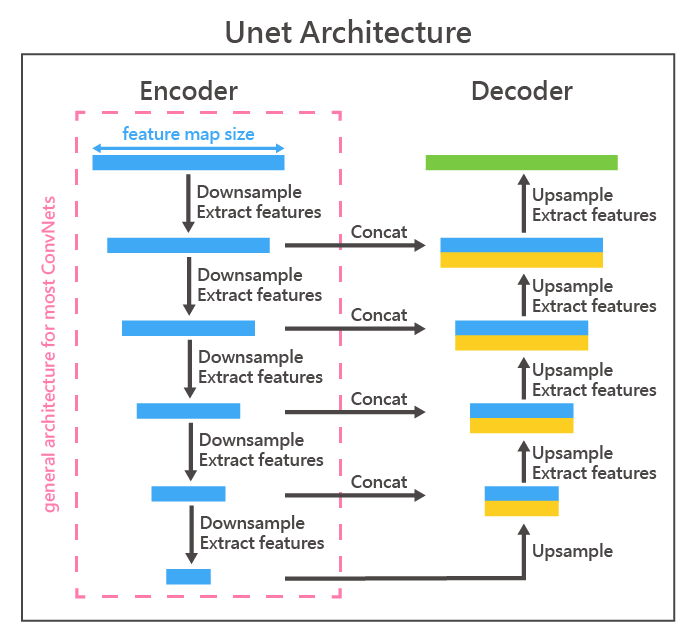
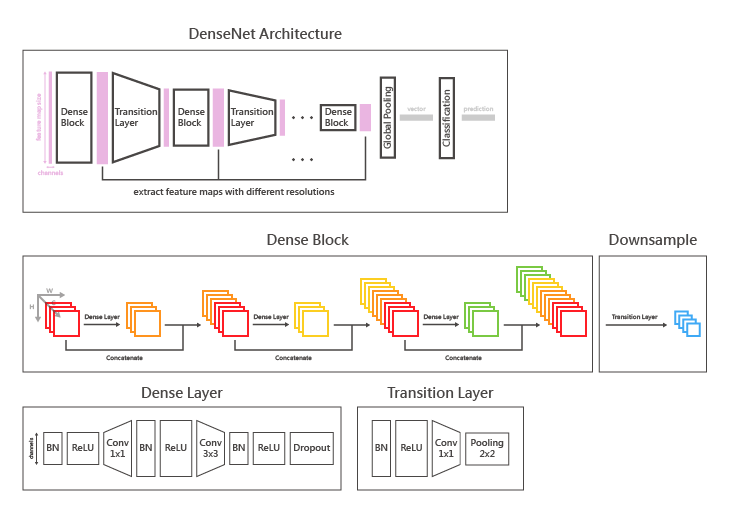
**UNet**

1. Architecture



**DenseNet**

1. Architecture



1. Text description

DenseNet follows a serial numbers of operation composed of extracting feature maps with dense layer and then reducing feature map resolution inside transition layer (shown in DenseNet Architecture).

Inside each dense block, there are several dense layers and they all output new feature maps with same number of channels, and the output will be concatenate with the input.

For example, an input feature map with 32 channel goes through a dense block with 3 danse layers that output 16 chennels feature map:

32[input] -> 16[output1] -> 16+32[concat] -> 48[input] -> 16[output2] -> 16+48[concat]

64[input] -> 16[output3] -> 16+64[concat]

1. Advantages and disadvantages

Advantages:

1. Gradient vanishing problem throughout back propagation is less likely to happen because there are directly connected route for each output.
2. Reusing of feature maps from lower to higher level makes the model focus on all different receptive field features more easily.
3. The final output number of channel of the dense block is the sum of the number of each layer’s output channel. This means each layer don’t need to output many channels, hence reduces the number of parameters.

