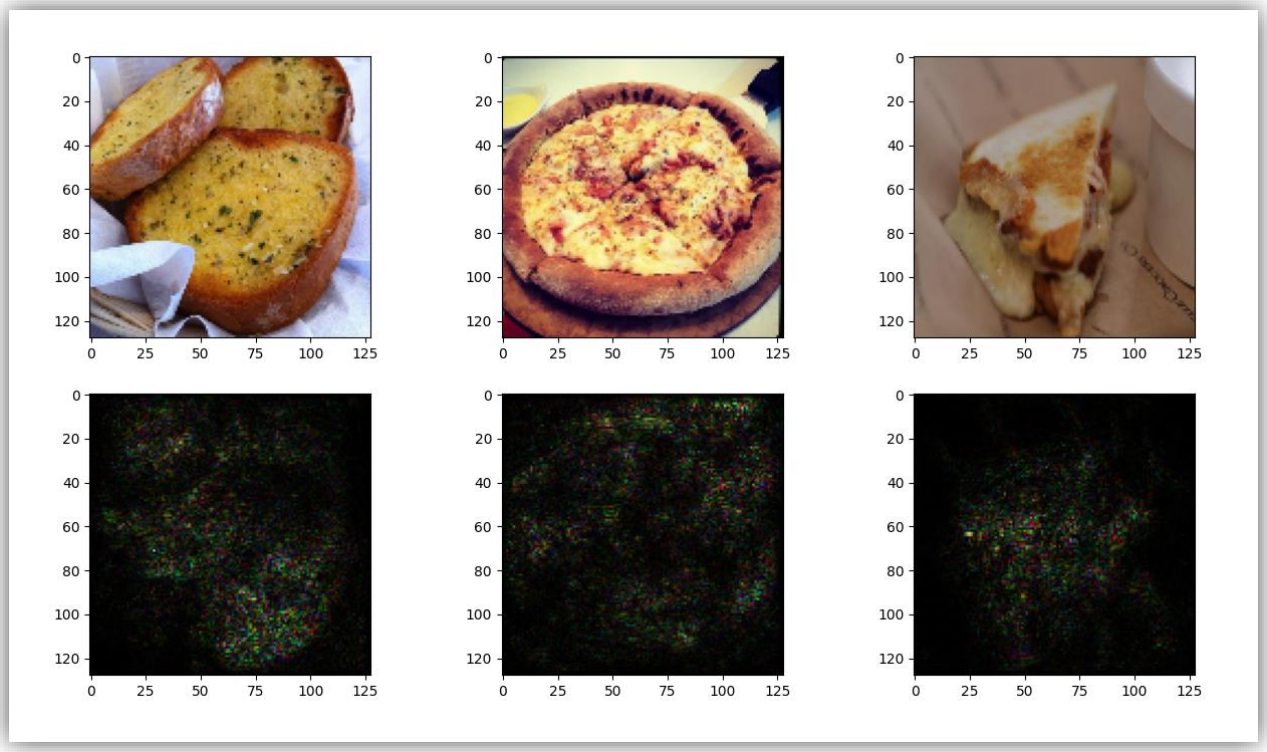
學號：B06901053　系級：電機三　　姓名： 謝承延

1. (2%) 從作業三可以發現，使用 CNN 的確有些好處，試繪出其 saliency maps，觀察模型在做 classification 時，是 focus 在圖片的哪些部份？
(Collaborators: X)

答：　我每次取3張圖片當作結果(有些class因結果較明顯有重複)。
分別為Saliency_Map1.png ~Saliency_Map11.png

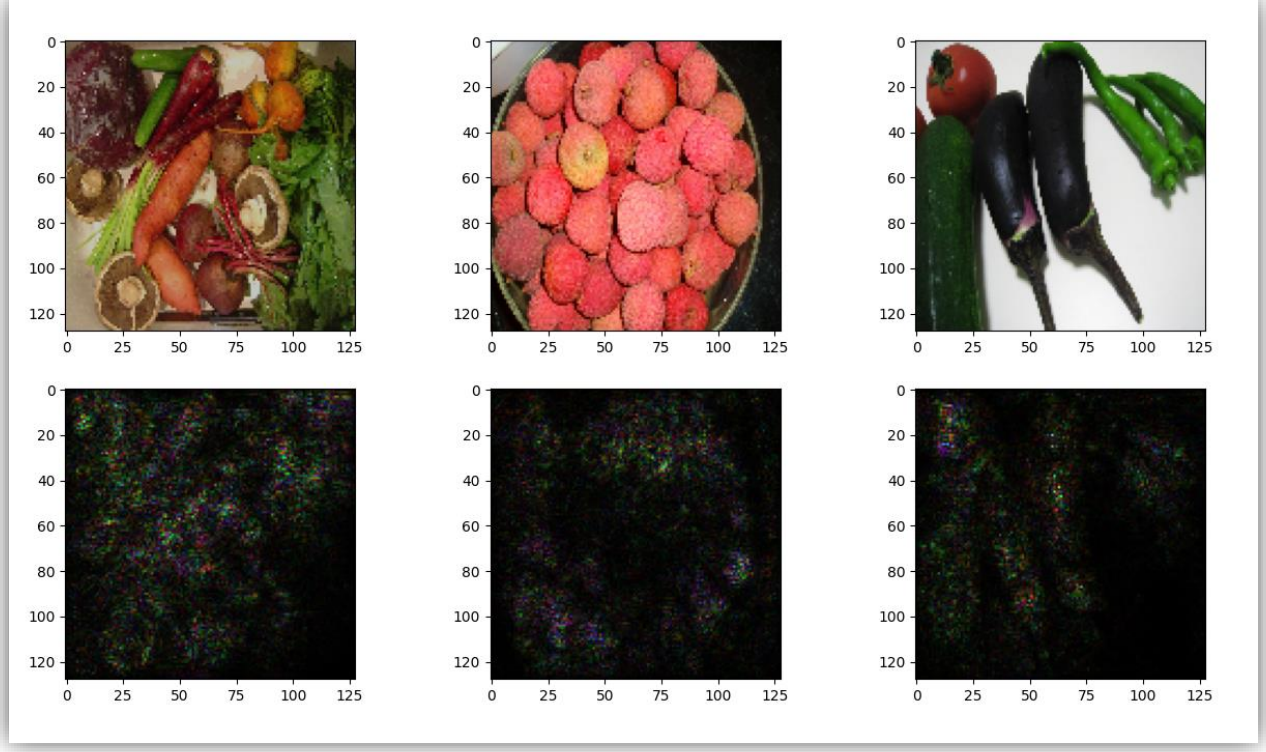
Class 1:

Bread 的圖片可以看到亮點大部分都是在食物上面，三張圖中有兩張亮點集中在食物內部，第二張圖片的亮點比較分散，大部分是在外框。



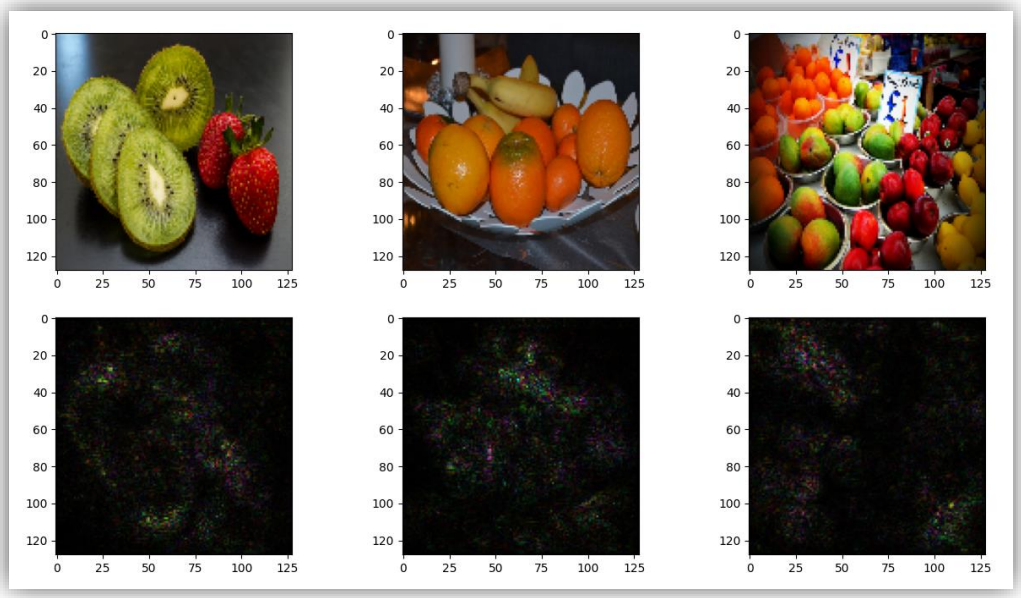
Class 2 :

第一張照片的亮點主要集中在食物密集處，第二章則是出現在碗的邊框，第三張的亮點則是呈現條狀，顯然是落在了圖片中條狀食物的地方。



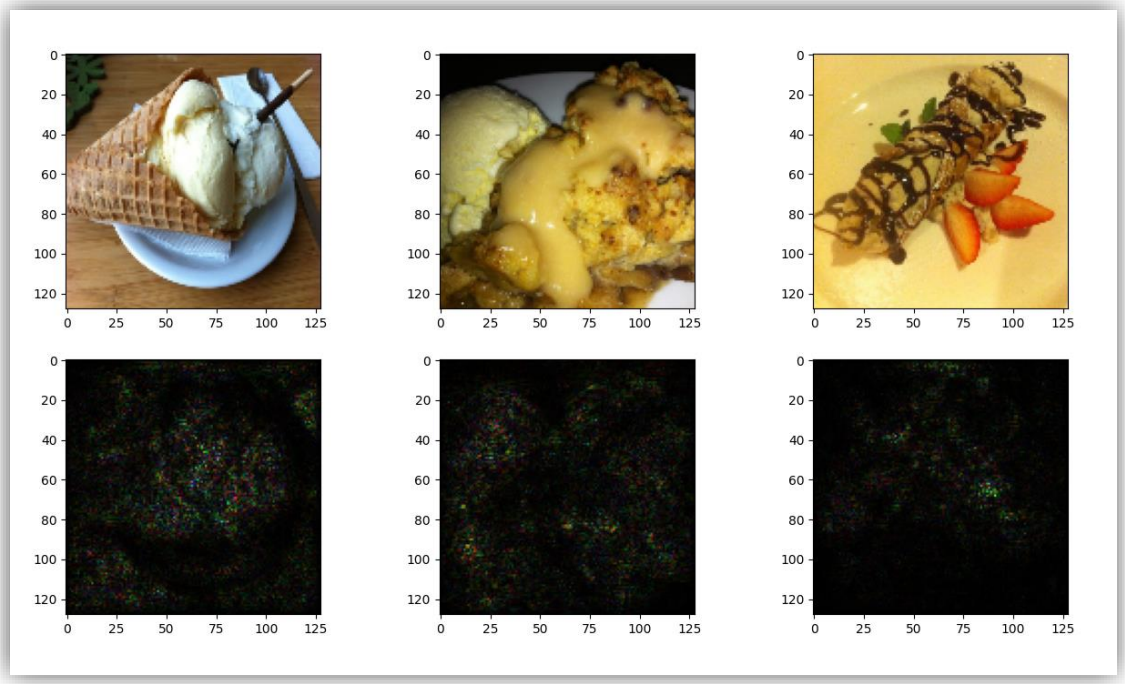
Class 3 :

