1. Inception Model

Dong-Gyu, Lee

Dept. of Statistics, KU

2020, Apr 10

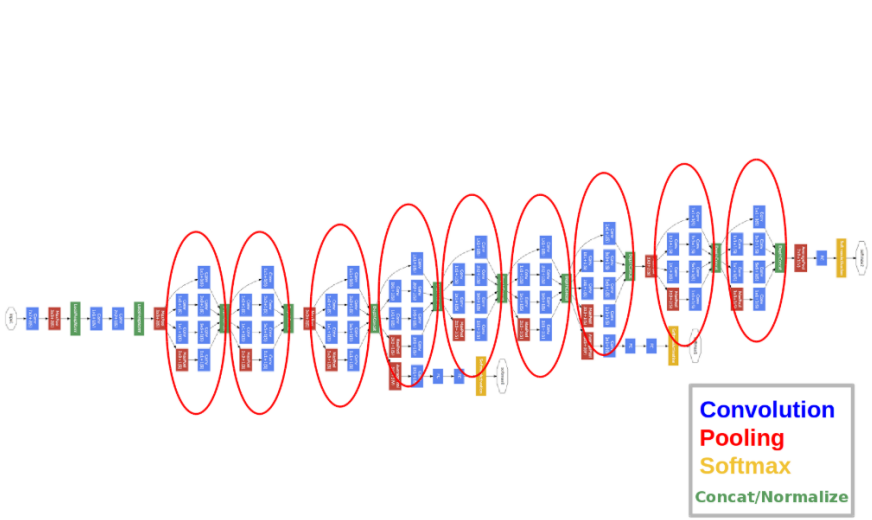
Contents

Today's Goal

- Based on GoogLeNet(Inception-v1), let's see a model that combines Inception and ResNet.
- Inception-v2/v3/v4 and Inception-Resnet-v1/v2 are covered in this section.

Inception-v1 Review

- The GoogLeNet(or Inception-v1) we learned in previous chapter is a CNN model with 22 layers and 5M parameters.
- It consists of 2 auxiliary classifier and 1 final classifier, coming from the global average pooling(GAP).



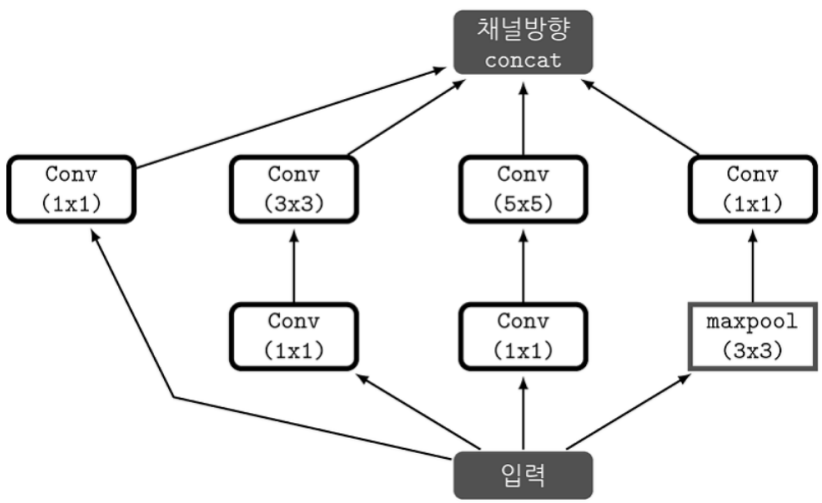


Figure 1: Inception-v1

Inception-v2/v3

- Inception-v2[3] was created to improve the performance of Inception-v1.
- And Inception-v3[3] is an optimized model by adjusting detailed options based on the Inception-v2 structure.

