**Problem1:  
 Please explain what is the residual block (two types) and give the pros and cons of each.**

Residual Block可以避免梯度消失問題，減少loss卡在critical points的機會。

結構示意圖:

一張含有 文字, 圖表, 字型, 行 的圖片

自動產生的描述

Basic Residual Block Bottleneck Residual Block

1. **Basic Residual Block**

由兩個連續的3x3 convolution layers組成，output為通過兩層convolution layers計算的結果與input之相加(shortcut connection)，這樣可以避免在遇到critical points時立刻停下，導致容易卡在鞍部而不是minimum的問題，進而繼續降低loss。

**Pros:**

相較於Bottleneck Residual Block結構更為簡單，對於較淺的network比較合適。

**Cons:**

計算效率較差、參數多，對於層數較深的模型表現不如Bottleneck Residual Block好。

1. **Bottleneck Residual Block**

由三個連續的convolution layers組成，分別為1x1、3x3、1x1，同樣output為經過三層convolution layers的data與input相加。

此種結構是為了減少參數的使用以及加快計算效率，頭尾的1x1 convolution layers可以在保留原始數據特徵的情況下增減data的維度，且數據皆不受其他數據所影響，因此第一層(1x1)對數據降維來減少參數的使用，第二層(3x3)可以學習局部空間特徵，再經由第三層(1x1)將維度升回，讓其可以與input相加。

**Pros:**

減少參數及計算量，讓模型更容易訓練，訓練速度也較快，適合層數較深的模型。

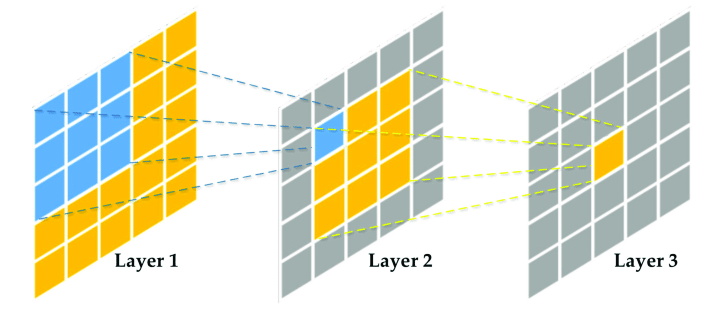
**Cons:**

結構較複雜，不適合較淺的模型。

**Problem2:**

**Please explain what is the receptive field and how to adjust the receptive field in the neural network.**

Receptive field是代表可以影響計算過程中一個像素的範圍，如下圖，Layer 1藍色部分即為Layer 2藍色像素的Receptive field。



它的大小決定了該模型捕捉局部/全局特徵的能力，若圖片中的主體很大，receptive field就不適合太小，反之若主體小，就不太大的receptive field，但也要注意越大的receptive field也就代表越大的計算量及越差的計算效率。

另外，pooling、stride也是造成receptive field改變的原因之一，若想要增加邊緣的receptive field，可以適當的增加pooling來達成，想要更快速的掃過大面積視野，就可以增加stride大小。

**Problem3:**

**Please give some methods to achieve feature map upsampling. Explain them with codes and images.**

Upsamping的功能是提升圖片的分辨率，但單純的放大圖片並沒有辦法增加圖片訊息，因此透過不同upsampling的方式使放大後的圖片擁有更多的訊息。以下為幾種常見的upsampling方式。

1. **Nearest Neighbour Interpolation**

PyTorch:

torch.nn.functional.interpolate(input, size, scale\_factor, mode='nearest', align\_corners, recompute\_scale\_factor, antialias)

直接將距離最近的像素數據填充至放大後的像素中，這種方式簡單快速，但也無法有效增加圖片訊息。

一張含有 螢幕擷取畫面, 圖表, 行, Rectangle 的圖片

自動產生的描述

1. **Bilinear Interpolation**

PyTorch:

torch.nn.functional.interpolate(input, size, scale\_factor, mode='bilinear', align\_corners, recompute\_scale\_factor, antialias)

這種方式與Nearest Neighbour Interpolation類似，但是填充將該點包圍的四點之加權平均，以兩點圍起部分之矩形面積與總面積的比例作為權重，如下圖，P為要填充的點，Q11、Q12、Q22、Q21為四個原先存在的點。這個方法可以有效避免鋸齒的出現。

d

c

一張含有 文字, 行, 螢幕擷取畫面, 繪圖 的圖片

自動產生的描述

b

a

一張含有 字型, 行, 文字, 螢幕擷取畫面 的圖片

自動產生的描述

1. **Unpooling**

PyTorch:

torch.nn.MaxUnpool2d(kernel\_size, stride, padding)

這個方式是先紀錄原先做Max pooling時最大值的位置，在完成中間的計算後，將結果填充至紀錄的位置，其餘位置則補上0，這樣可以最大限度保留原訊息。

一張含有 文字, 圖表, Rectangle, 正方形 的圖片

自動產生的描述