







# Homework 1



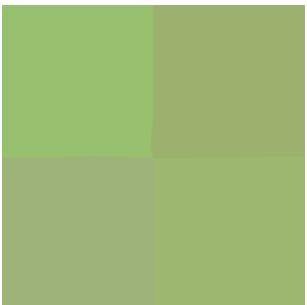



## Deep Learning for Computer Vision

學生:葉政樑 學號:R08945006

Problem1-1: K-means






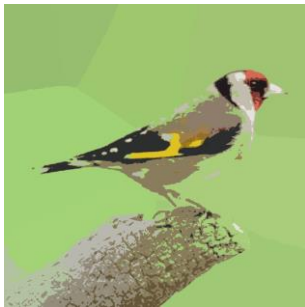
Original Image	K=2	K=4
		
K=8	K=16	K=32
		

Problem1-2: add (x,y) as feature

Original Image	K=2	K=4
		
K=8	K=16	K=32
		



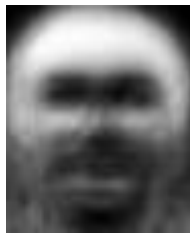


### Problem1-3: 比較與改進

比較 1-1、1-2 的結果可以看到在 1-2 單純加入影像(x,y) location 資訊當作 feature 的效果並不好，原因並不是 feature 沒用。由觀察 feature 得知前三維度 feature: RGB 值分布在 0~255，而(x,y) location 則是分布在 0~1023 間，因此 feature 間尺度差異過大造成 1-2 的結果明顯被位置(x,y)特徵影響分成特定區塊。因此我採用資料正規化策略，將 features 的範圍都壓縮成 0~1 之間，在進行 K-means 分類，結果如下：







Original Image	K=2	K=4
		
K=8	K=16	K=32
		

由以上結果可以發現資料正規化的重要性，將資料合理的縮放，讓 features 間尺度不差過大才能有較好的成果(與 1-2 相比結果進步許多)。

### Problem2-1: PCA

Mean Face	1 <sup>st</sup> eigenface	2 <sup>nd</sup> eigenface	3 <sup>rd</sup> eigenface	4 <sup>th</sup> eigenface
				

Problem2-2、2-3: Reconstructed images and MSE

Original Image	N=3、MSE= 93.4996	N=50、MSE= 66.0264
		
N=170、MSE= 36.0276	N=240、MSE= 13.5641	N=345、MSE= 0.5780
		

Problem2-4: Choose hyperparameter and Cross Validation


params	split0 validation score	split1 validation score	split2 validation score	Mean validation score
{'myKNN__n_neighbors': 1, 'myPCA__n_components': 3}	0.683333	0.633333	0.608333	0.641667
{'myKNN__n_neighbors': 1, 'myPCA__n_components': 50}	0.95	0.941667	0.966667	0.952778
{'myKNN__n_neighbors': 1, 'myPCA__n_components': 170}	0.958333	0.958333	0.95	0.955556
{'myKNN__n_neighbors': 3, 'myPCA__n_components': 3}	0.608333	0.608333	0.608333	0.608333
{'myKNN__n_neighbors': 3, 'myPCA__n_components': 50}	0.875	0.925	0.883333	0.894444
{'myKNN__n_neighbors': 3, 'myPCA__n_components': 170}	0.875	0.908333	0.9	0.894444
{'myKNN__n_neighbors': 5, 'myPCA__n_components': 3}	0.6	0.608333	0.491667	0.566667
{'myKNN__n_neighbors': 5, 'myPCA__n_components': 50}	0.816667	0.841667	0.858333	0.838889
{'myKNN__n_neighbors': 5, 'myPCA__n_components': 170}	0.808333	0.833333	0.841667	0.827778

Hyperparameter 選擇在 cross validation 中平均分數(accuracy)最高的，k=1、n=170。

Problem2-5: Recognition rate of the testing set

Accuracy on testing set = 0.95 (95%)

Problem3-1: 2D Gaussian filter

Original Image	Gaussian Filter
	

經過高斯濾波之後可以發現圖片變得稍微模糊一點。



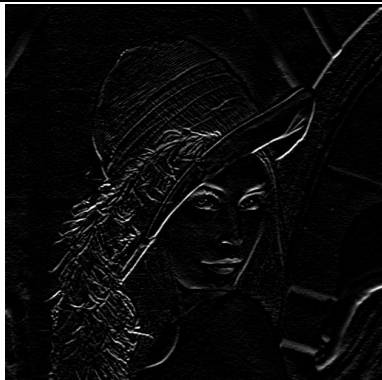
Problem3-2: first-order methods.

$$I_x(x,y) = \frac{\partial I}{\partial x} \approx \frac{1}{2} (I(x+1,y) - I(x-1,y)) \quad I_x = k_x * I$$





$$I_y(x,y) = \frac{\partial I}{\partial y} \approx \frac{1}{2} (I(x,y+1) - I(x,y-1)) \quad I_y = k_y * I$$

$$k_x \Rightarrow \text{mask} = \begin{bmatrix} +\frac{1}{2} & 0 & -\frac{1}{2} \\ +\frac{1}{2} & 0 & -\frac{1}{2} \\ +\frac{1}{2} & 0 & -\frac{1}{2} \end{bmatrix}$$

$$k_y \Rightarrow \text{mask} = \begin{bmatrix} +\frac{1}{2} & +\frac{1}{2} & +\frac{1}{2} \\ 0 & 0 & 0 \\ -\frac{1}{2} & -\frac{1}{2} & -\frac{1}{2} \end{bmatrix}$$

Original Image	$I_x$	$I_y$
		

Problem3-3: Gradient Magnitude image

Original Image	Use Original Image
	
Gaussian-filtered image	Use Gaussian-filtered image
	

由 3-3 結果來看：經過高斯濾波後的影像雖然以肉眼來看只能發現很些微差異，但是從梯度強度圖來看可以明顯看到梯度邊緣變得較模糊，符合高斯濾波模糊化的原理。

參考資料:

1. Computer Vision 圖像標準化  
<https://blog.csdn.net/xylin1012/article/details/81217988>
2. Sklearn PCA  
<https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html>
3. Eigenfaces  
[https://scikit-learn.org/stable/auto\\_examples/applications/plot\\_face\\_recognition.html#sphx-glr-auto-examples-applications-plot-face-recognition-py](https://scikit-learn.org/stable/auto_examples/applications/plot_face_recognition.html#sphx-glr-auto-examples-applications-plot-face-recognition-py)
4. Sklearn preprocessing  
[https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.minmax\\_scale.html](https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.minmax_scale.html)
5. 世上最生動的 PCA：直觀理解並應用主成分分析  
<https://leemeng.tw/essence-of-principal-component-analysis.html>
6. Reconstruction image(PCA)  
<https://www.youtube.com/watch?v=1iWAKme88II>
7. Sklearn KNN  
<https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html>
8. GridSearch prob  
<https://stackoverflow.com/questions/46618669/i-am-using-gridsearchcv-and-fit-is-giving-me-a-typeerror-get-params-missing-1/46619645>