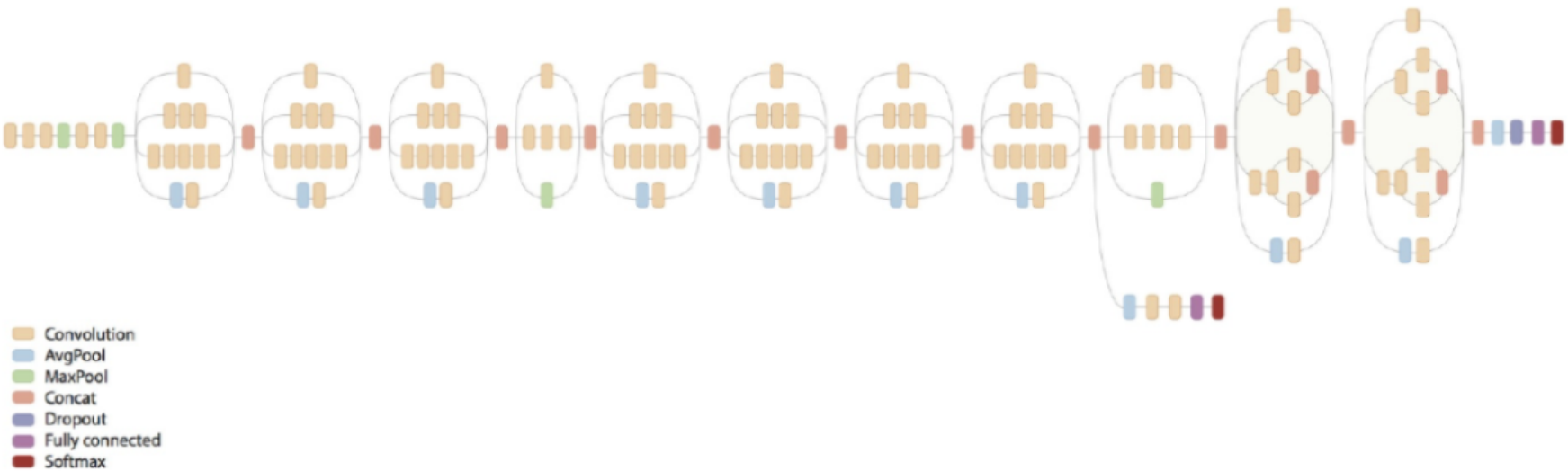


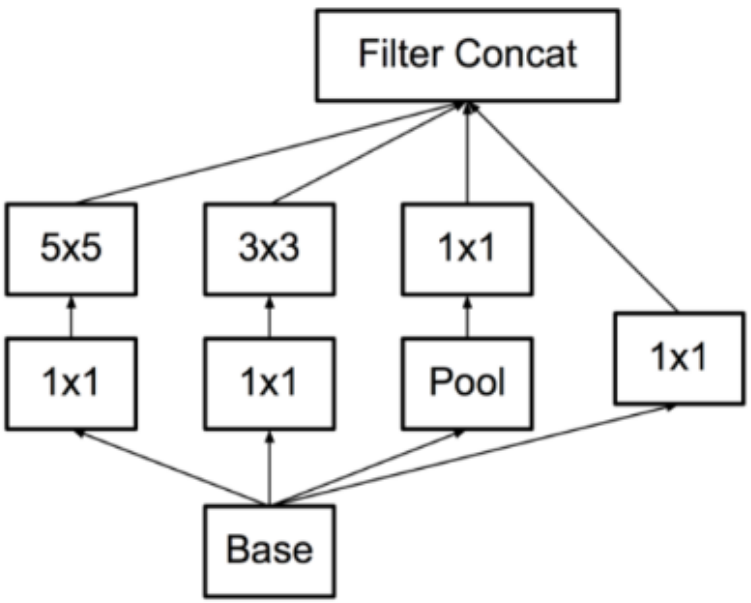
Figure 2: Inception-v2 Structure

Inception-v2/v3

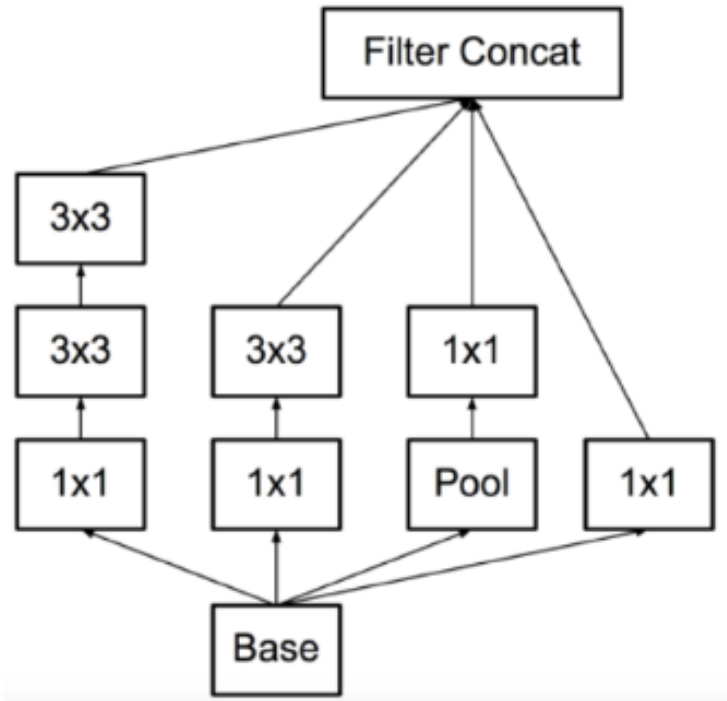
- Inception-v2/v3 have the following characteristics roughly :
  1. First auxiliary classifiers is deleted.
  2. Using only 3×3 filter. (Fig. 3)
  3. Factorization using asymmetric convolution. (Fig. 4)
  4. Efficient grid reduction. (Fig. 6)
- In addition, Inception-v2/v3 have its own criteria for calculating the number of filters in each stage, such as GoogLeNet.

Inception-v2/v3

- Using only 3×3 filter means that :



(a) Original Inception



(b) Imporved Inception

Figure 3: Using only 3×3 Filter

Inception-v2/v3

- Also, factorization using asymmetric convolution means that :

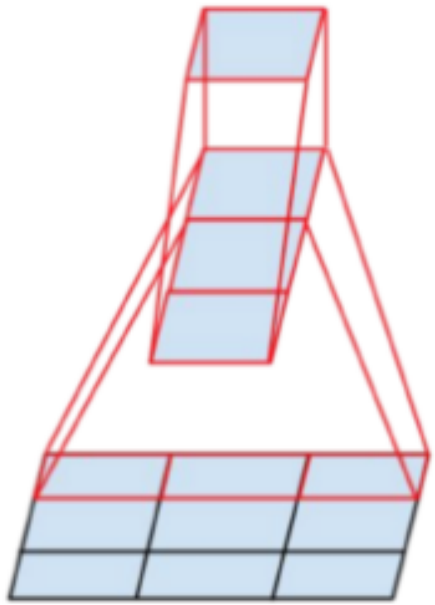
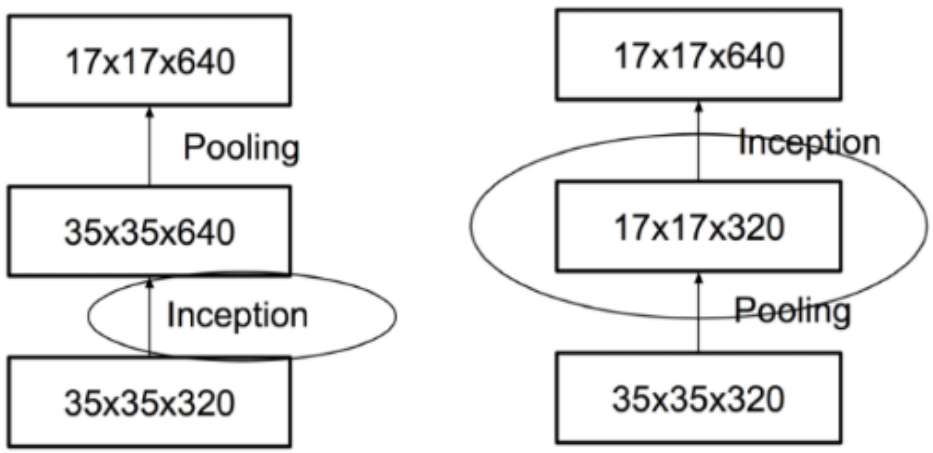