這 3 張圖片的亮點比較不明顯，但大部分是集中在外框以及圖片中條狀構造(香蕉)。



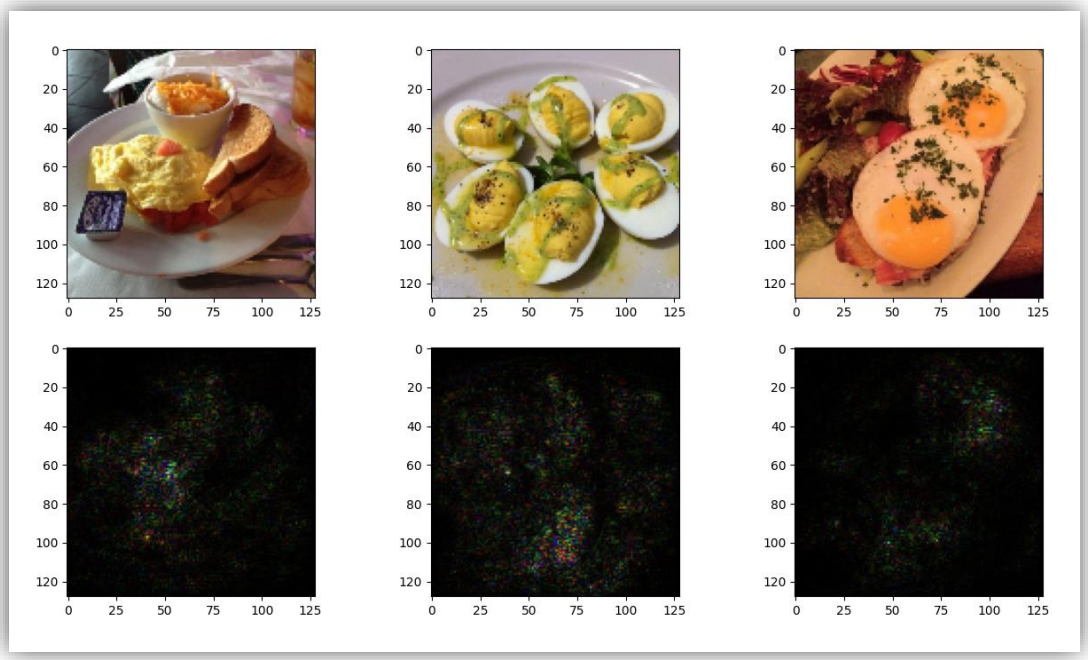
Class 4:

亮點集中在冰淇淋的球體部分，第三張照片則落在草莓。



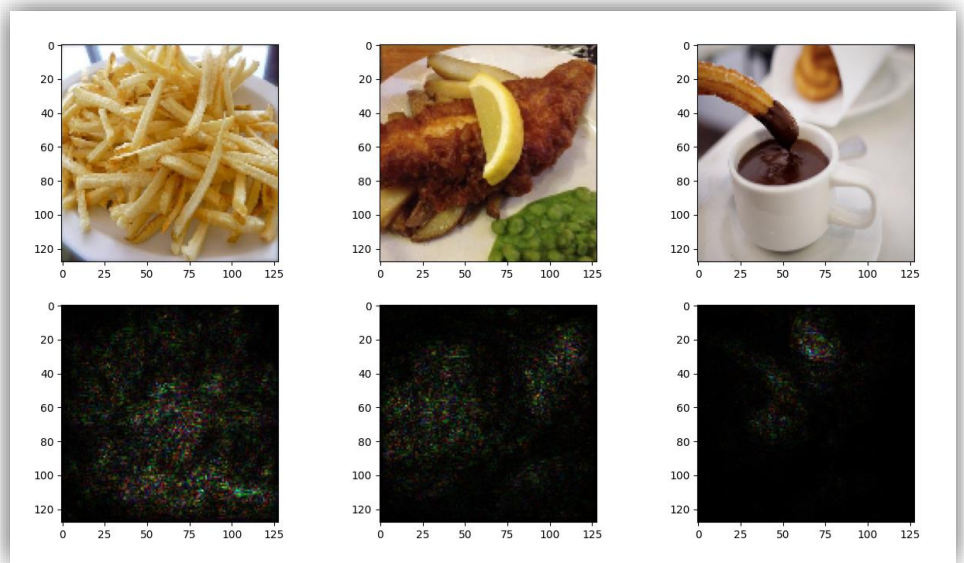
Class 5:

亮點集中在食物中較為白色的部分，第三張圖片則不太明顯



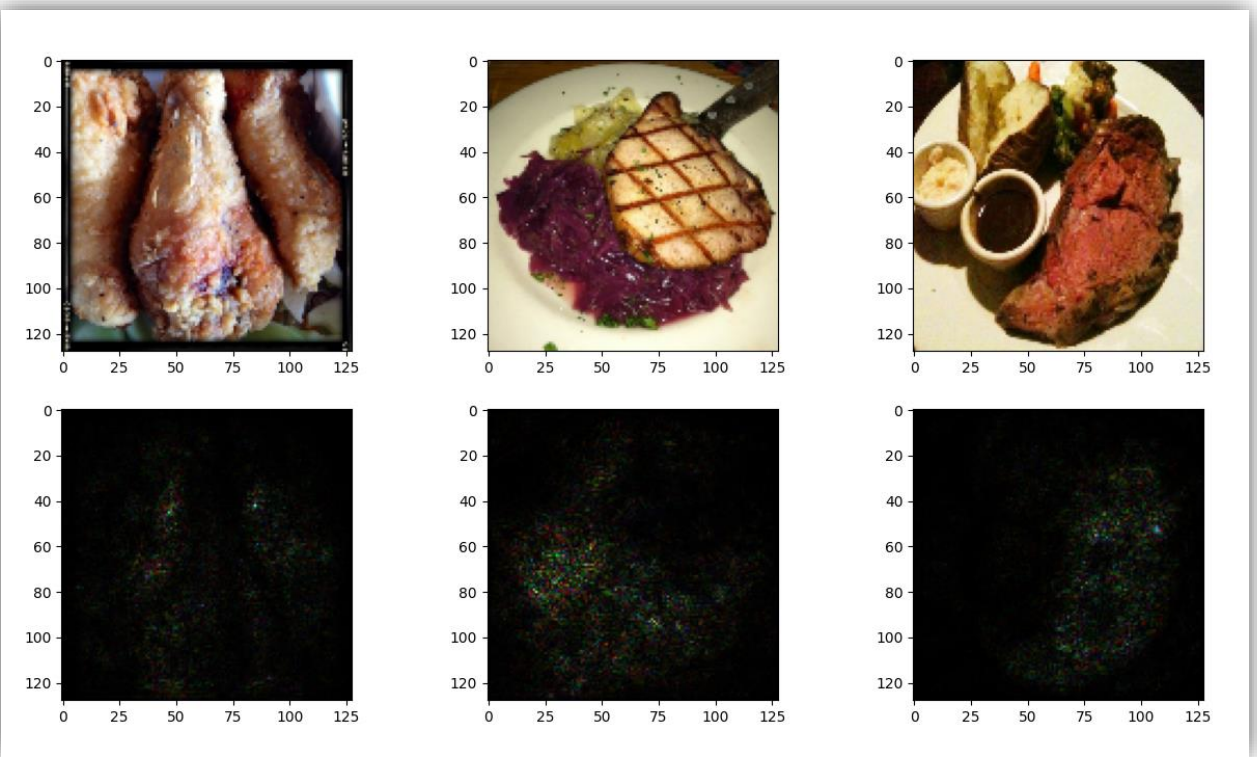
Class6:

這三張照片的亮點明顯多了，集中在食物的整個實體，第三張照片甚至亮點出現在較遠處不太明顯的食物上面。



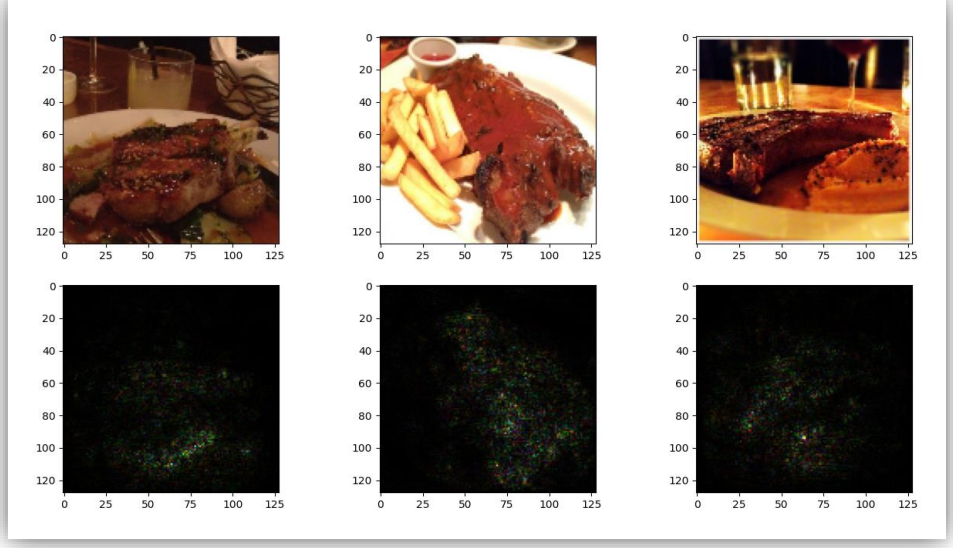
Class7:

第一張照片不太明顯，第二張第三張照片則落在醬料、肉身上。



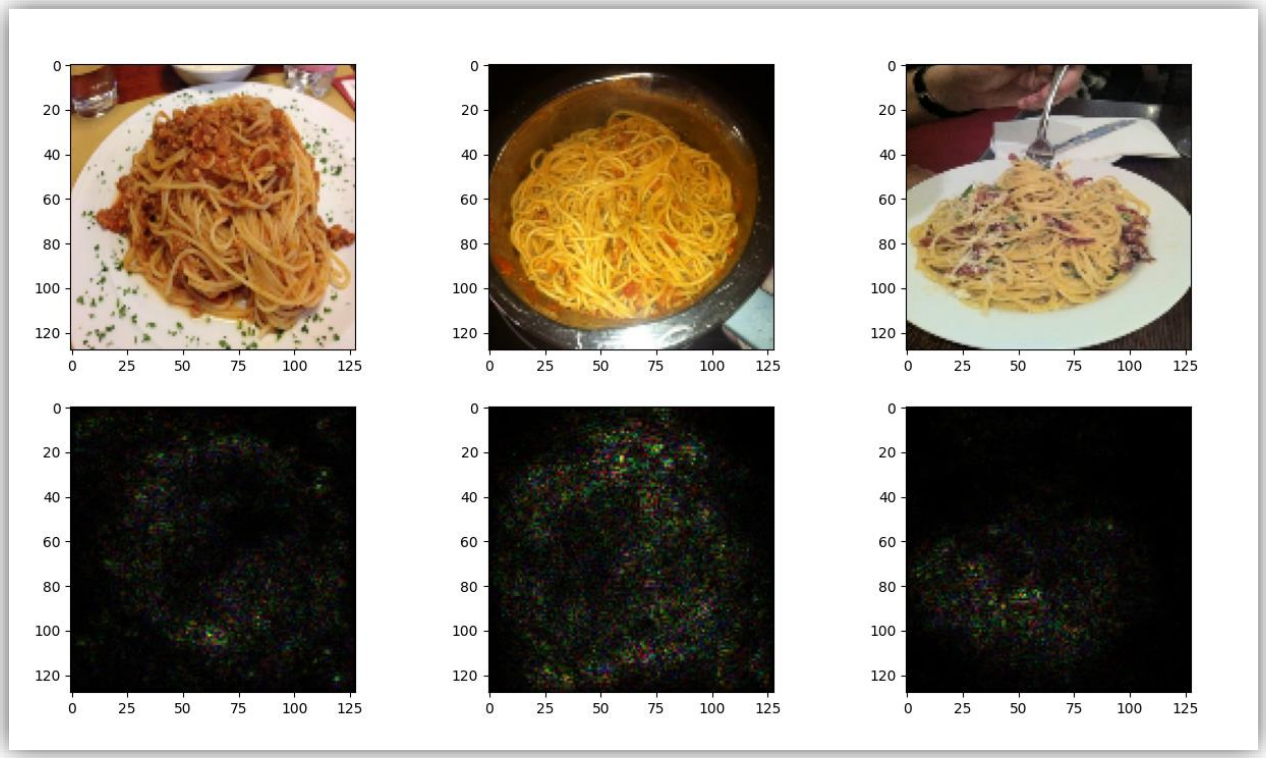
Class8:

亮點出現在肉的形狀上。



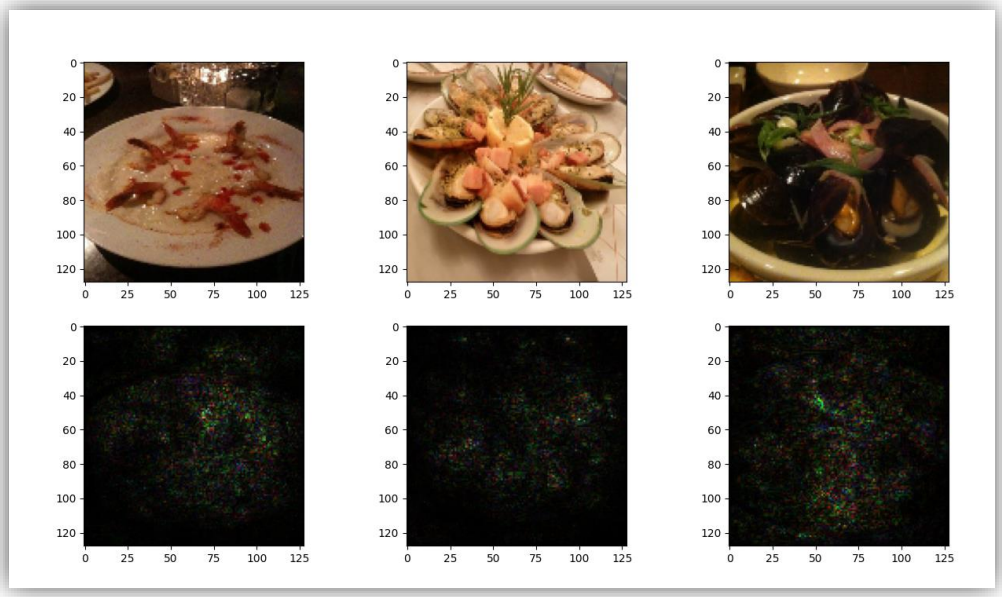
Class9:

亮點較分散，但大部分落在有麵條的地方。



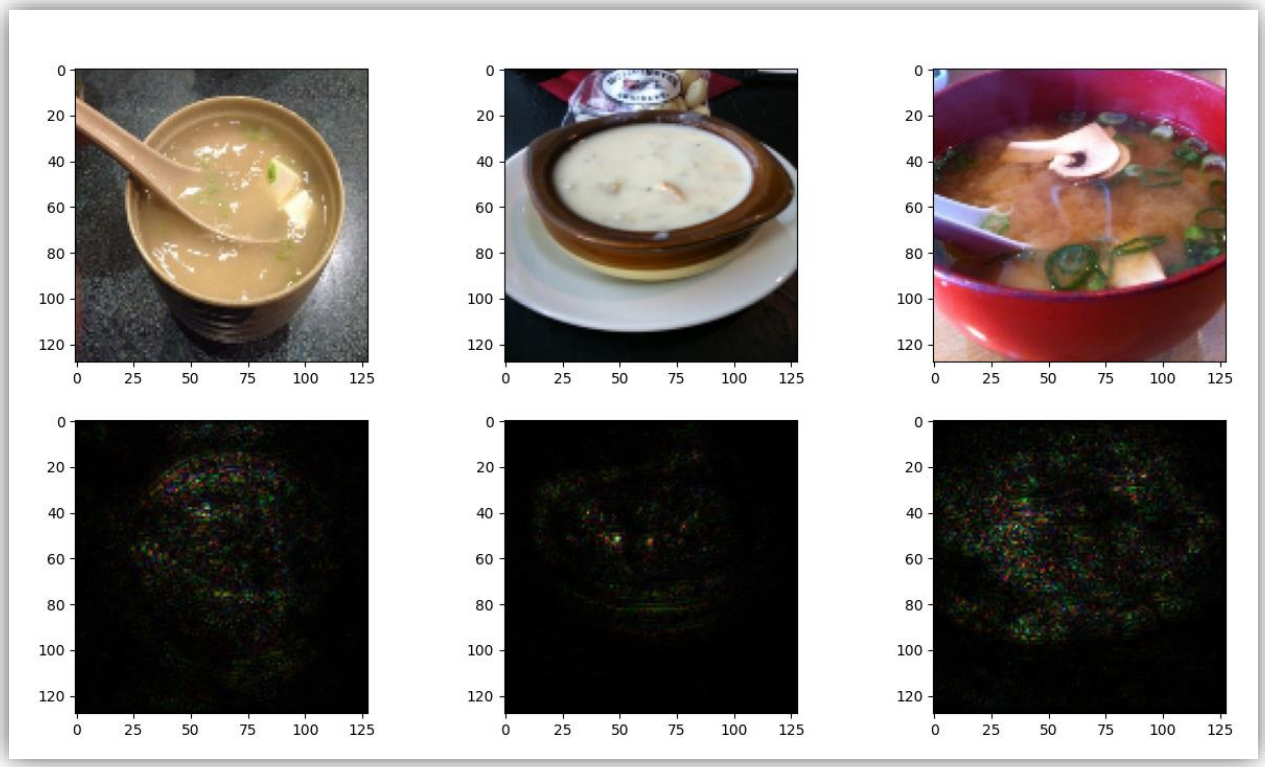
Class10:

亮點分散，主要分布在食物中顏色較亮處。



Class11:

亮點分布在有湯水的地方，整體而言比較分散，很難說是特別針對甚麼部分。



小結語:整體來說感覺 model 的辨識效果還可以，每個食物幾乎亮點都出現在食物的外框或食物本身，然而有些圖片亮點不是很明顯，我認為這有可能是遇到 gradient saturation，因此畫 saliency map 的方法給我的感覺是雖然比較直觀，但是是否準確其實還是很難說，畢竟很難說因為這個點的 gradient 比較大，就代表他一定是整個 model 判斷的重點。

但若真的要判斷這個 model 對圖片做分類的依據的話，從結果看起來我認為食物密集處的亮點較多，很有可能 model 是專注在分析整張圖片某部分相似 pixel。

假如有一部分 pixel 附近都很相似，那 model 就可以根據這一部分 pixel 的值去分析他的分類。

2. (3%) 承(1) 利用上課所提到的 gradient ascent 方法，觀察特定層的 filter 最容易被哪種圖片 activate 與觀察 filter 的 output。(Collaborators:)

答：

CNNid:15（顯示通過第 15 層 cnn 時的結果）

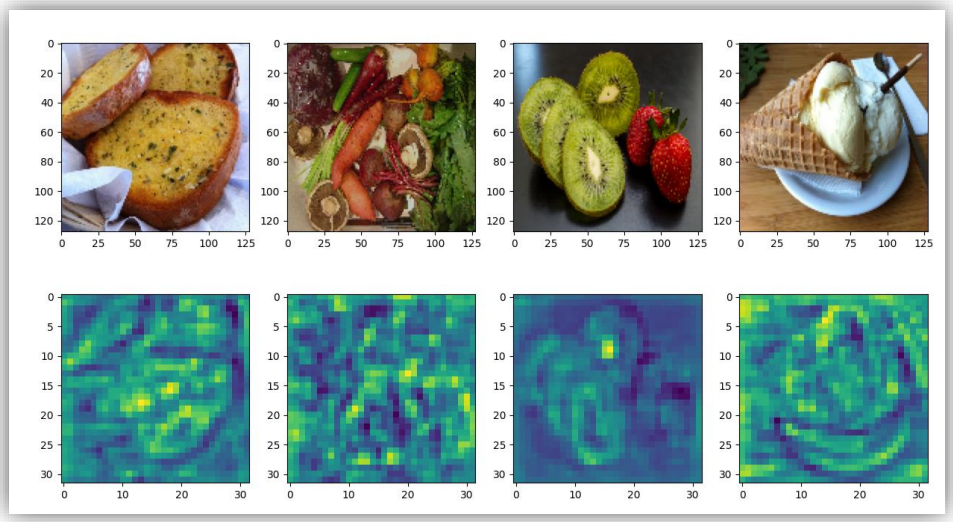
此層為一個256＊3＊3的 filter

```
nn.Conv2d(128, 256, 3, 1, 1),
```