Disadvantages:

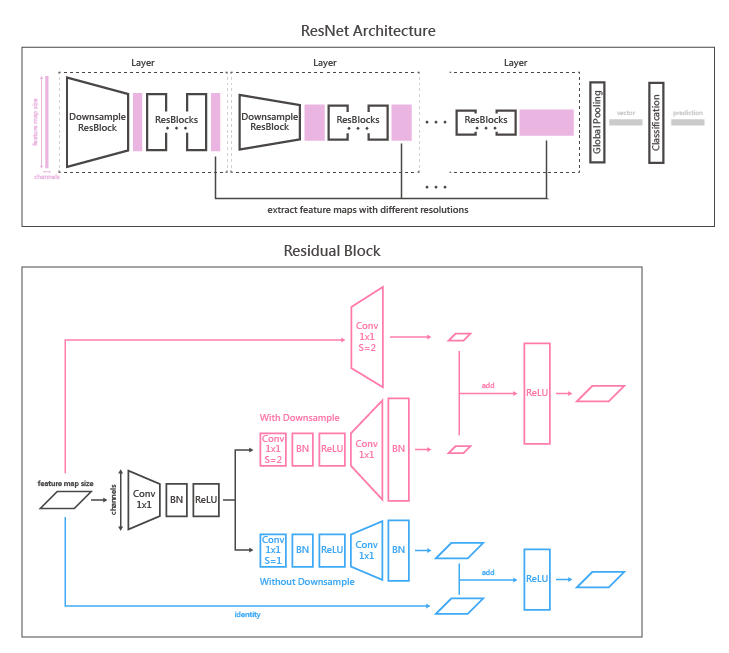
1. Reusing feature map is memory consuming and complex when doing back propagation.
2. Performance explanation

The reason why it doesn’t perform well on our experiment maybe due to each layer inside dense block is not extracting enough information. Each dense layer gives feature map with only a small number of channels. Maybe our use case need to more focus on large receptive field to perform well. So the channel count of higher level feature map is more import. DenseNet gives same number of channels for all feature map levels.

1. Reference
   1. <https://ithelp.ithome.com.tw/articles/10265328>
   2. <https://arxiv.org/pdf/1608.06993.pd>

**ResNet**

1. Architecture



1. Text description

This is the most commonly seen CNN model. It can be very deep without facing the gradient vanishing problem. The ideal of adding the input directly to the output, so called residual block, gives the previous layer the chance to directly calculate the gradient and update throughout back propagation, so it prevents the gradient vanishing problem when the model is deep. And for the reason that the model is widely used and it’s well known property, we choose ResNet as our solid baseline model.

1. Advantages and disadvantages

Advantages

* 1. It easy to implement and the performance is good, so it’s suitable for a startup.
  2. Residual block prevents the gradient vanishing problem when the model is deep

Disadvantages

1. The model can be very deep so the number of parameters of the model is huge.
2. Performance explanation

It’s a model from 2015 and many models are based on it and more specialized. So it’s not the best one, but it’s still gives us an insight that we can use deep learning method to work on the task.

1. Reference
   1. https://zhuanlan.zhihu.com/p/31852747
   2. https://arxiv.org/pdf/1512.03385.pdf

**EfficientNet**

1. Architecture diagram
2. Text description

The concept of EfficientNet is to find a balance between computational efficiency and accuracy. They conduct a series of experiments to proof that scaling model’s depth (number of layers), width (number of channels), resolution (feature map size) at the same time is better than scaling only one of them.

And they gave the formula to make sure that the models are comparable when scaling:

****

* 1. N: the base neural network model:
     + s: the number of blocks for the base neural network model (7 in the paper)
     + Fi: the layer type for the i-th block
     + Li: the repeat number for Fi in the i-th block
     + X: the input feature map for that layer
     + Hi, Wi, Ci: the height, width and channel of the input feature map in i-th block
     + d, w, r: multiplier for Li, Ci and (Hi,Wi), controlling the scaling of depth, width, resolution of the baseline model respectively.
  2. Accuracy(N(d,w,r)): the accuracy of the baseline model scaled by d, w, r
  3. Memory(N) <= target\_memory: the memory usage should not exceed the target memory under all combinations of (d, w, r)
  4. FLOPS(N) <=target\_flops: the flops should not exceed the target flops under all combinations of (d, w, r)

