

Deep Learning for Computer Vision

Homework 2

R06922072 黃宇謙

Problem 1: Kernel Trick

$$\mathbf{x} = [x_1 \ x_2]^T$$

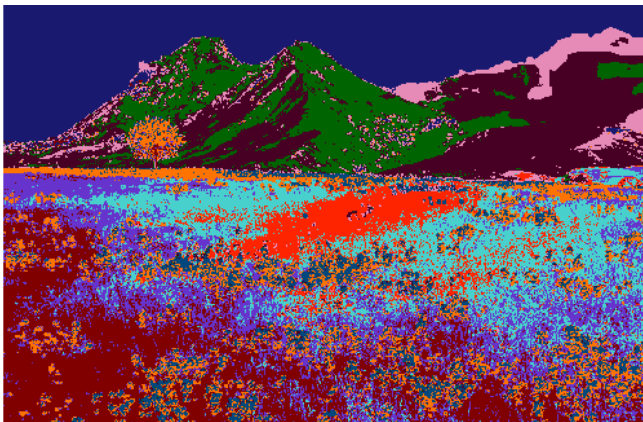

$$K(\mathbf{x}, \mathbf{x}') = (\mathbf{x}^T \mathbf{x}')^2 = \left([x_1, x_2] \begin{bmatrix} x'_1 \\ x'_2 \end{bmatrix} \right)^2 = (x_1 x'_1 + x_2 x'_2)^2$$

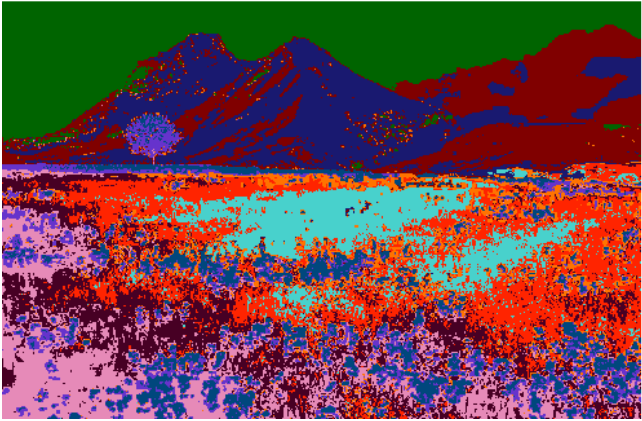

$$= (x_1 x'_1)^2 + 2x_1 x'_1 x_2 x'_2 + (x_2 x'_2)^2 = [x_1^2, \sqrt{2}x_1 x_2, x_2^2] \begin{bmatrix} x'^2_1 \\ \sqrt{2}x'_1 x'_2 \\ x'^2_2 \end{bmatrix}$$
$$= \Phi(\mathbf{x})^T \Phi(\mathbf{x}')$$

$$\text{得證：} \Phi(\mathbf{x}) = \Phi\left(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}\right) = \begin{bmatrix} x_1^2 \\ \sqrt{2}x_1 x_2 \\ x_2^2 \end{bmatrix}, \quad \Phi(\mathbf{x}) \in \mathbb{R}^3$$

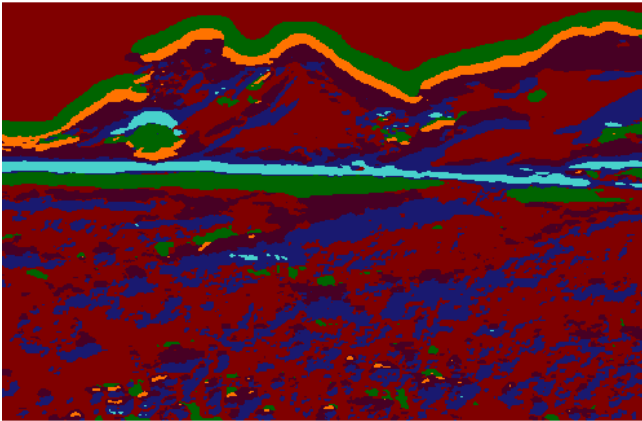
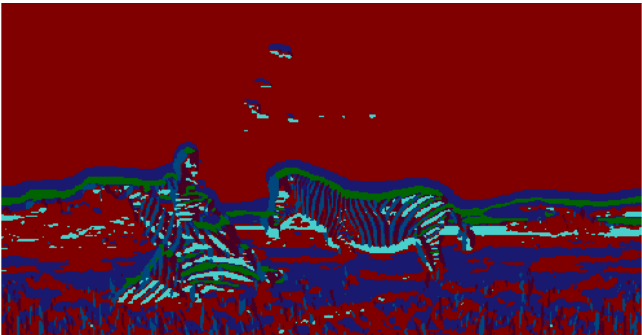
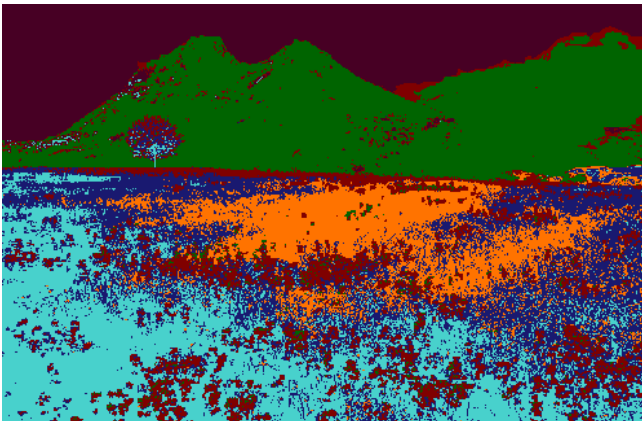