Figure 4: Asymmetric Convolution

Inception-v2/v3

- And last, efficient grid reduction means that :
  1. In general, if we reduce the input size using convolution operation, then increase the filter size for the purpose of trade-off.
  2. Therefore, the order of convolution and pooling is a major concern when they produce same output size.
  3. If you do pooling first, then 'representational bottleneck' occurs.
  4. Otherwise, the amount of calculation is quadruple. $(2(\frac{d}{2})^2k^2 \rightarrow 2d^2k^2$, $d = 35, k = 320$)

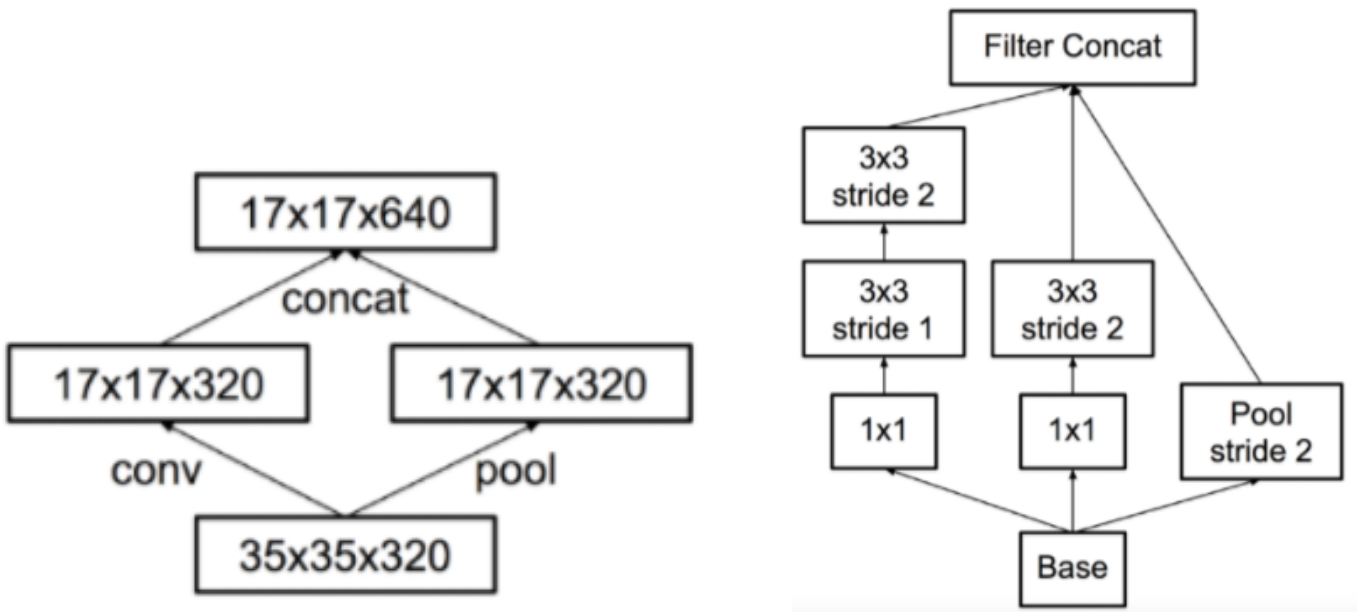


(a) Inception First (b) Pooling First

Figure 5: Inception & Pooling Trade-off

Inception-v2/v3

- So, consider the picture below(Fig. 6) that compromises these two.
- And this grid reduction will be used after the 'Inception-Block-A/B' operation.
- You can check the 'Inception-Block-A/B' in Fig. 8.



(a) Compromise (b) Final Reduction Shape

Figure 6: Grid Reduction

Inception-v2/v3

- So, the Inception-v2/v3 structure are as follows :

type	patch size/stride or remarks	input size
conv	3×3/2	299×299×3
conv	3×3/1	149×149×32
conv padded	3×3/1	147×147×32
pool	3×3/2	147×147×64
conv	3×3/1	73×73×64
conv	3×3/2	71×71×80
conv	3×3/1	35×35×192
3×Inception	As in figure 5	35×35×288
5×Inception	As in figure 6	17×17×768
2×Inception	As in figure 7	8×8×1280
pool	8 × 8	8 × 8 × 2048
linear	logits	1 × 1 × 2048
softmax	classifier	1 × 1 × 1000

Figure 7: Inception-v2/v3 Summary

- "figure 5,6,7" in Fig. 7 are on the next slide.

