clock tower(103), Front door Child(170), Front door Sejong(98), Stone statue(103)로 구성되어 있다. 특징 벡터를 추출하는 우리의 네트워크 중 Convolutional Network 은 기존에 있는 다양한 pretrained CNN 모델 중 Resnet18, Resnet50, Vgg16, Vgg19, Mobilenet[7], Densenet121[8] 로 학습을 진행하였다.

이미지 검색의 성능평가를 위한 테스트에 사용한 데이터 셋은 Test(81개)이고, 각각 클래스 이미지는 AI(14), Clock tower(22), Front door Child(14), Front door Sejong(24), Stone statue(7)로 구성되어 있다.

## 4. Experiments and Results

테스트 데이터 셋 중 하나를 query 이미지로 이를 제외한 나머지를 reference 이미지로 구분했고, Query 의 vector 와 reference 의 vector 간의 유클리드 거리를 측정하여 query 와 가장 유사한 순으로 reference 이미지를 순서대로 나열하였다. 정보 검색 평가(Information Retrieval Evaluation) 방법 중 MAP(Mean Average Precision)[9]을 참고하였다. Test 데이터로 다른 CNN 을 활용한 네트워크를 사용한 검색 성능을 비교한 결과이다.

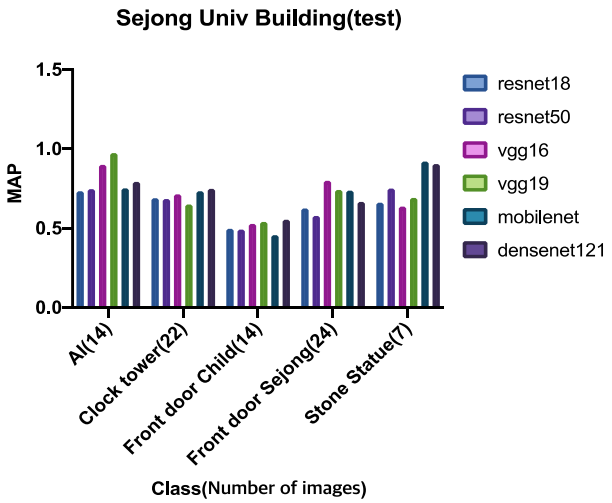


그림 4. Test 데이터를 사용한 여러가지 네트워크 성능 비교

다음은 NEW train 의 일부분으로 테스트를 진행한 결과이다

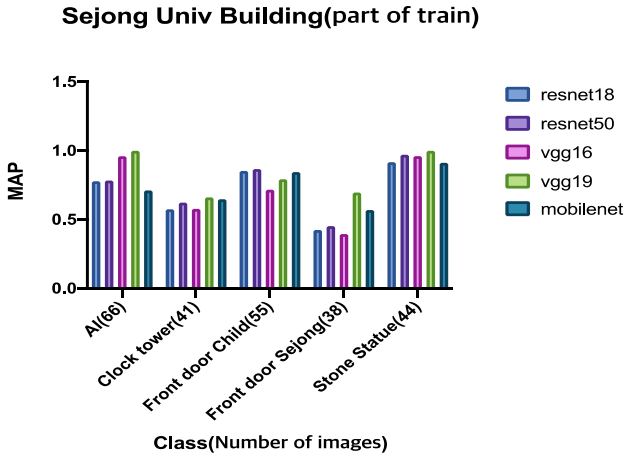


그림 5. NEW train 의 일부분으로 여러가지 네트워크 성능 비교

| Query | Rank_1 | P(5) | AP |
|---|---|---|---|
| 대양AI센터 | 1 | 0.97 | 0.83 |
| 시계탑 | 2 | 0.88 | 0.65 |
| 어린이대공원 정문 | 3 | 0.76 | 0.48 |
| 세종대학교 정문 | 2 | 0.66 | 0.63 |
| 석상 | 1 | 0.85 | 0.72 |

Rank_1 : 검색되는 첫 번째의 관련된 이미지 순위
P(5) : 검색된 관련된 5번째 이미지 이후의 precision
AP : 검색된 관련된 전체 이미지 평균의 precision

유사한 사진들이 많고 사진의 수가 적은 데이터셋을 가지고 있기 때문에 유사한 사진들을 변환하여 사진을 늘려서 학습한다면 조금 더 좋은 결과를 낼 것으로 기대했다. 가장먼저 resnet18 을 사용한 모델의 경우 클래스를 맞춘 정확도는 0.78 다. 이후 imgaug 의 함수를 사용하여 여러가지를 전처리 시도를 해보며 정확도 테스트를 하였다. 다음은 사용한 전처리 기법이다.

