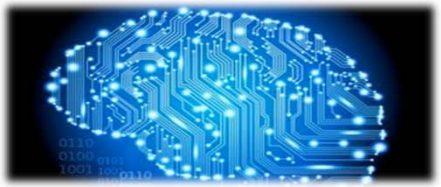
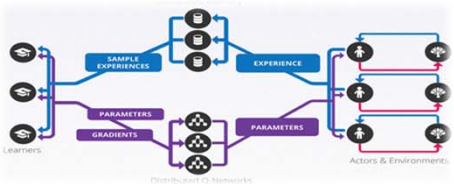



Inception V4 구현 및 실험 결과 보고

AutoML 연구실
석사과정 문아성
20.08.11

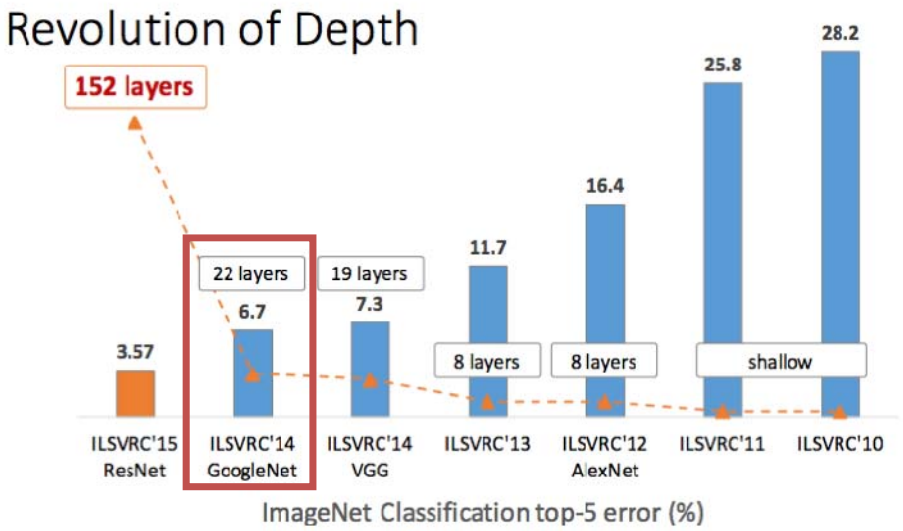
Introduction – "Inception"

Inception-v1

- 2014년 ImageNet Challenge (ILSVRC) 에서 우승 (GoogLeNet)
- 기존 방법에 비해 훨씬 적은 양의 파라미터 수로 소개

(2012. Alexnet: 63M, 2014. VGG: 138M, 2014. GoogLeNet: 7M)

(M: Million)



"Inception"

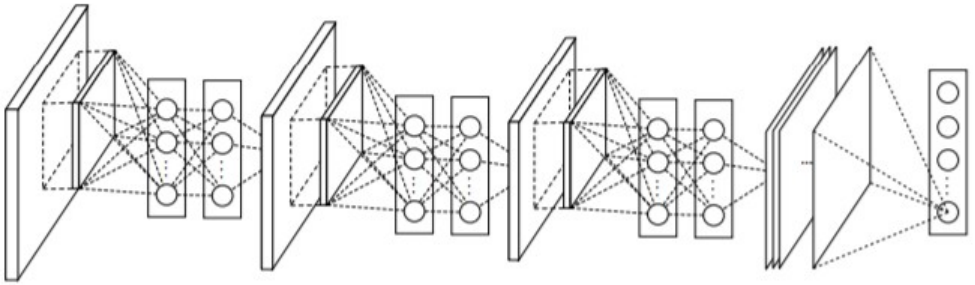
- 딥러닝은 망이 깊고 레이어가 넓을수록 성능이 좋음
- 하지만 실제로는 overfitting, vanishing 문제가 발생