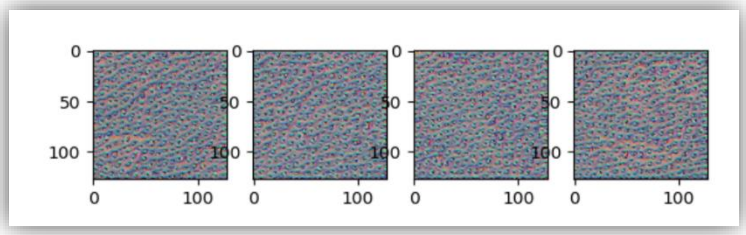
filter:0（顯示第 1 個 filter 的 activation 結果 以及 maximize 第一個 filter 的 x）

進行方式:每個 class 挑一張照片觀察結果，每張照片上有 4 個 class，共有 6 張照片

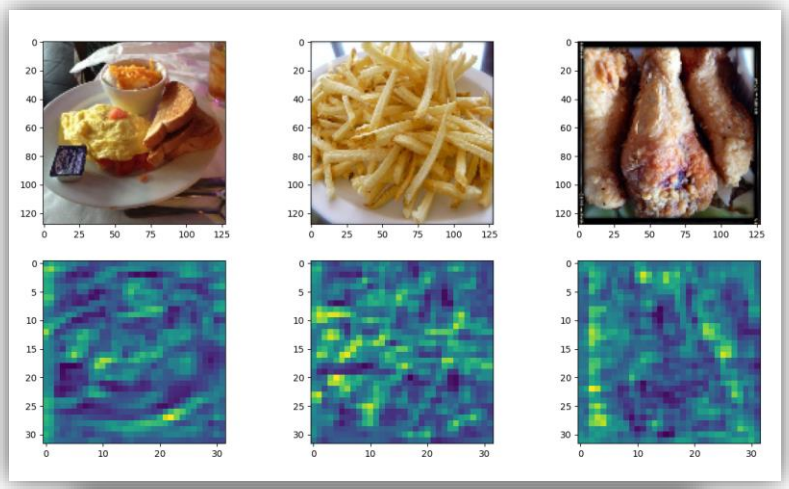
從 filter activation 可看到這層 filter 成功找出了食物大致的輪廓以及線條。

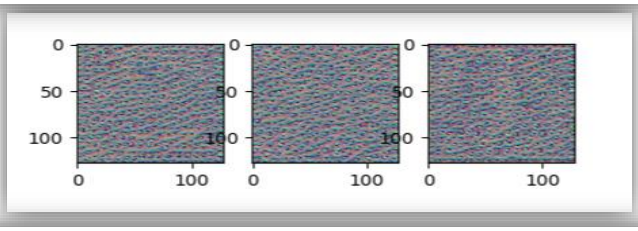


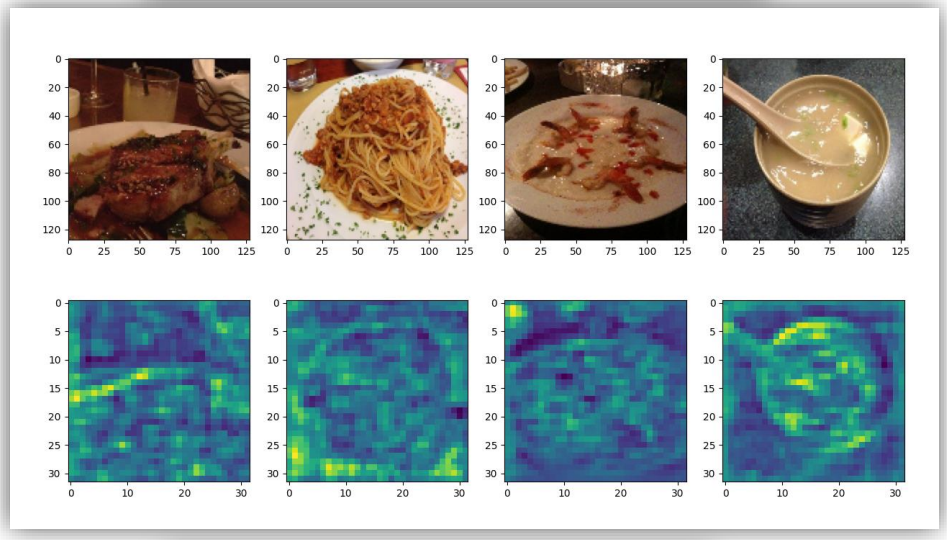
Input 是一堆看不懂的東西，但遠看的話會發現稍微有一些食物輪廓的感覺，近看的話還可以發現原來你有密集恐懼症。

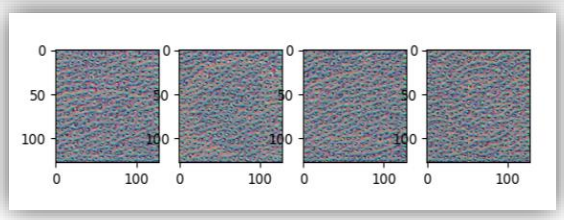


其他照片結果也很類似，activation 證明可以找到大致上的輪廓，input 稍微顯示了些微的輪廓，可見這一層的目的大概是要偵測輪廓。









第二個 filter:

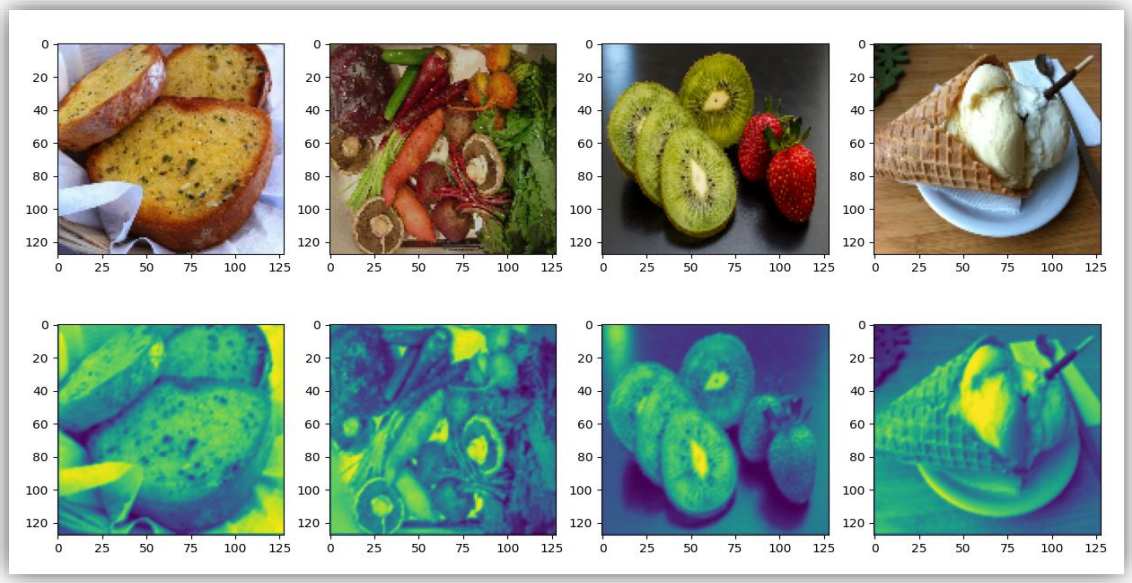
CNNid:1（顯示通過第 1 層 cnn 時的結果）

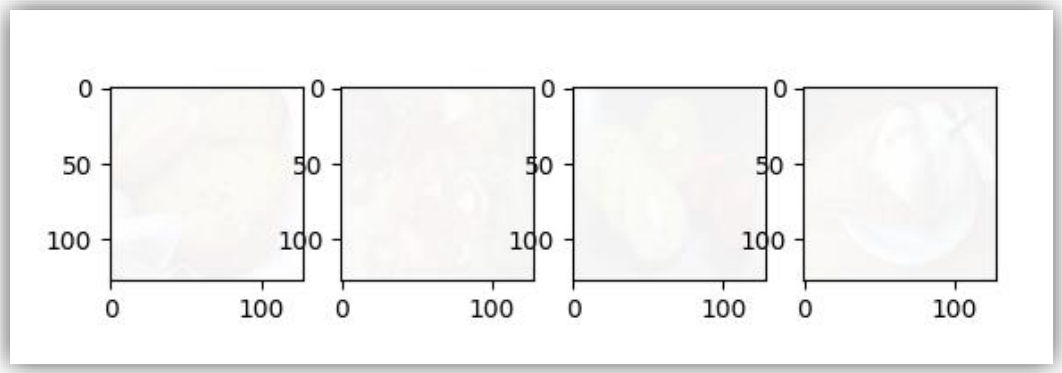
此層為一個64 * 3 * 3的 filter

```
nn.Conv2d(3, 64, 3, 1, 1)
```

filter:1(顯示第 1 個 filter 的 activation 結果 以及 maximize 第一個 filter 的 x)

進行方式:選四個 class 進行結果分析。





你可以看到 activation 的結果跟原本 input 的圖片長得幾乎差不多，我認為這是因為整個照片只經過第一次 convolution layer 處理，所以變化還不是很明顯，畢竟這是第一層 convolution layer 的第 10 個 filter 的結果。
然而假如仔細看 filter visualization 的結果，你會發現這近乎全白的圖片中，可以勉強看到一點原本圖片的輪廓以及淡淡的顏色。
因此我認為這個 filter 主要的工作，就是分析圖片中這些細微的顏色變化跟輪廓變化趨勢，儘管變化還很不明顯。

第三個 filter:

