Introduction to Computer Vision Assignment #4

2019-23191 박상욱

본 프로젝트는 Bag of Word를 이용해서 semantic scene segmentation을 수행한다. Python 3.6을 이용해 작성하였으며, 프로그램의 효율을 위해 Numpy, Scipy, Skimage, Matplotlib, Sklearn 패키지를 이용하였다.

1. Constructing dictionary using Texton features

a) Filter banks for texton features

주어진 Image의 Feature를 뽑아내기 위한 Filter bank를 디자인하였다. Filter bank를 생성하기 위해 3개의 Gaussian Filter($\sigma = 1, 2, 4$)를 이용하였고, 4개의 LoG(Laplcian of Gaussians)(with $\sigma = 1, 2, 4, 8$)을 사용하였다. DoG(Derivative of Gaussians) Filter는 $\sigma = 2,4$를 이용해 scale=3, orientation은 6방향으로 하였다. Filter의 Support는 49로 생성하였다. 최종적으로 생성한 filter는 총 19개로 Fig.1이 사용한 Filter bank이다.

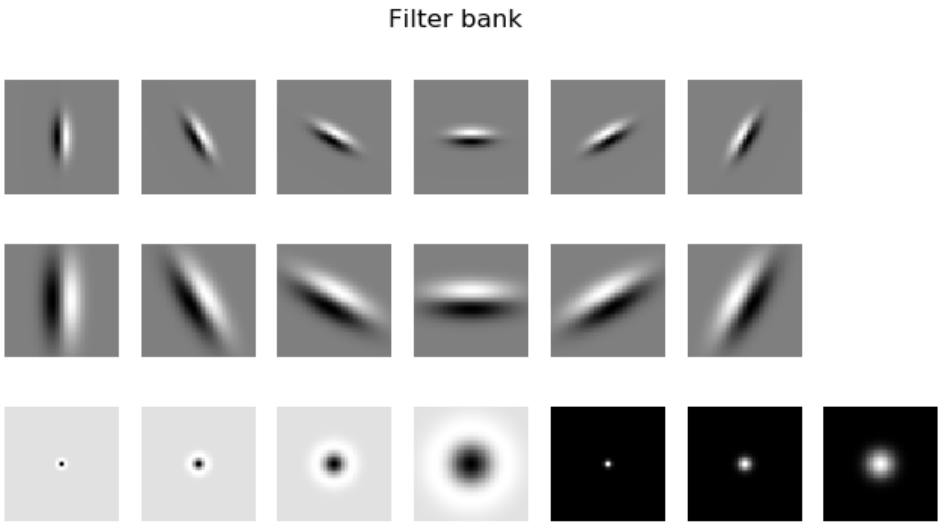


Fig. 1 Designed Filter bank

b) Texton features

각 이미지에 대해 a)에서 구한 Filter Bank의 각 Filter를 Convolution하여 이미지의 pixel마다의 feature를 생성하였다. Feature의 dimension은 Gaussian Filter를 L, a, b 채널에 적용하고, LoG와 DoG 필터는 L 채널에만 적용하여 총 25(9+4+12)가 된다. 각 이미지에 대한 feature는 추출하여 *data/features* 폴더에 저장하였다.

c) Constructing Dictionary

Filter Bank를 이용하여 추출한 Train image 모두의 Feature에 대하여 Kmeans 알고리즘을 적용해 centroid를 찾는다. 이를 위해 Python의 Sklearn 패키지의 Kmeans 모듈을 사용하였다. K=100을 사용해 총 100개의 centroid를 train image features로부터 얻어 '*k_means_cluster*'라는 이름의 파일로 저장한다.

2. Superpixel representation

각 이미지를 Superpixel로 segmentation 하여 superpixel마다 class를 적용한다. SLIC 알고리즘을 이용해 이미지를 Superpixel로 나타내기 위해 Skimage 모듈을 사용하였다. 이미지를 Superpixel로 나타내면 아래와 같은 결과를 얻을 수 있다.