주안점: 깊어질수록 손실이 큰데,
이것을 어떻게 없애면서 깊게 들어갈지가 핵심

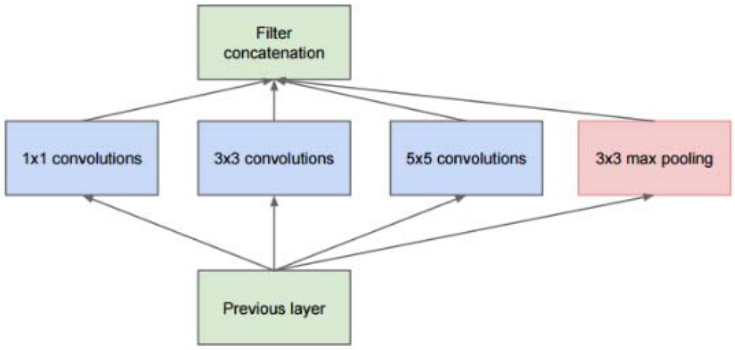
- "Inception" 영화 장면 중에서 -

Inception-v1 Module

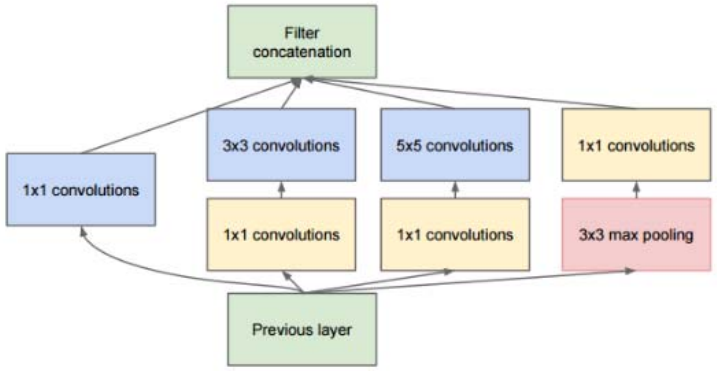
- 전체적으로 망내 연결은 줄이면서, 세부적인 행렬 연산은 최대로 Dense 연산을 하도록 처리 하는 것이 목표



- 실제 네트워크의 구조 -



(a) Inception module, naïve version



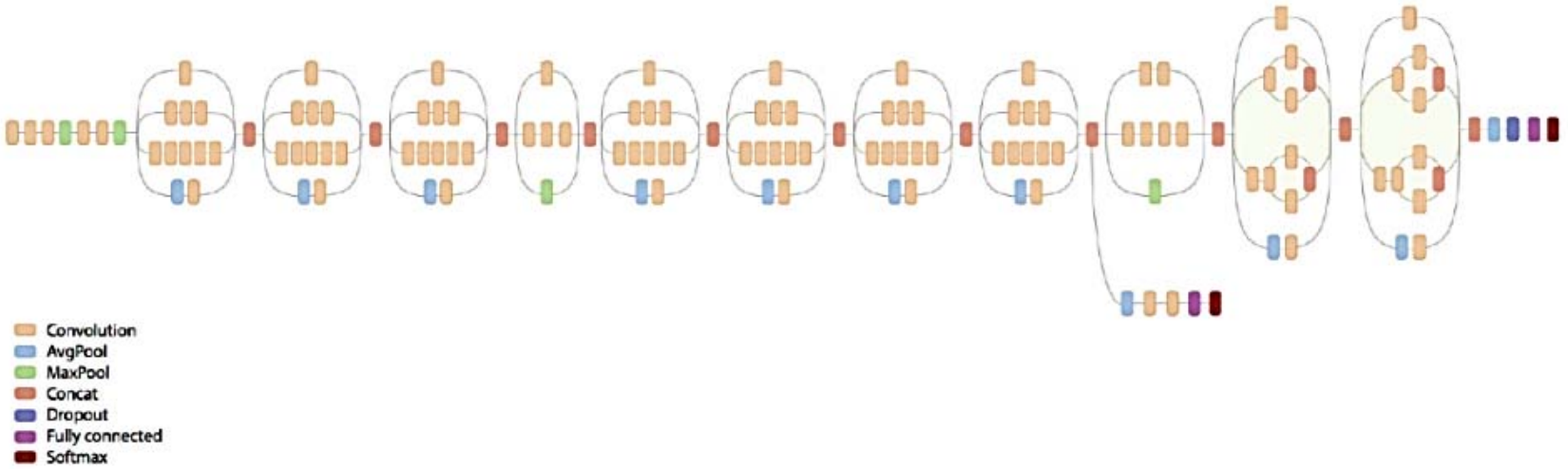
(b) Inception module with dimension reductions