Problem 2: Color and Texture Segmentation

(a) Color Segmentation

RGB color space	
Mountain	Zebra
	

Lab color space	
Mountain	Zebra
	

(b) Texture Segmentation

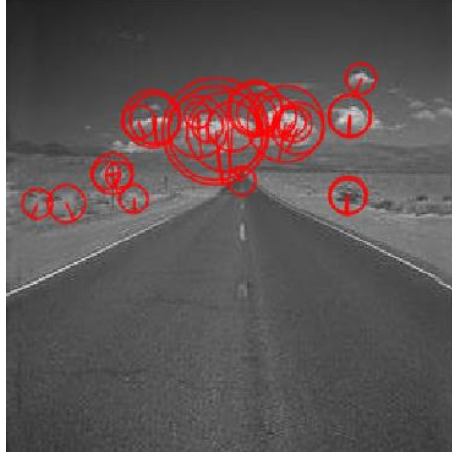
Texture	
Mountain	Zebra
	
Combine color and texture	
Mountain	Zebra
	

在 Texture segmentation，我運用 scipy 中的 convolve2d 來對原圖做 filter 的 convolution。

Problem 3: Recognition with Bag of Visual Words

(a)

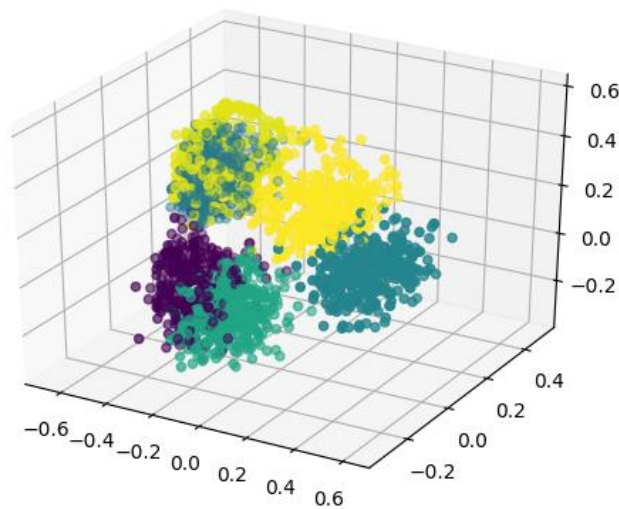
這題主要運用 SURF 來找尋圖片的 interest point 並計算出他們的 descriptors，因此我使用 opencv 中的 `cv2.xfeatures2d.SURF_create()`，在這個 code 中可以設定 threshold，網路上建議為 300~500，因此在這次的 homework 中，我將 threshold 都設定為 300。在藉由 `detectAndCompute()` 便可以計算出其 interest point 和 descriptors，如第一小題要求，下面畫出前 30 個 interest point。



(b)