Inception-v2/v3

- Each Inception Block-A,B,C has the following form :

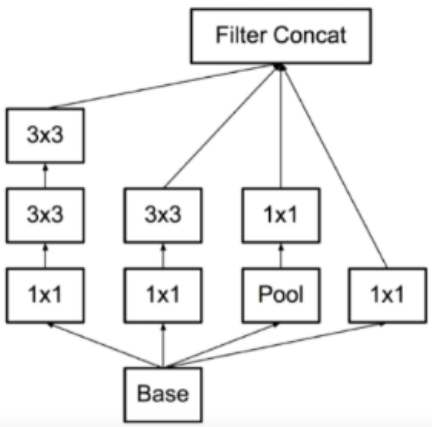


Fig. 5. 3 x Inception-A

(a) Inception-A

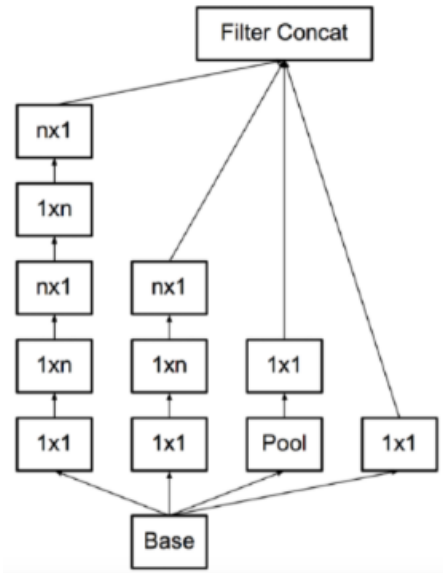


Fig. 6. 5 x Inception-B

(b) Inception-B

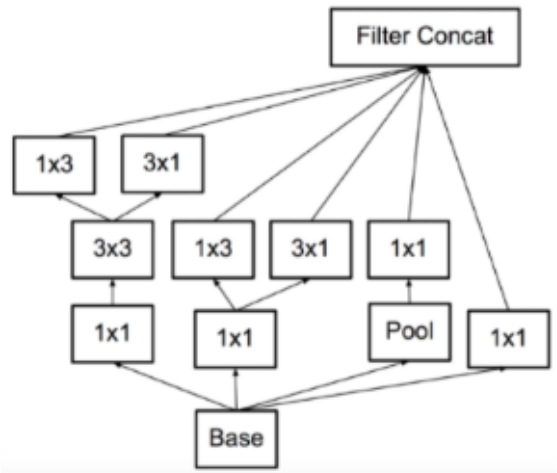


Fig. 7. 2 x Inception-C

(c) Inception-C

Figure 8: Inception Blocks in v2/v3

Inception-v2/v3

- As Inception-v2 changed to Inception-v3, the following contents were corrected :
  1. Momentum $\rightarrow$ RMSProp.
  2. Label Smoothing Regularization(LSR) : The main purpose of the LSR is to prevent the difference between the largest logit and others. (probability bias prevention)
  3. BN in auxiliary classifier : the fully connected layer of the auxiliary classifier is also batch-normalized, not just the convolutions.

Advanced Inception

- Inception-v4, Inception-ResNet-v1 and Inception-ResNet-v2 were published in the same paper[2].
- And Inception-v4, Inception-ResNet-v1, Inception-ResNet-v2 are an improved model of Inception-v3, Inception-v3(with ResNet), Inception-v4 respectively.
- As tensorflow[1] was developed in 2015, it became possible to develop Inception models by removing limitations.

Inception-v4

- Let's see the Inception-v4.
- Inception-v4 structure is as follows :

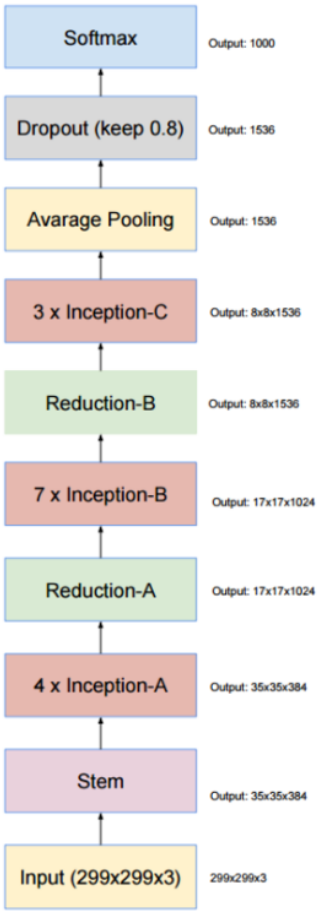


Figure 9: Inception-v4 Structure

Inception-v4

- Inception-v4 is a model in which the structure of Stem, Inception, and Reduction Block is slightly modified in Inception-v3.
- Furthermore, auxilary classifier part is all deleted in Inception-v4.
- Stem, Inception, Reduction Block are presented in the next slide.

Inception-v4

- Stem Block in Inception-v4 has the following form :

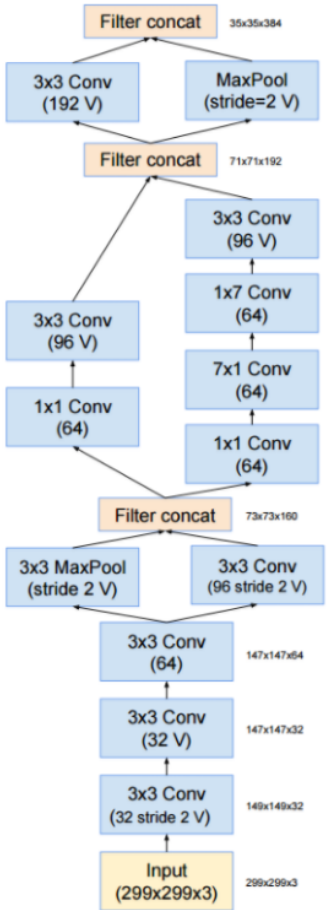
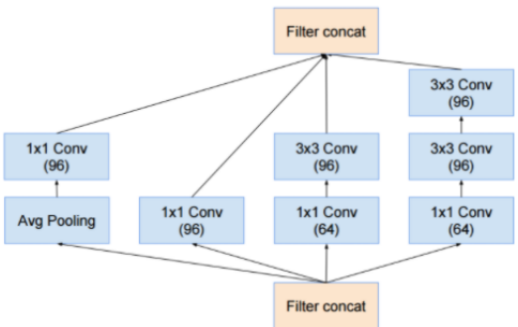