Fig 2. Examples of superpixel representation

3. BoW Representation

이미지의 각 Superpixel에서 pixel마다의 feature를 앞서 학습한 K-means visual word cluster에서 nearest centroid를 찾고, K size의 histogram을 생성하였다. 본 프로젝트에서는 superpixel의 histogram들을 이용하여 void, building, grass, tree, cow, horse, sheep, sky, mountain, aeroplane, face, car, bicycle의 총 13개 카테고리에 대해 class histogram을 생성하였다. 학습은 Training image의 Ground Truth(*data/mask*에 위치)를 이용하였다. MSRC Dataset의 Ground Truth에 대한 정보는 아래와 같다.

object class	R	G	B	Colour
void	0	0	0	
building	128	0	0	
grass	0	128	0	
tree	128	128	0	
cow	0	0	128	
horse	128	0	128	
sheep	0	128	128	
sky	128	128	128	
mountain	64	0	0	
aeroplane	192	0	0	
water	64	128	0	
face	192	128	0	
car	64	0	128	
bicycle	192	0	128	

Fig 3. Color information of Ground Truth

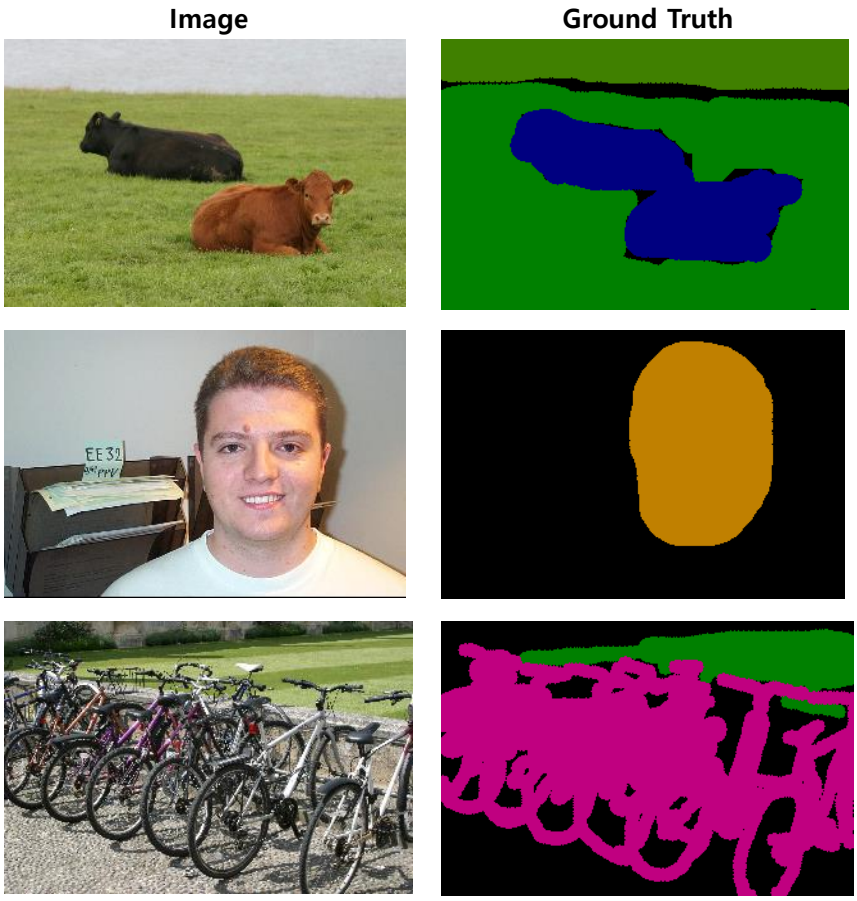
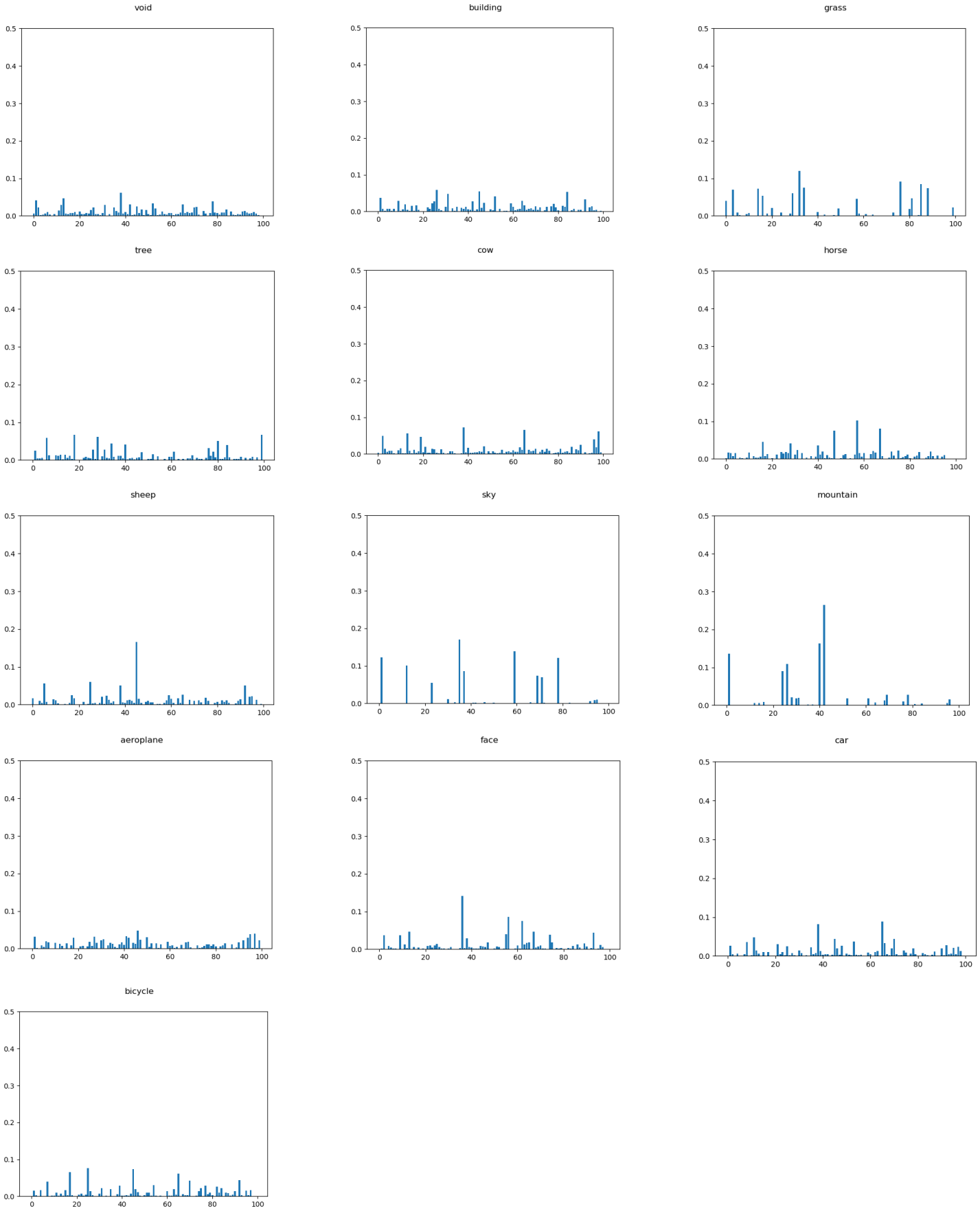


Fig 4. Image and corresponding Ground Truth

130장의 Training Image를 통해 얻은 BoW의 13개 class의 Histogram은 아래와 같다. 각 class에 대한 Histogram은 *res/voca* 폴더에서 확인할 수 있다.