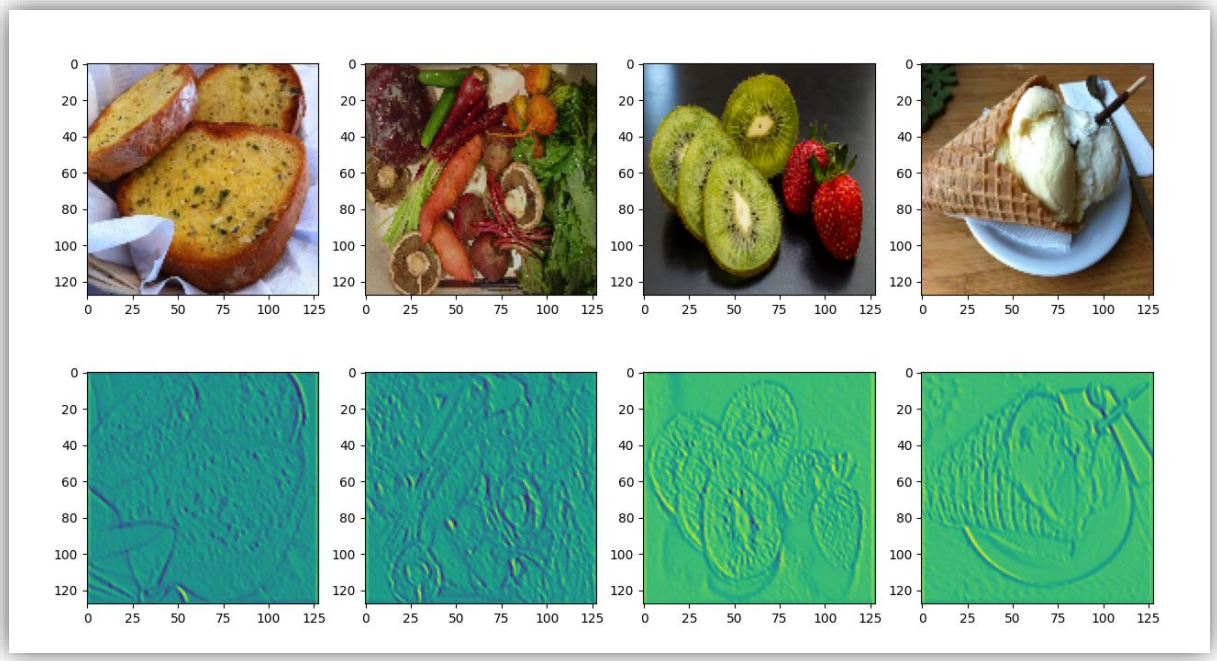
CNNid:5（顯示通過第 5 層 cnn 時的結果）

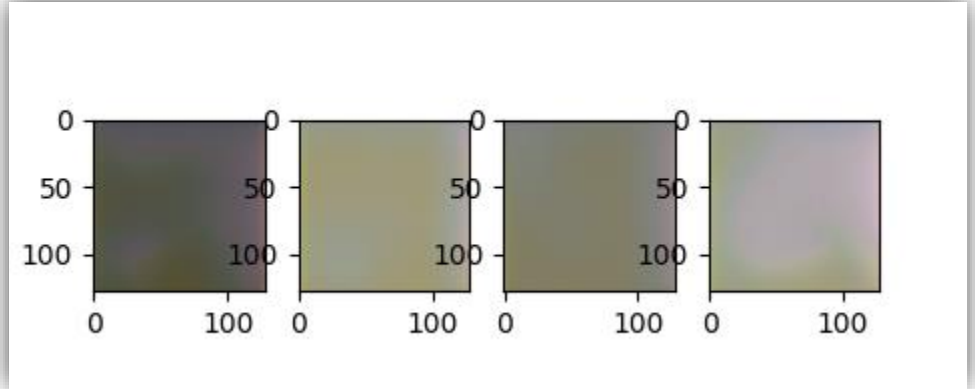
此層為一個 128* 3 * 3的 filter

```
nn.Conv2d(64, 128, 3, 1, 1),
```

filter:10(顯示第 10 個 filter 的 activation 結果 以及 maximize 第一個 filter 的 x)

進行方式:選四個 class 進行結果分析。



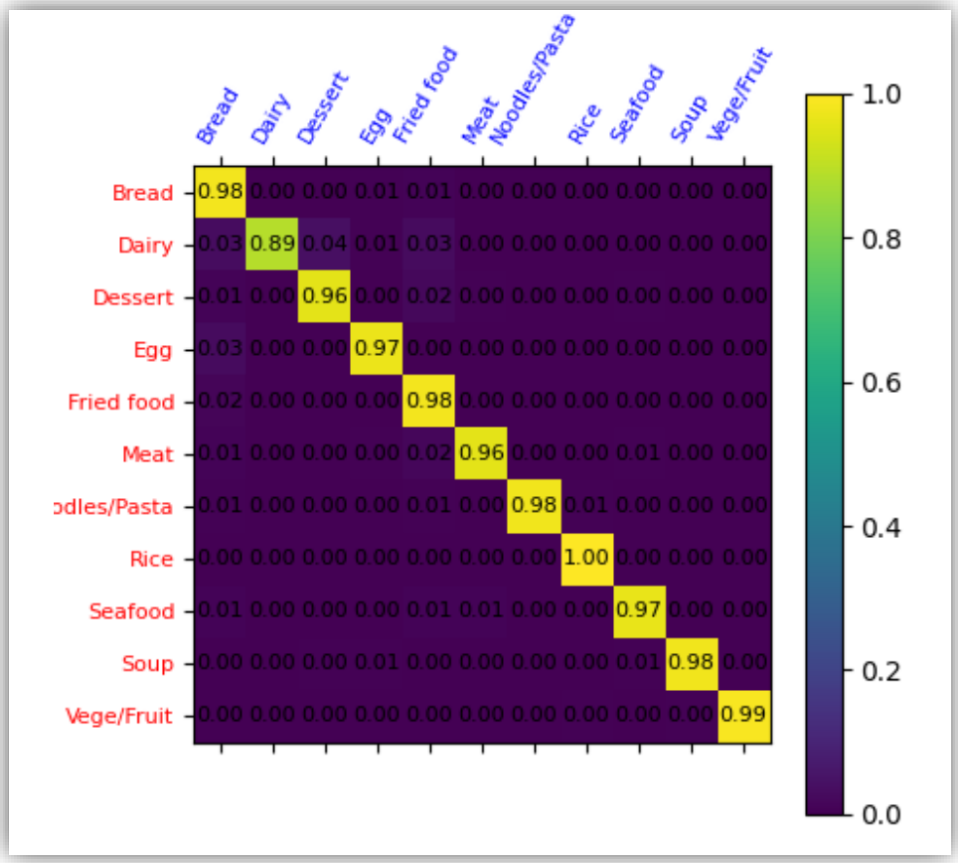


雖然 filter visualization 是四團糊的要命的東西，但是 activation map 相對就非常的清楚，可以看到 activation map 很像拿樹枝在沙地上畫的圖，雖然顏色已經不太相像，但是食物的線條以及食物外框裡面的紋理都非常明顯。

因此顯然這一層的 filter 的工作就是把圖片過濾後留下一些比較明顯的線條紋路。

3. (2%) 請使用 Lime 套件分析你的模型對於各種食物的判斷方式，並解釋為何你的模型在某些 label 表現得特別好（可以搭配作業三的 Confusion Matrix)。