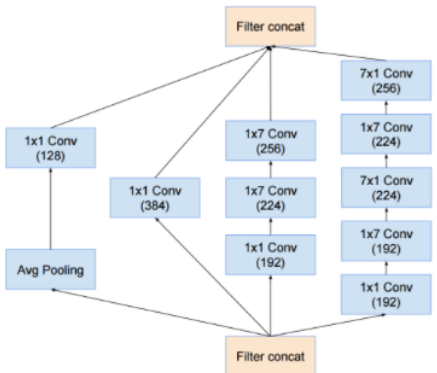
Figure 10: Stem Block

Inception-v4

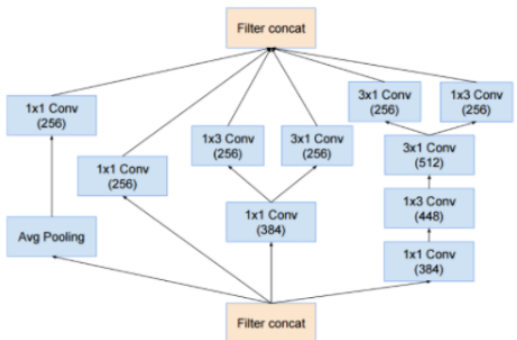
- Each Inception Block-A,B,C in Inception-v4 has the following form :



(a) Inception-A



(b) Inception-B

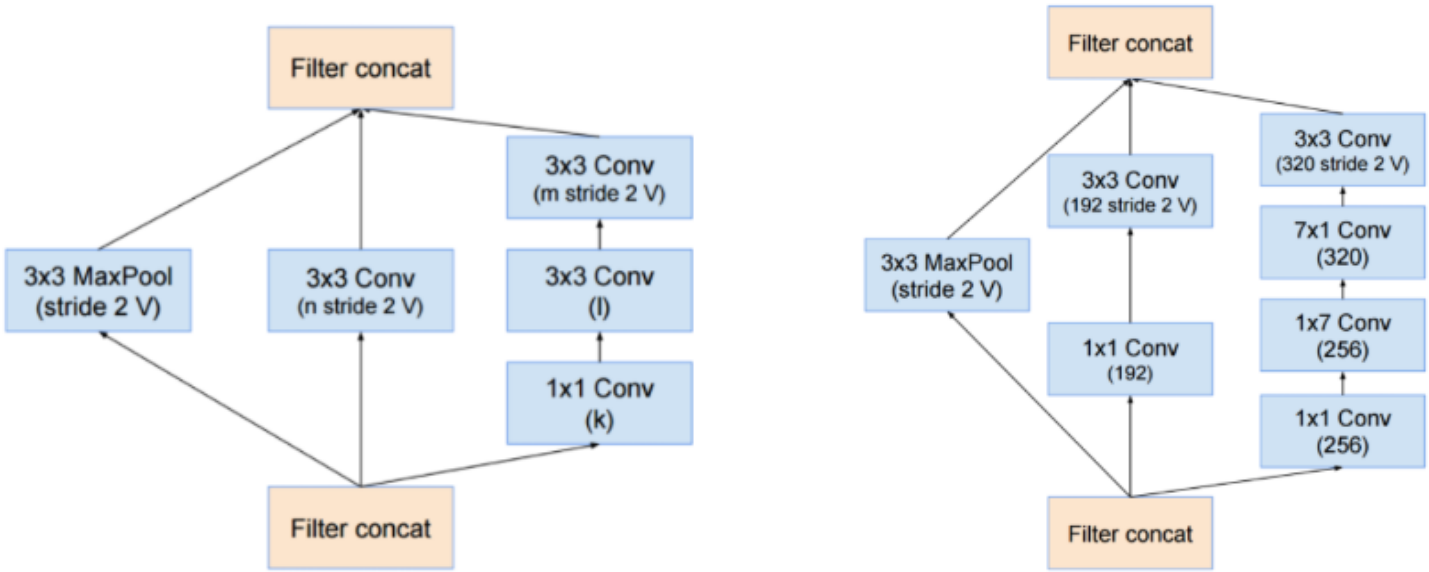


(c) Inception-C

Figure 11: Inception Blocks in v4

Inception-v4

- Each Reduction block-A,B in Inception-v4 has the following form :



(a) Reduction-A (b) Reduction-B

Figure 12: Reduction Blocks in v4

Inception-v4

- k, l, m, n corresponding to Reducion-A(Fig 12) follows the values in the picture below :

Network	k	l	m	n
Inception-v4	192	224	256	384
Inception-ResNet-v1	192	192	256	384
Inception-ResNet-v2	256	256	384	384

Figure 13: k,l,m,n in Reduction-A

Inception-ResNet-v1/v2

- Now we consider the model, Inception-ResNet-v1/v2.
- Inception-ResNet-v1/v2 has only partially changed in the structure of Inception-v3/v4.
- One particular thing is that scaling of the residuals was considered.