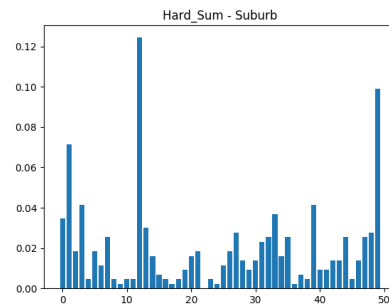
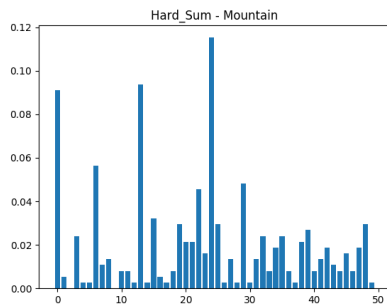
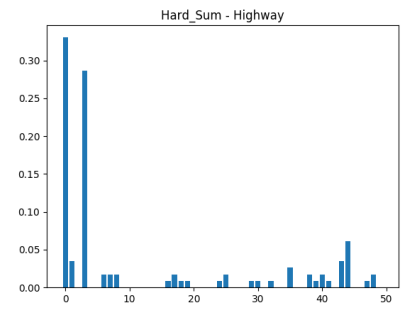
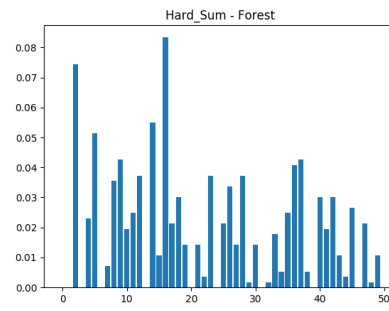
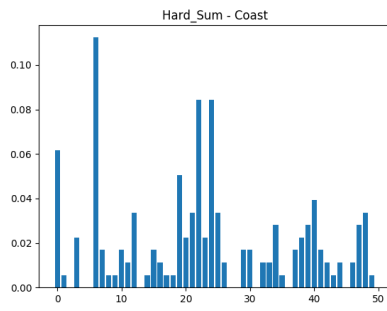
第二小題希望我們能將 50 張圖片中得到的所有 interest point 用 Kmeans 先做分群，在選其中的六群做 pca 降維成三維，畫在三維的空間裡，以觀看分類結果。

因此同上題我們先用 SURF 將 50 張圖片中的 interest point 和其 descriptors 都先找出來，在將其統整起來，以各自的 descriptor 當作依據做 Kmeans 的分群行為，在這裡我使用 sklearn 的 Kmeans 將 `n_clusters` 設定為 50。執行完 Kmeans 後，每個 interest point 得到一個 label(代表所在的群的代號)。接著隨機從裡面挑 6 群，將這六群所有的 interest point 都取出來，藉由 pca 降至三維，在這一樣使用 sklearn 中的 pca 實作，接著把得到降維結果畫出來並根據不同的 cluster 上不同的顏色。

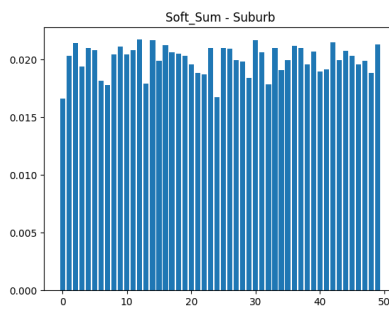
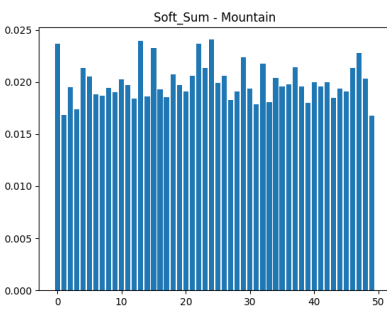
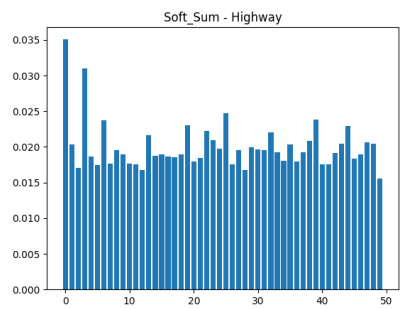
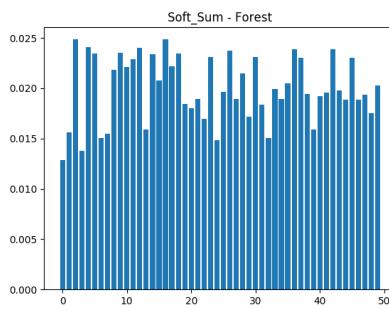
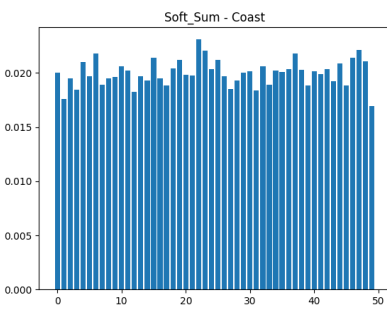


(c)