- 1 x 1 Convolution
  - 최적화 과정에서 채널 간의 특징을 추출
  - 채널을 줄인 후 3 x 3, 5 x 5 convolution 을 하면 파라미터 개수도 절약 가능

Inception-v2 Module - ①

Inception-v2 주요 특징

- Convolution Filter Factorization
- Auxiliary Classifier
- Avoid Representational Bottleneck



Inception V2 & V3 Model

Inception-v2 Module - ②

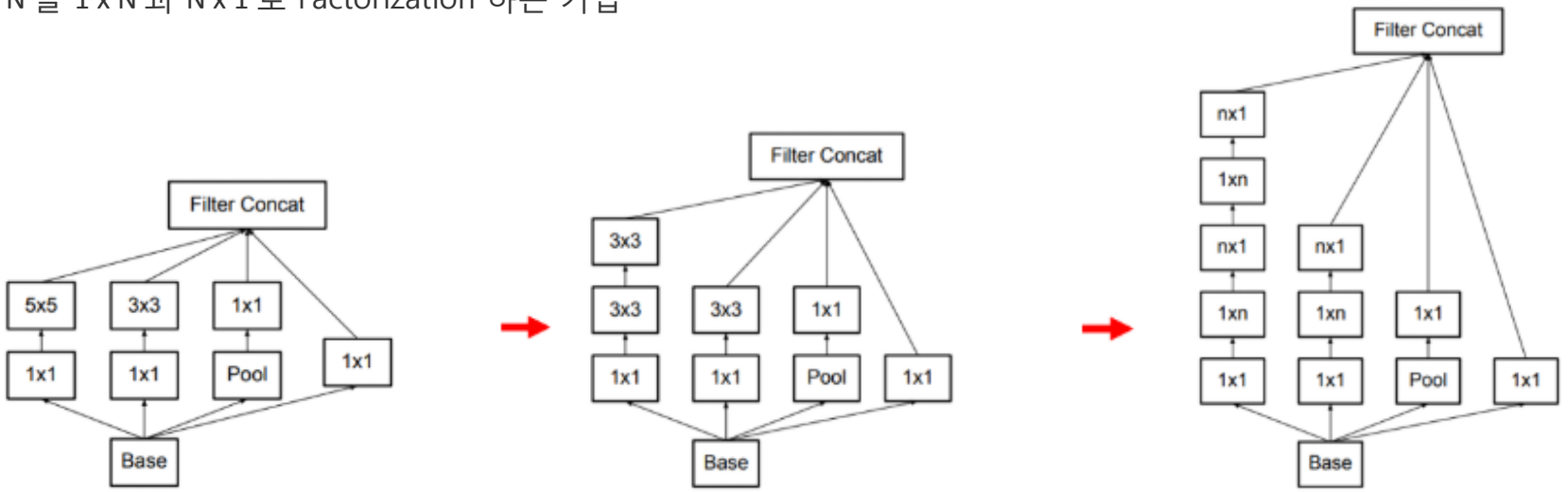
Convolution Filter Factorization

- 5 x 5 conv를 3 x 3 conv 2개로 대체하는 방법을 적용

  5x5xN : (3x3xN) + (3x3xN) = 25 : 9+9 = 25 : 18 (약 28% 의 reduction 효과)