Inception-ResNet-v1

- Let's see the Inception-Resnet-v1.
- Inception-ResNet-v1 structure is as follows :

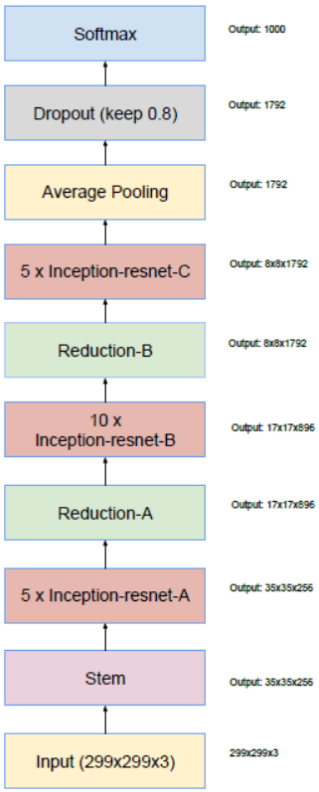


Figure 14: Inception-ResNet-v1 Structure

Inception-ResNet-v1

- Stem Block in Inception-ResNet-v1 has the following form :

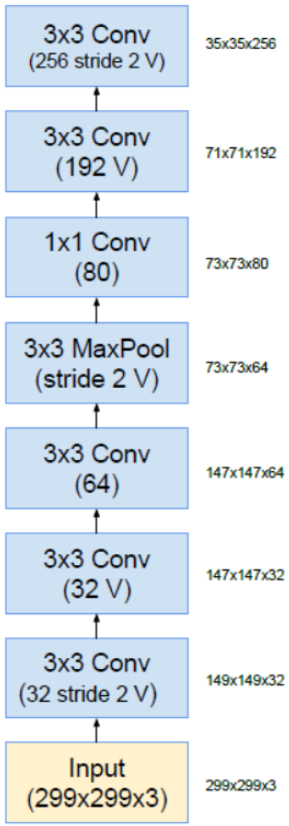
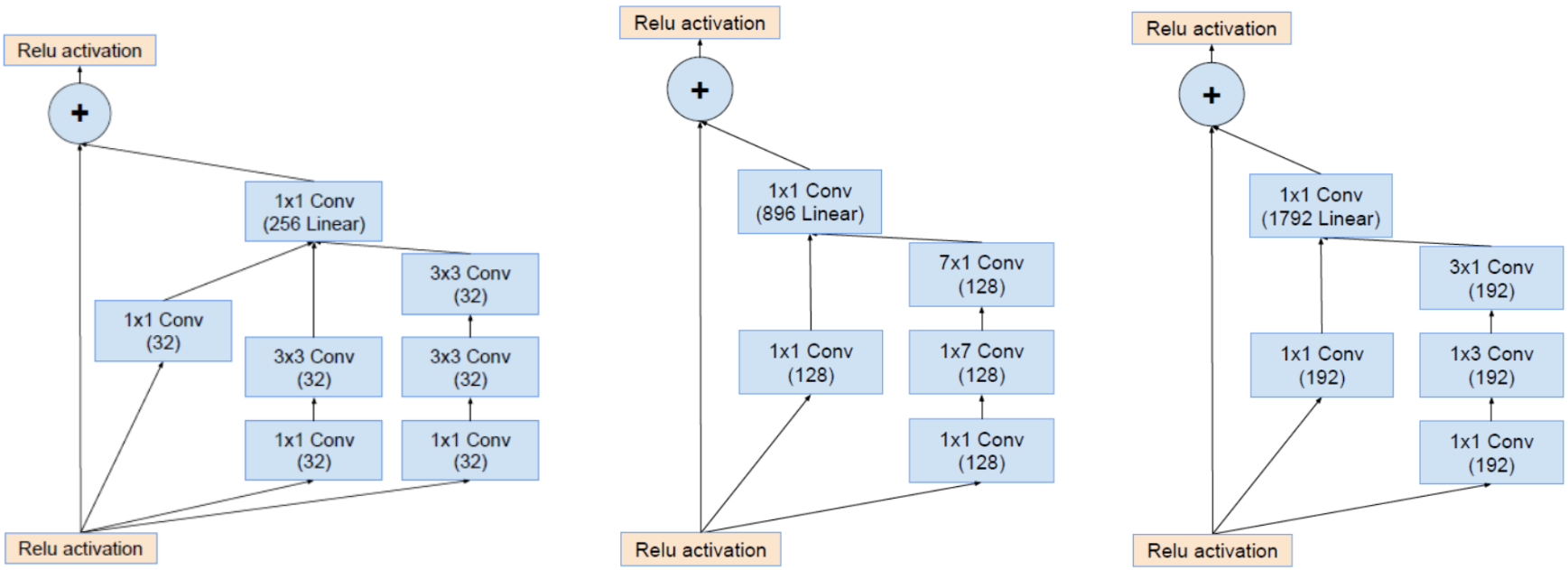


Figure 15: Stem Block

- It is same as Inception-v3.

Inception-ResNet-v1

- Each Inception Block-A,B,C in Inception-ResNet-v1 has the following form :