答：



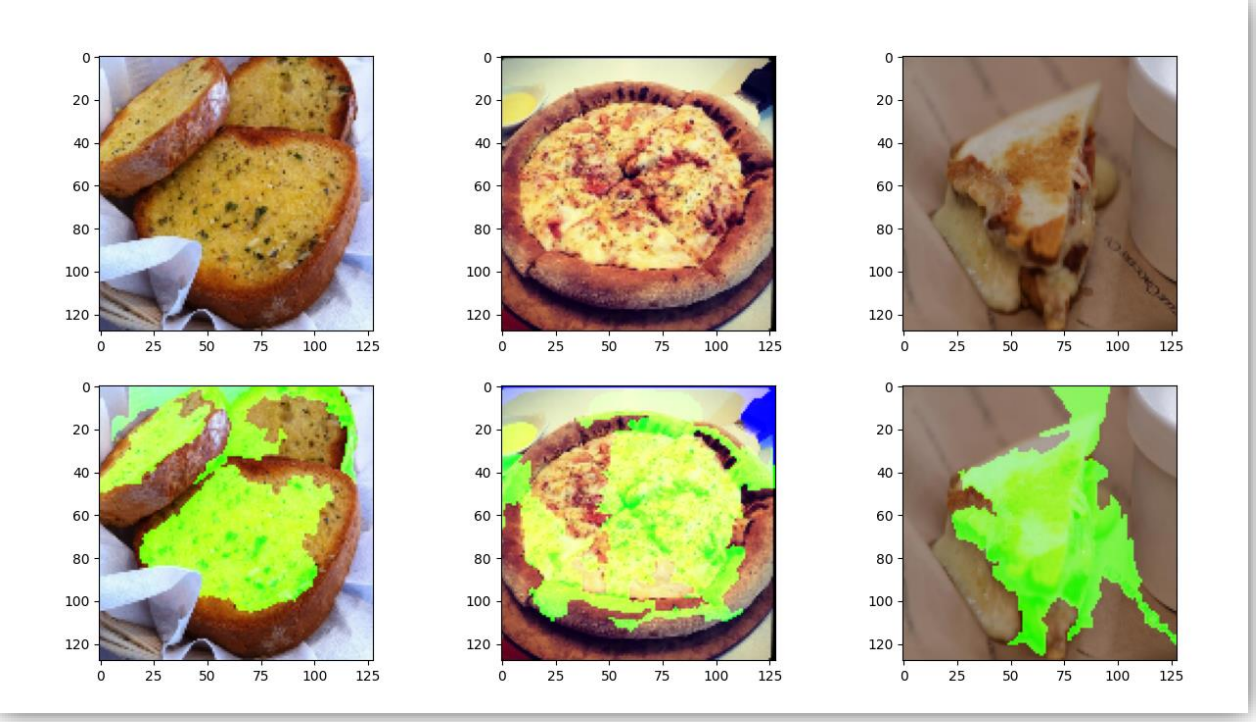
紅色代表實際 label

藍色代表預測結果

因為大部分的預測都蠻準確的，所以我將討論為何一個 class 有可能被誤判成其他 class，以及為何有些 class 表現特別好。

我討論的 class 有 1.Bread 2.Vegetable/Fruit 3.Dessert

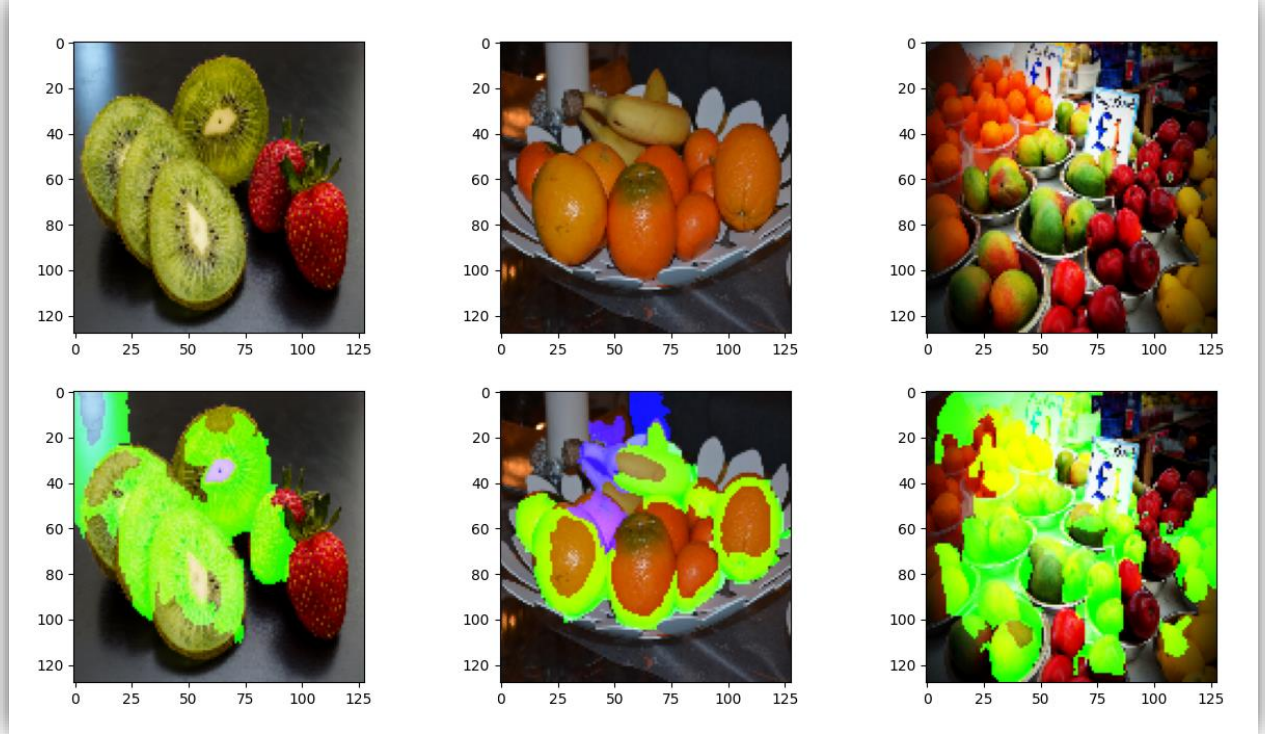
Class1 : BREAD



BREAD 在 confusion matrix 中可發現，他最常跟 EGG 去搞混，因此我們可以觀察一下 BREAD 的綠色區域，你可以發現除了第三張比較完美的符合整個麵包的形狀以外，其他兩張的綠色區域都不算很吻合，而且形狀幾乎都是糰狀。

若我們假設這個 model 判斷 bread 是根據形狀去判斷的話，那它自然很有可能去把 bread 誤認成是 EGG，因為這兩種東西形狀很像。

Class2: Vegetable and fruit

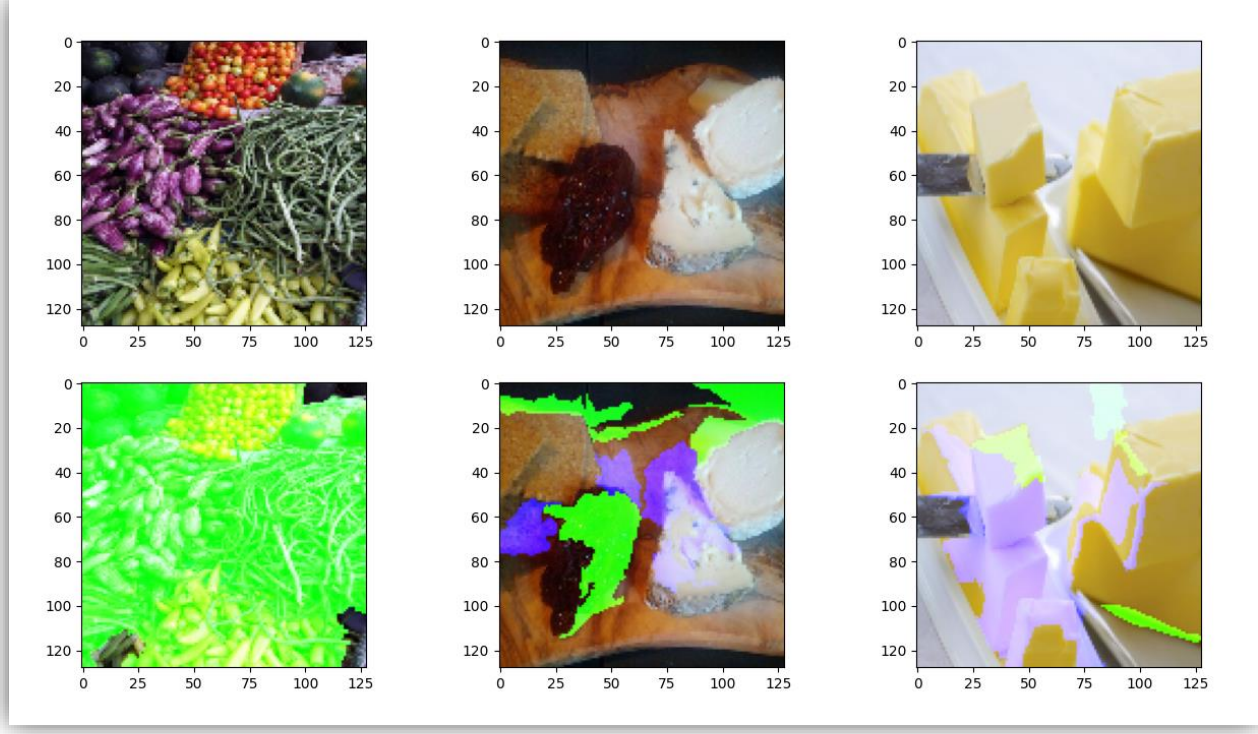


這個 class 是整個 confusion map 中辨認的第二好的部分(由於我認為 RICE 的 LIME 結果並不好因此忽略)，我的猜測如下。

Fruit 每次都是好幾個同樣的種類擺在一起，而且跟麵類相比，比較能區分出每一個形狀，你可以看到第二張照片，綠色幾乎圍住了每一個水果。

當 model 看到一張照片內有很多個同樣的東西，且可以找出每一個的外框，他可能就會猜測這是水果，因為其他種食物都比較難以有很多”個”特樣的特徵，要馬是無法用”個”去區分，要馬就是數量不會有水果來的這麼多。

Class3:



Dairy 是 model 中辨識率最低的一個 class，觀察圖片可以發現，activation map 中除了第一張圖片以外，其他兩張都沒辦法很準確地蓋到有食物的部分，可能 model 本身就無法獲取食物正確位置的資訊。

以奶油為例，被綠色覆蓋到的部分相當的少。

假設 model 總是認為存在食物資訊的 Pixel 是無用的話，那他可能就會忽略掉一些辨識 dessert 的重要資訊，導致 accuracy 不好。

4. (3%) [自由發揮] 請同學自行搜尋或參考上課曾提及的內容，實作任一種方式來觀察 CNN 模型的訓練，並說明你的實作方法及呈現 visualization 的結果。

答：

SMOOTH GRAD:　　Reference:助教上課內容 More about explainable AI ppt P.8

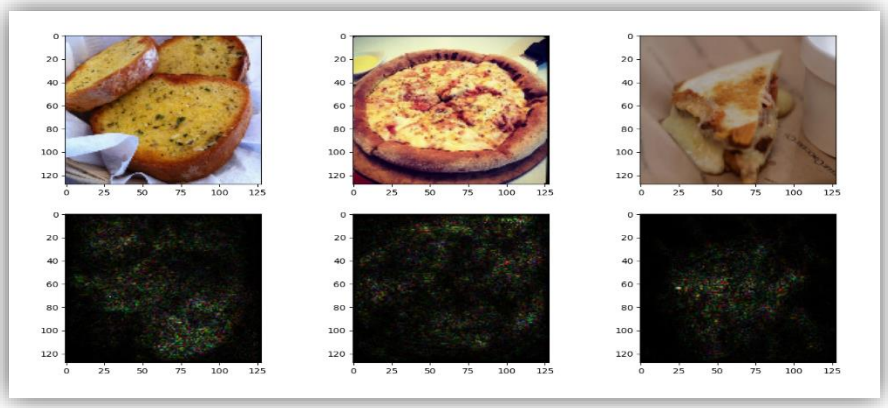
Question1 使用的 Salinecy MAP 是使用 Local gradient-based 的 vanilla gradient，然而這樣的方法受到 image 本身可能存在的 noise 影響，他的 gradient 可能為顯得很雜亂，因此可以用 Smooth Grad 這個方法去解決。

$$\hat{M}_c(x) = \frac{1}{n}\sum_{1}^{n} M_c(x + \mathcal{N}(0, \sigma^2))$$

其中 Mc(x)代表 image 的 gradient，N(0, σ^2)代表 Gaussian distribution 求出來的 noise，在把 input 丟給 model 之間，每一張圖片都會先加上一次 noise，並取 gradient，最後取 n 次結果的平均。