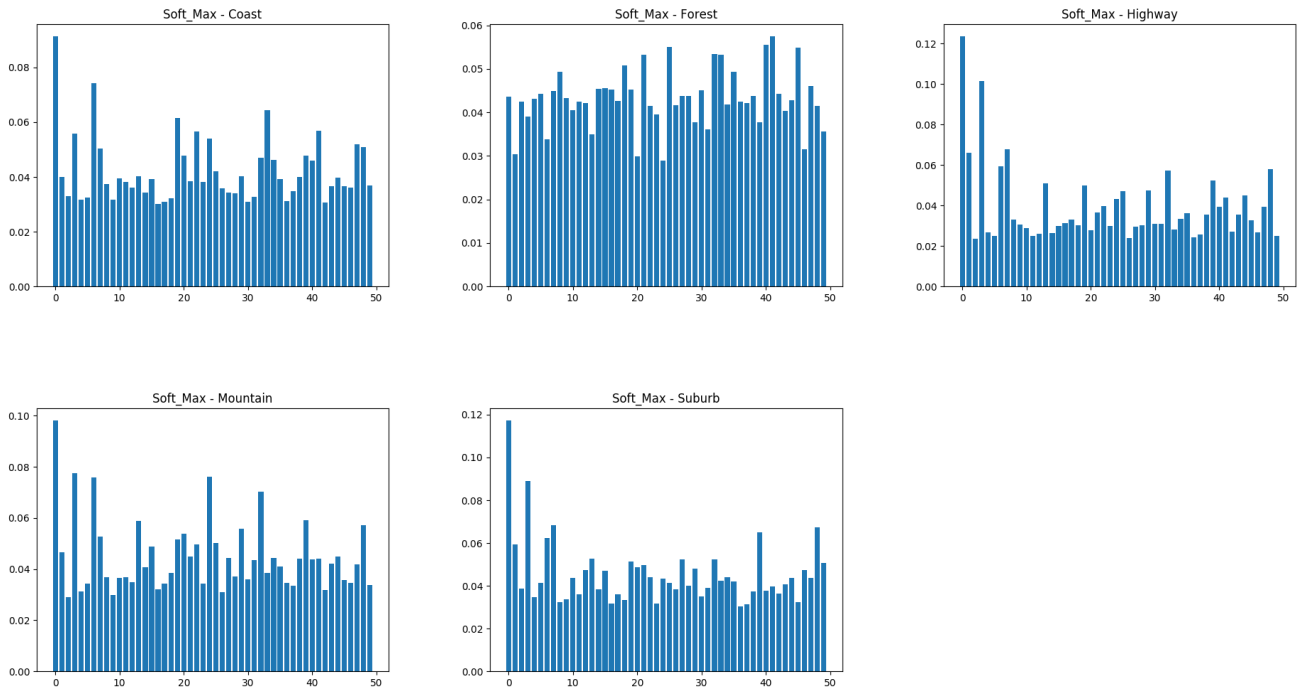
Hard_Sum



Soft_Sum



Soft_Max



由上述結果可以看出在 `hard_sum` 以及 `soft_max` 方法下做出的 Bag of Words，圖片間比較有明顯的差異，藉由這些比較大的差異，可能可以使得最後的 KNN 能找出比較分類其為不同類別的法則，故推測 `hard_sum` 以及 `soft_max` 做出的 Bag of Words 在 KNN 會比 `soft_sum` 做出來的 Bag of Words 有更好的表現。

(d)

Kmeans 分群為 50，`max_iteration` 為 5000，KNN 的分類為 5

	Hard_Sum	Soft_Sum	Soft_Max
Train-10	53.4%	50.2%	54.8%
Train-100	66%	67%	68.8%

上圖分別是用 Train-10 dataset 和 Train-100 dataset 做出來的結果。在 Train-10 的部分中，可以看到 `hard_sum` 以及 `soft_max` 的表現比 `soft_sum` 好，符合上一題所推測的。在擴大 dataset 之後(用 Train-100)可以發現各種取 Bag of Words 方法的表現皆有顯著得上升。

該題提到或許可以改變 Bag of Words 維度。由於 Train-100 有更多的 data，因此在下面我藉由改變 Kmeans 的分群數(增加 Bag of Words 的 Word 數量)，來做準確率的討論。

	C=50, Iter=5000, K=5	C=100, Iter=20000, K=5	C=150, Iter=20000, K=5
Train-100	68.8%	71.6%	73.8%

圖中的 C 為 Kmeans 的分群數，Iter 為 Kmeans 的 `max_iteration`，K 為 KNN 的分類數。

由上圖可以看到藉由增加 Bag of Words 的 Word 數量，可以幫助更深入的解析圖片，也有助於 KNN 的分類結果。