And to further understand what is the relationship between flops and (d, w, r) and how to do scaling under the constraint, they gave the formula:



1. α∙β2∙γ2 ≒ 2: means that they want to restrict the scaled model not to exceed 2 times the flops than that of the unscaled model. The reason there are power of 2 for β and γ is that when you increase r the scaled feature map (r∙H,r∙W) will be r2 times bigger than (H,W) and when you increase the w the parameters of a convolution layer will be (w∙in,k,k,w∙out), w2 larger than (in,k,k,out).
2. Φ is an exponential controller. The number increased by 1, the flops will approximately times by 2 since α∙β2∙γ2 ≒ 2.

The search problem and restrictions are defined. It’s time to find a baseline model.

They use architecture similar to that use in MnasNet, including the search space and neural architecture search. The optimization goal is defined as ACC(m)\*[FLOPS(m)=T ]w, where ACC(m) and FLOPS(m) denote the accuracy and FLOPS of model m, T is the target FLOPS and w=-0.07 is a hyperparameter for controlling the trade-off between accuracy and FLOPS. They proposed EfficientNet-B0. It’s a derivative from MobileNet and uses MBConv blocks with Squeeze-and-excitation block inside and swish for activation function.

After the baseline model EfficientNet-B0 is designed, they search for the appropriate combination for (α, β, γ) and set Φ = 1. The result is (1.2, 1.1, 1.15) and the new scaled model is EfficientNet-B1. And they set Φ = 2, without searching for new combination for (α, β, γ), and create EfficientNet-B2, so on and so forth.



The EfficientNet family is able to keep high accuracy while using fewer parameters and running fast on CPU than other models. And it can be run on a mobile device.

1. Advantages and disadvantages

Advantages

1. Depthwise convolution can reduce parameters making the model smaller.
2. The depth, width, and resolution are well optimized, so it saves some time to find what combination of depth, width, and resolution is better. We can use it empirically for most tasks.

Disadvantages

1. depthwise convolution reduces the flops and make it possible to run on mobile device, but it has high latency when running on GPU, so the training process on GPU is relatively slow.
2. Performance explanation
   1. inverted bottleneck MBConv can preserve important information after non-linear operation.
   2. Squeeze-and-excitation is a little bit like attention. It helps model to focus on important channel.
   3. Swish activation function has better derivative property than ReLU making the model to converge at a better place.
   4. The combination of model’s depth, width, and resolution is optimized.

All the properties above maybe the reason why EfficientNet performs best.

1. Reference
   1. <https://medium.com/@jimmyyoung1995/%E8%AB%96%E6%96%87%E7%AD%86%E8%A8%98-ef%EF%AC%81cient-net-rethinking-82047aea363e>
   2. <https://medium.com/ai-blog-tw/efficientnet-v2%E7%9A%84%E8%83%8C%E5%BE%8C-%E9%87%8B%E6%94%BEmobilenet%E5%9C%A8gpu-tpu%E4%B8%8A%E7%9A%84%E6%95%88%E7%8E%87-f19abde55b05>
   3. <https://ttumiel.github.io/blog/mobilenet-to-efficientnet/>
   4. <https://medium.com/ai-academy-taiwan/efficient-cnn-%E4%BB%8B%E7%B4%B9-%E4%BA%8C-mobilenetv2-7809721f0bc8>
   5. https://medium.com/@hupinwei/%E6%B7%B1%E5%BA%A6%E5%AD%B8%E7%BF%92-senet-squeeze-and-excitation-networks-52ad0a7fd307
   6. EfficientNet: <https://arxiv.org/pdf/1905.11946.pdf>
   7. MnasNet: <https://arxiv.org/pdf/1807.11626.pdf>
   8. MobileNet V2: <https://arxiv.org/pdf/1801.04381.pdf>
   9. Squeeze-and-excitation Net: <https://arxiv.org/pdf/1709.01507.pdf>