Histograms of 13 classes



4. MRF Formulation

Image의 Markov random field energy 계산을 위해 아래의 식을 사용하였다.

$$E(l) = \sum_{p} f(l_p) + \sum_{(p,q)\in N} \theta_{pq}(l_p, l_q), \quad (1)$$

$f(l_p)$는 superpixel p의 histogram과 BoW Vocabulary의 l_p Class의 historgram 간의 chi-square distance로 정의된다. Pairwise potentials는 Potts model을 사용하며 λ가 상수일 때

$$\theta_{pq}(l_p, l_q) = \begin{cases} 0 \; if \; l_p = l_q \\ \lambda \; if \; l_p \neq l_q \end{cases}$$

를 이용하여 Energy를 계산하였다.

Pairwise potential 계산을 위하여 Superpixel의 Neighbor 관계를 알아야하므로, Superpixel간의 관계를 scipy 모듈의 Delaunay 모듈을 이용하여 triangle로 대략적인 neighbor 그래프를 그렸다. 이를 통해 Image의 Energy를 계산할 수 있다.

5. Inference

4 에서 구한 Energy를 Minimize하기 위해 Alpha expansion을 활용할 수 있다. Alpha expansion은 Graph cut 알고리즘을 통해 Binary MRF를 최적화한다. 이를 이용해 총 13개의 class에 대해 클래스별 MRF Energy를 최적화하는 Binary Graph cut을 수행할 수 있다. 이 과정을 Energy 개선이 없을 때까지 진행하여 Optimization을 수행한다. 이 과정을 통해 Inference에 있어 잘못 유추된 노이즈 Segmentation이 사라질 것으로 기대했으나, 코드를 제대로 적용하지 못하여 결과를 얻진 못하였다.

6. Results

data/test 이미지에 대한 모든 Inference 결과는 *res/inference* 에서 확인할 수 있다.

Train 데이터에 대한 Inference의 일부는 Fig 5.에 Test 데이터에 대한 Inference는 Fig 6.에 나타내었다.

Fig 5. Inference examples for train images

Fig 6. Infernce examples for test images

표 1. Confusion matrix

Inference / G. T.	void	building	grass	tree	cow	horse	sheep	sky	mountain	aeroplane	face	car	bicycle
void	0.062	27.763	12.204	35.947	19.462	0	0	**60.259**	0	33.135	31.46	29.783	52.137
building	0.012	12.531	0.175	2.59	3.145	0	0	18.36	0	29.863	6.999	8.99	**40.385**
grass	0.01	0.208	**77.928**	11.52	0.376	0	0	0.118	0	0.209	0.783.	0.035	0.134
tree	0	2.515	15.962	**39.664**	3.13	0	0	7.9	0	12.121	0.114	2.498	9.91
cow	0.004	1.581	3.679	1.992	**36.968**	0	0	1.518	0	2.734	9.015	8.317	5.396
horse	0	0	0	0	0	0	0	0	0	0	0	0	0
sheep	0	0	0	0	0	0	0	0	0	0	0	0	0
sky	0	0.188	0	0.004	0	0	0	**88.684**	0	1.435	0	1.912	0.633
mountain	0	0	0	0	0	0	0	0	0	0	0	0	0
aeroplane	0	1.502	0.683	2.712	1.454	0	0	5.285	0	**15.274**	1.472	3.019	10.647
face	0	0.536	0.114	0.403	4.14	0	0	0.945	0	1.141	**59.541**	6.02	3.792
car	0.011	1.754	0.118	0.497	6.685	0	0	16.745	0	12.634	6.946	**33.172**	22.164
bicycle	0.005	4.364	0.336	1.682	1.072	0	0	7.076	0	5.232	1.055	2.942	**58.247**

Fig. 1은 110개 Test image에 대해 semantic scene segmentation을 수행하고 얻은 Confusion matrix이다. Horse, sheep, mountain은 Test image에서 Ground Truth Class가 하나도 없어 값이 나오지 않았다. Void와 같은 Class의 경우 Train과정에서 다양한 텍스쳐가 모두 영향을 주기 때문에 제대로 추론하지 못하는 것을 확인할 수 있다. 전반적으로 grass, sky 같이 확실한 텍스쳐가 있고, 단순한 이미지의 경우에 Inference 확률이 높은 것을 확인할 수 있다.