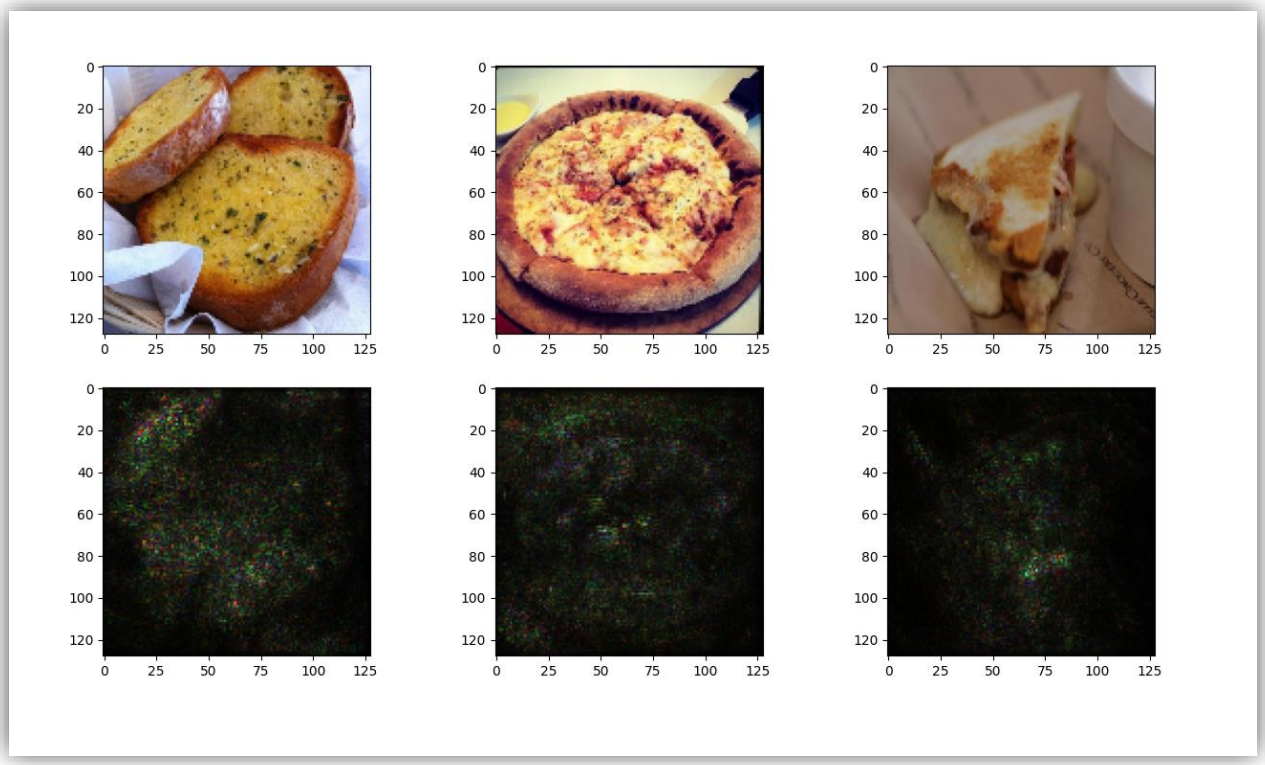
進行方式: 我將針對 question1 的前四張圖片，並觀察取用不同的 sigma 對結果的影響

先放上原圖:



1.n=10 σ=Max(x)-Min(x) （image pixel value 的全距）

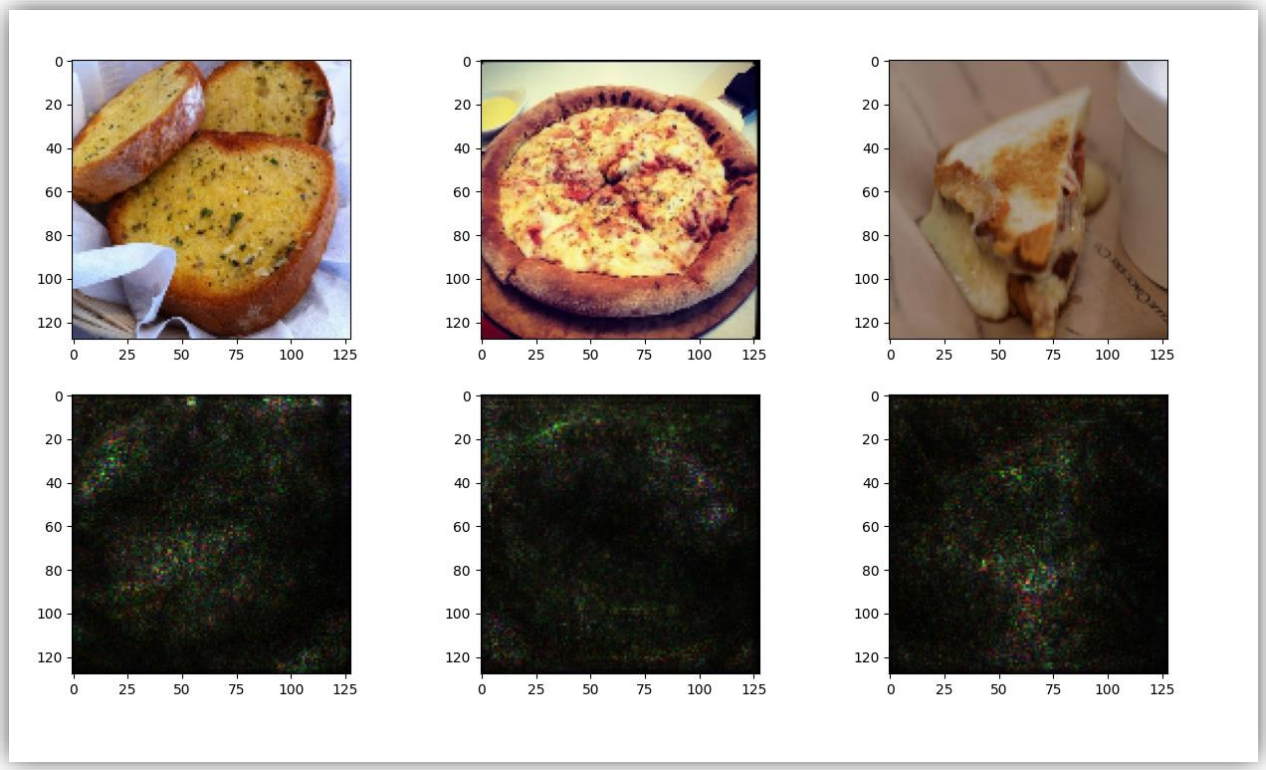
檔案名稱:Smooth_grad_Sigma1.png



變化仍然不太明顯，不過第三張圖逐漸出現了三角形的形狀，pizza 的形狀也變得更加明顯了，顯然若將σ繼續往上調可以得到更平畫的曲線。

2.n=10 σ=2*(Max(x)-Min(x)) （2 倍 image pixel value 的全距）

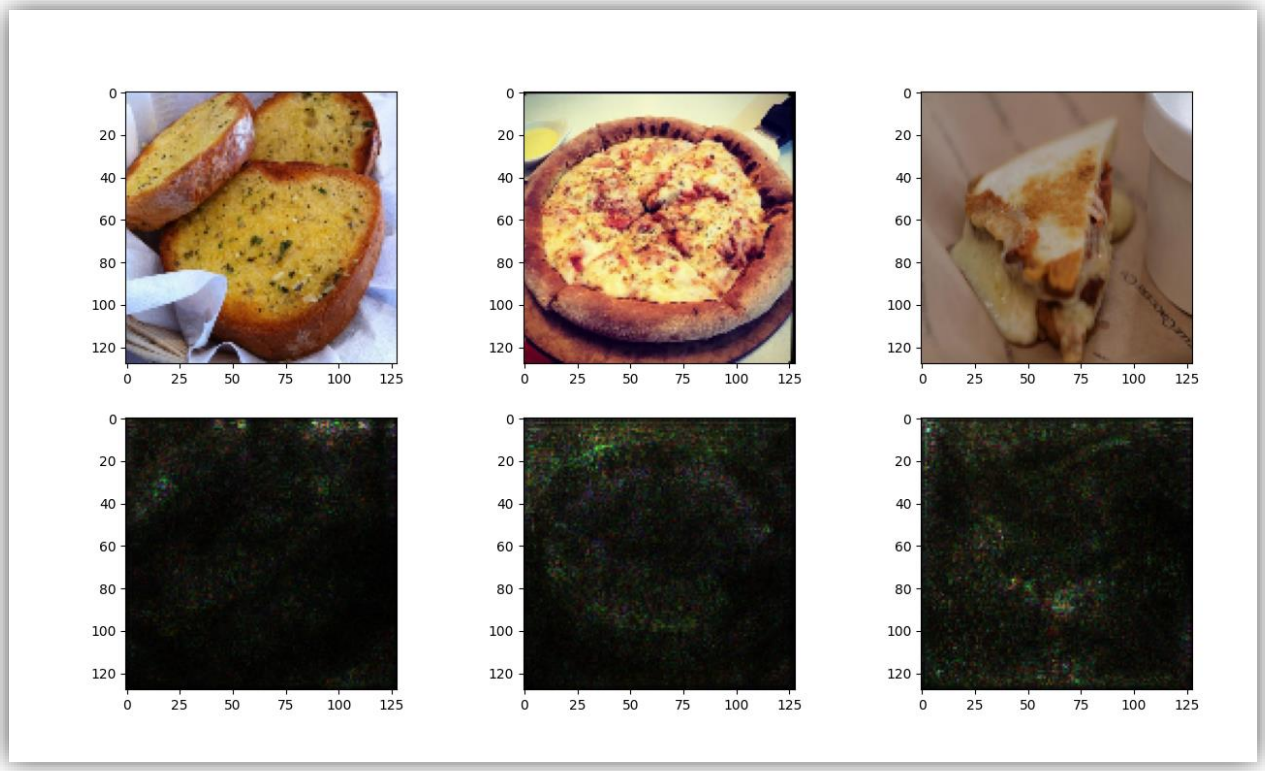
檔案名稱:Smooth_grad_Sigma2.png



第二張圖片的亮點漸漸變的不太明顯，第三張圖片的形狀越來越跟食物相似。

3.n=10 σ=3*(Max(x)-Min(x)) (3 倍 image pixel value 的全距)