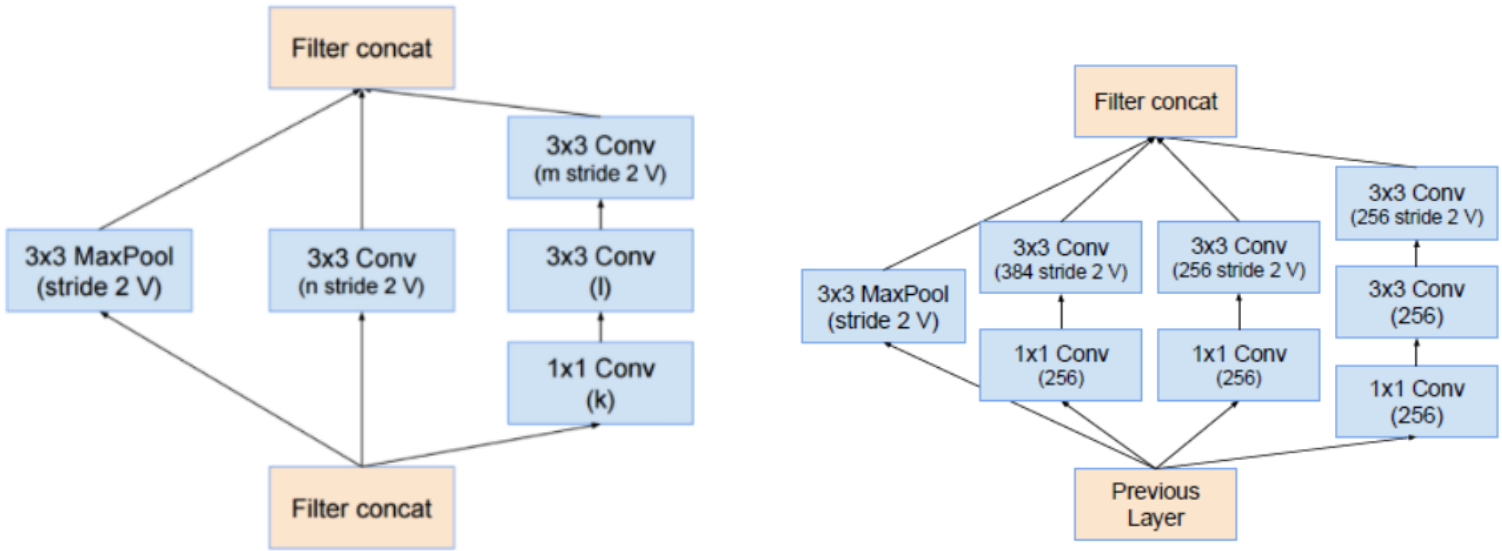


(a) Inception-A (b) Inception-B (c) Inception-C

Figure 16: Inception-ResNet Blocks in v1

Inception-ResNet-v1

- Each Reduction Block-A,B in Inception-ResNet-v1 has the following form :



(a) Reduction-A (b) Reduction-B

Figure 17: Reduction Blocks in Inception-ResNet-v1

- k,l,m,n in Reduction Block-A are in Fig 13.

Inception-ResNet-v2

- Let's see the Inception-Resnet-v2.
- Inception-ResNet-v2 structure is as follows :

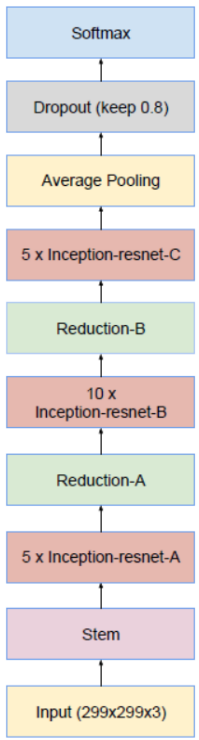


Figure 18: Inception-ResNet-v2 Structure

- This structure is same as Inception-ResNet-v1.

Inception-ResNet-v2

- Stem Block in Inception-ResNet-v2 has the following form :

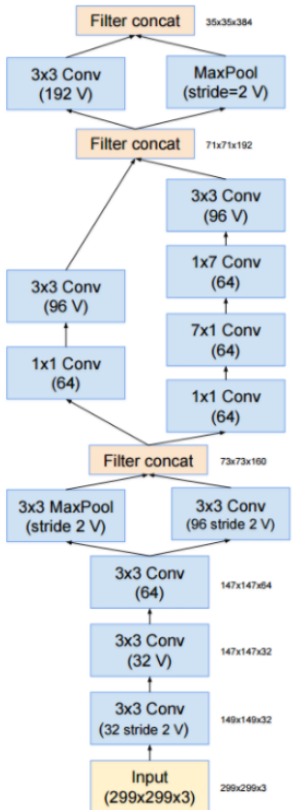
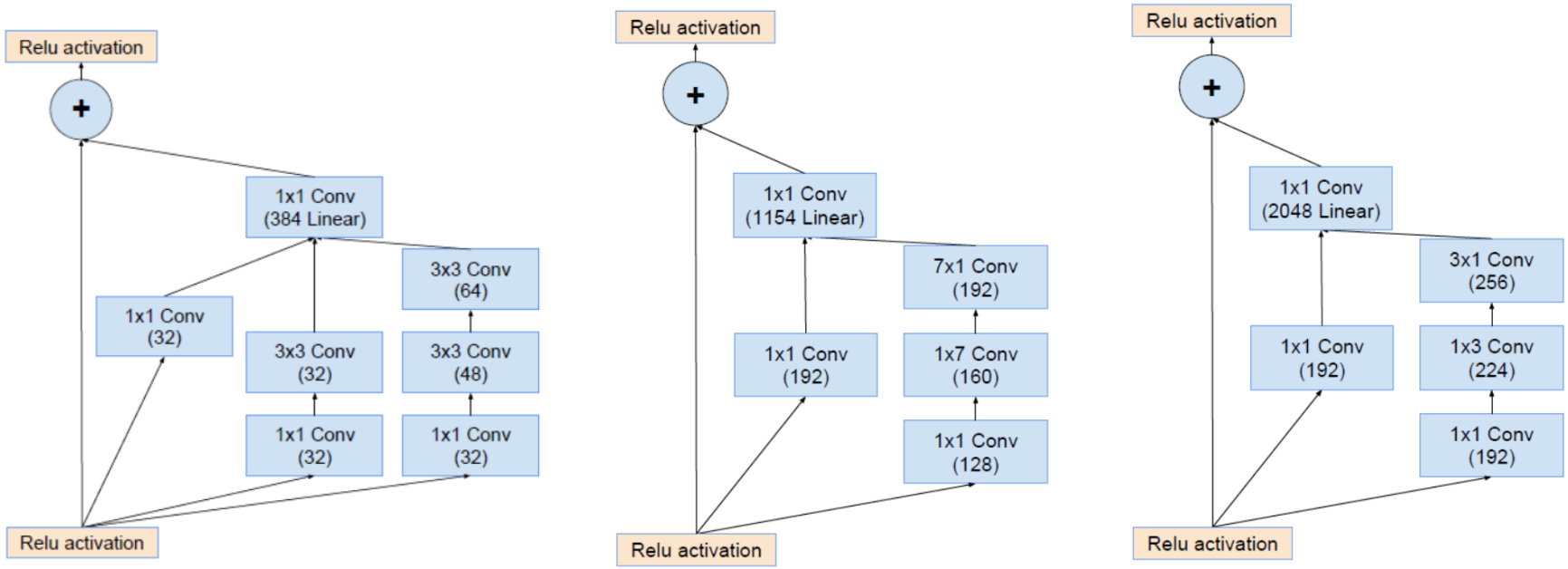


Figure 19: Stem Block

- It is same as Inception-v4.