**시도 1** : 0.25 확률로 (-25 도~25 도) 회전,0.25 확률로 (1.0~1.2)배 확대, 0.25 확률로 잡음추가

**시도 2** : 0.25 확률로 (-25 도~25 도) 회전,0.25 확률로 (1.0~1.2)배 확대, 0.25 확률로 밝기변화(128 을 임계점으로 설정)

**시도 3** : 0.25 확률로 (-25 도~25 도) 회전,0.25 확률로 (1.0~1.2)배 확대, 0.5 확률로 구름추가

**시도 4** : 0.25 확률로 (-25 도~25 도) 회전,0.25 확률로 (1.0~1.2)배 확대, 0.25 확률로 잡음추가,0.5 확률로 밝기변화(128 을 임계점으로 설정),0.25 확률로 구름추가

**시도 5** : 0.25 확률로 (-25 도~25 도) 회전,0.25 확률로 (1.0~1.2)배 확대, 0.25 확률로 잡음추가, 0.25 확률로 색 변환, 0.25 확률로 구름추가

**시도 6** : 0.25 확률로 (-25 도~25 도) 회전,0.25 확률로 (1.0~1.2)배 확대, 0.5 확률로 흑백변환

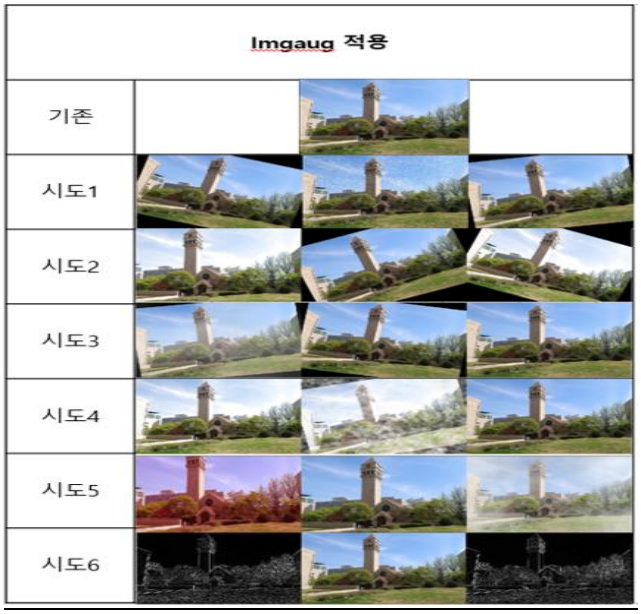


그림 6. Imgaug 를 적용한 사진 예시

표 1. 6 개의 전처리에 대한 정확도 분석

| | 기존 모델 | 시도 1 | 시도 2 | 시도 3 | 시도 4 | 시도 5 | 시도 6 |
|---|---|---|---|---|---|---|---|
| 정확도 | 0.78 | 0.68 | 0.77 | 0.79 | 0.80 | 0.83 | 0.62 |

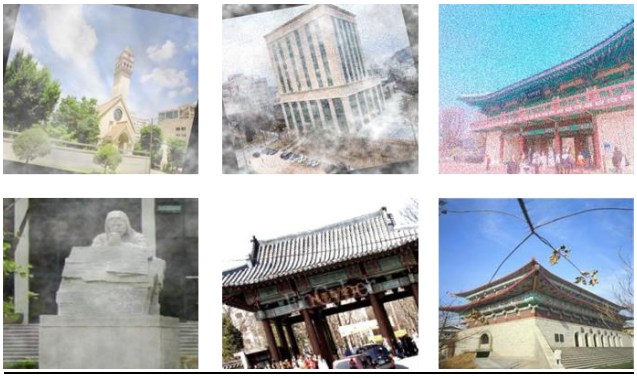

그림 7. 어려운 이미지 예시

기존의 테스트 이미지에 0.5 확률로 (-25 도~25 도) 회전, 0.5 확률로 (1.0~1.2)배 확대, 0.5 확률로 잡음추가, 0.5 확률로 밝기변화(128 을 임계점으로 설정), 0.5 확률로 좌우대칭, 0.5 확률로 구름추가를 주어 어려운 환경에서의 테스트 셋을 구축하고 평가하였다.

표 2. 6 개의 전처리에 대한 어려운 이미지 정확도 분석

| | 기존 모델 | 시도 1 | 시도 2 | 시도 3 | 시도 4 | 시도 5 | 시도 6 |
|---|---|---|---|---|---|---|---|
| 정확도 | 0.75 | 0.62 | 0.80 | 0.81 | 0.81 | 0.77 | 0.62 |

모델의 optimize 방식에 따라 정확도가 달라지기 때문에 어떤 optimize 를 사용하였을 때 가장 정확도가 우수할 것인지 찾아보기 위하여 여러가지 optimize 를 사용하여 실험해보았습니다. 같은 모델에서 optimize 만 변경해 보았을 때의 정확도는

아래와 같습니다.

표 3. Optimizer 별 성능 분석

| | SGD | Adam | AdaDelta | Adagrad | RMSprop |
|---|---|---|---|---|---|
| 정확도 | 0.76 | 0.58 | 0.48 | 0.57 | 0.52 |

## References


[1] Arandjelovic, Relja, et al. "NetVLAD: CNN architecture for weakly supervised place recognition." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016.

[2] He, Kaiming, et al. "Deep residual learning for image recognition." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016.

[3] Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." *arXiv preprint arXiv:1409.1556* (2014).

[4] Howard, Andrew G., et al. "Mobilenets: Efficient convolutional neural networks for mobile vision applications." *arXiv preprint arXiv:1704.04861* (2017).

[5] Huang, Gao, et al. "Densely connected convolutional networks." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017.

[6] https://web.stanford.edu/class/cs276/handouts/EvaluationNew-handout-1-per.pdf

[7] https://imgaug.readthedocs.io/en/latest/index.html