檔案名稱:Smooth_grad_Sigma3.png



很意外的，第一張圖片結果幾乎整個爛掉，幾乎沒有任何亮點存在了

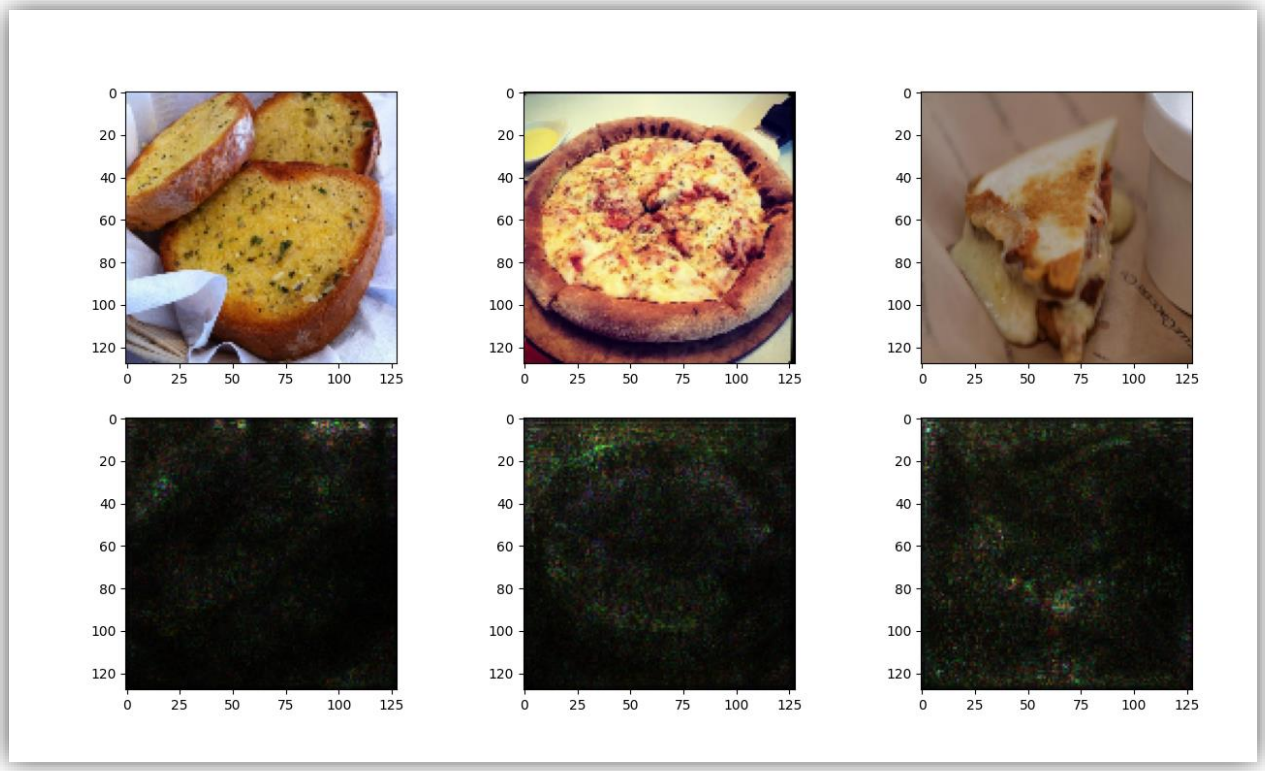
第二章圖片仍淡淡的保有 pizza 的外型，但整體也變得非常不明顯了。

第三張圖片幾乎已經看不出來他是甚麼食物了。

到這裡我想可以推斷 noise 的結果已經對原本圖案影響過大了。

4.n=10 σ=4*(Max(x)-Min(x)) (4 倍 image pixel value 的全距)

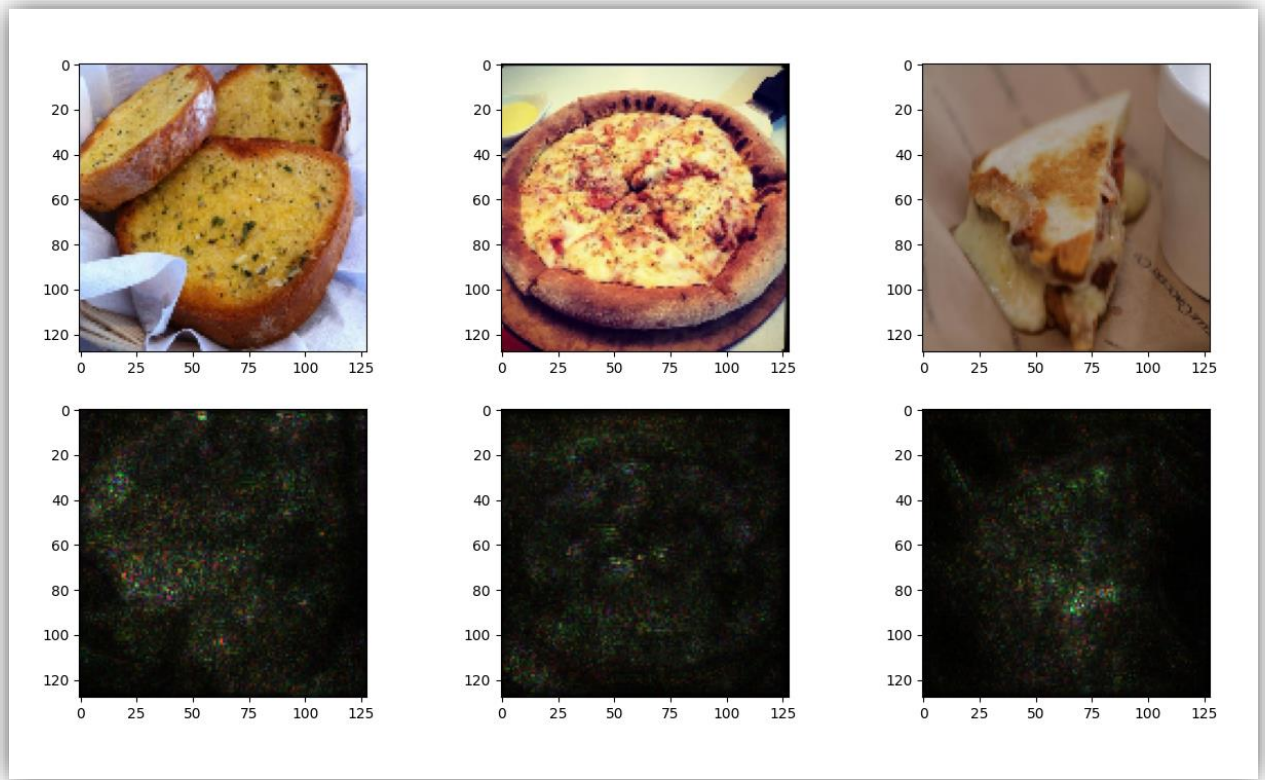
檔案名稱:Smooth_grad_Sigma4.png



正如上面所述，noise 已經過大，所以結果自然也不會太好，只剩下第二張照片還能看出食物的外型了。

5. n=50 σ=3/4*(Max(x)-Min(x)) (3/4 倍 image pixel value 的全距)

檔案名稱:Smooth_grad_Sigma5.png



這張照片算是我找到前幾好的結果了，雖然沒辦法看到明顯形狀，但大部分亮點分部位置跟食物是吻合的。

結語:Smooth Grad 的結果並沒有我想的好，本來認為這個技術能把亮點更集中一些(因為變平滑之後比較不會有極端的 gradient 變化，所以 gradient 高的地方應該會聚集在一起)，然而從實驗結果看起來大部分的亮點都變得越來越不明顯了，因此我認為 saliency map 不清楚的原因可能本身就不是出在 noise 上面，可能是 gradient 本身的數值的問題，也有可能是 gradient saturation

GRAD TIMES IMAGE:

為了呈現出更清楚的 saliency map，我決定再嘗試一種 visualization 的方法-Gradient times image。

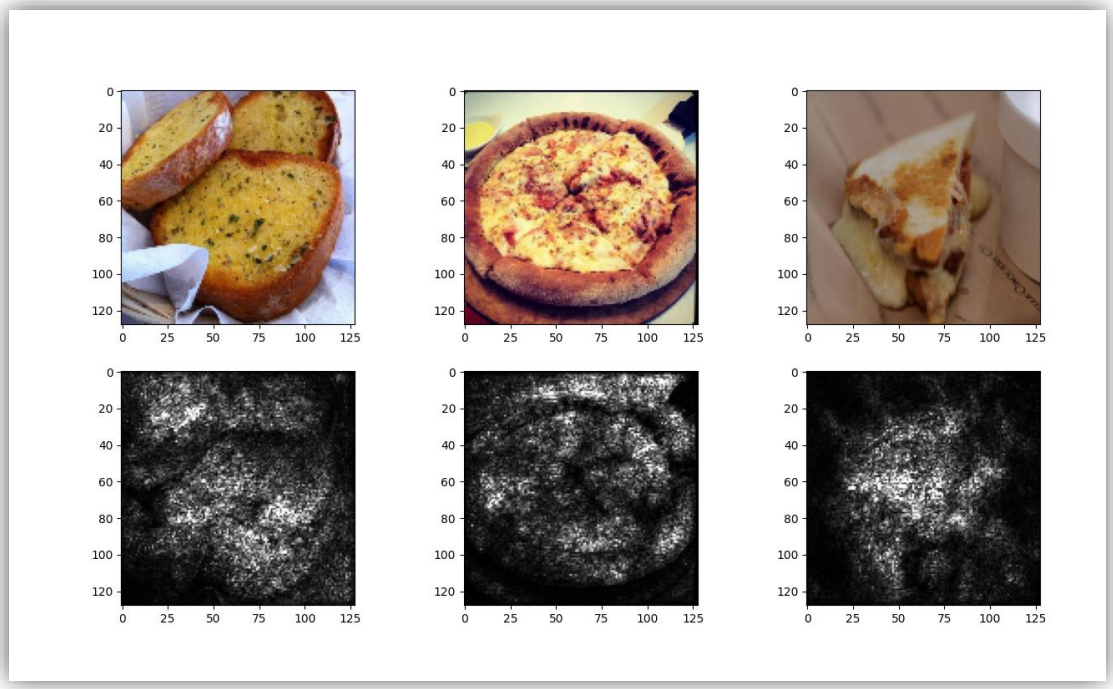
這個方法是直接把 image 的 gradient 乘上原本的 value，這麼做能夠讓 gradient 比較大的地方變得更明顯，gradient 小的地方變的不明顯，如此便能將差距拉開。因此能夠得到更清楚的 saliency map。

Reference:

1. https://github.com/utkuozbulak/pytorch-cnn-visualizations#grad-times-image

2. Shrikumar, P. Greenside, A. Shcherbina, A. Kundaje. *Not Just a Black Box: Learning Important Features Through Propagating Activation Differences* https://arxiv.org/abs/1605.01713

我使用一般的 local based graident 的方法(question1)並將 gradient 的結果乘上 input。



我將結果以灰階圖來表示，結果還不錯，第二張圖片甚至能明顯看見 pizza 的外型。

當然我認為其中一個原因是以 gradient 的角度而言，灰階圖一定更能夠看出明顯的變化。

我認為這個方法最大的好處是，他保留了原本 image 的 pixel value 資訊，所以我們可以猜測結果應該長相會更像原本 image 的樣子。

不過還是可以觀察到食物的外型輪廓仍然沒有很清楚的線條，我認為要解決這個問題可能就需要一些改良的 Gradient 方法，如 Deep LIFT、Integrated Gradient 等等。

整體來說我認為 saliency map 的方法帶給我最直觀的感受，因為可以直接從圖片中去看到哪一部分亮點比較多，藉此判斷其對於 prediction 的影響性。