Inception-ResNet-v2

- Each Inception Block-A,B,C in Inception-ResNet-v2 has the following form :

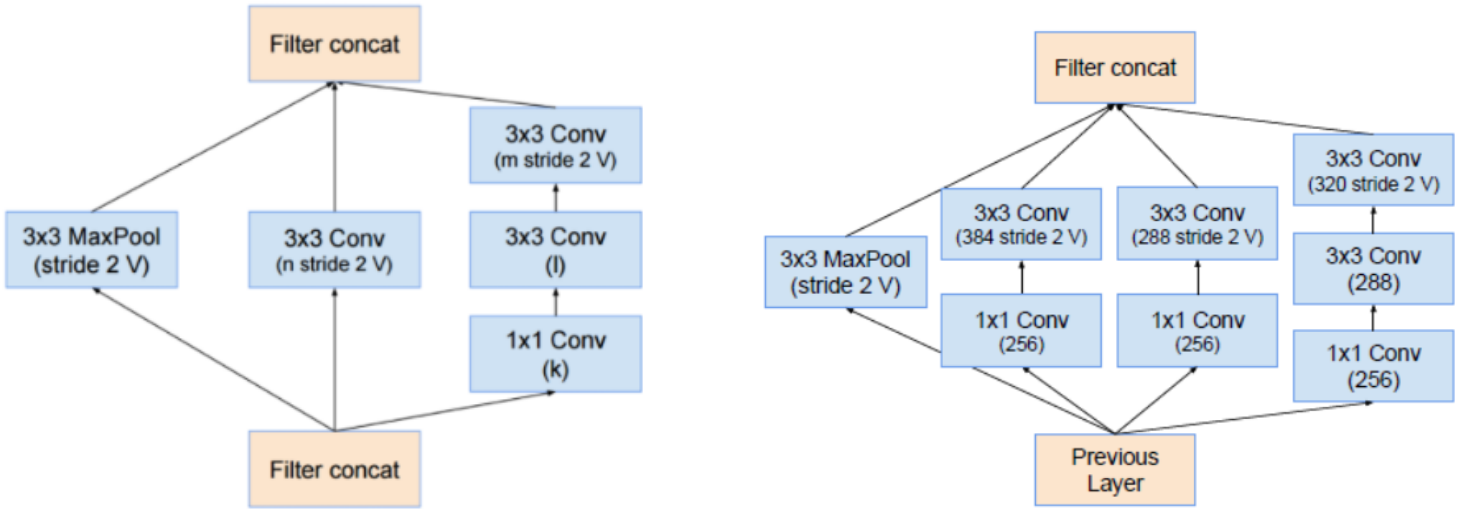


(a) Inception-A (b) Inception-B (c) Inception-C

Figure 20: Inception-ResNet Blocks in v2

Inception-ResNet-v2

- Each Reduction Block-A,B in Inception-ResNet-v2 has the following form :



(a) Reduction-A (b) Reduction-B

Figure 21: Reduction Blocks in Inception-ResNet-v2

- k,l,m,n in Reduction Block-A are in Fig 13.

Inception-ResNet-v1/v2 : Scaling of the Residuals

- And last, if the number of filters exceeded 1000, the residual variants started to exhibit instabilities and the network just "died" early in the training, meaning that the last layer before the average pooling started to produce only zeros after a few tens of thousands of iterations.
- This could not be prevented, either by lowering the learning rate, or by adding an extra batch-normalization to this layer.
- In general we picked some scaling factors between 0.1 and 0.3 to scale the residuals before their being added to the accumulated layer activations.
- Even where the scaling was not strictly necessary, it never seemed to harmed the final accuracy, but it helped to stabilize the training.

Inception-ResNet-v1/v2 : Scaling of the Residuals

- And this can be usually solved by giving a linear activation function.
- Note that only residual terms are assigned a scale between 0.1 and 0.3.

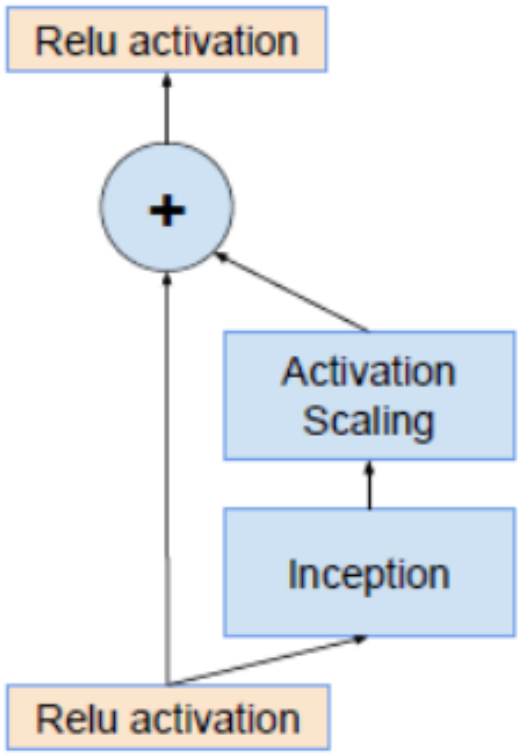


Figure 22: Scaling of the Residuals

Reference I

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*, 2016.

[2] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Thirty-first AAAI conference on artificial intelligence*, 2017.

[3] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.