- 더 나아가 비대칭(Asymmetric) Conv 방법을 적용

  N x N 을 1 x N 과 N x 1 로 Factorization 하는 기법



Inception V1 Model

5 x 5 Conv 를 3 x 3 Conv 2개로 변경

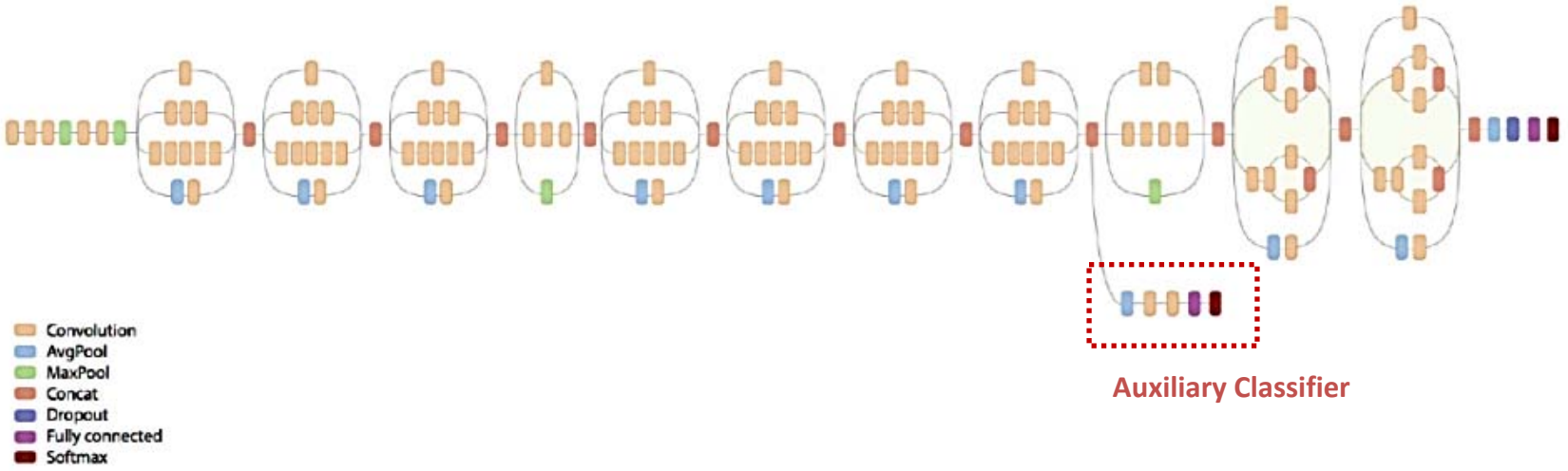
N x N 을 1 x N 과 N x 1 로 변경

Inception-v2 Module - ③

Auxiliary Classifier

- 깊은 망을 구성하기 때문에 모델 중간에 예측하는 부분을 추가하여
  Loss 값을 구해놓음으로써, Backpropergation 할 때 gradient를 적절하게 조절하는 역할

- Vanishing Gradient Problem 을 방지



Inception V2 & V3 Model

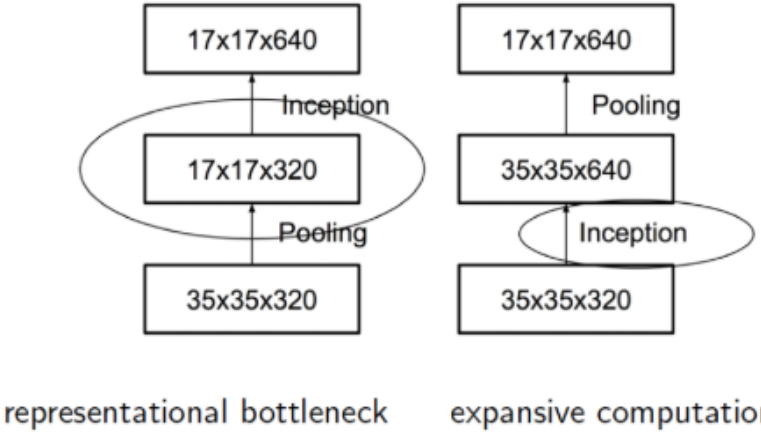
Inception-v2 Module - ④

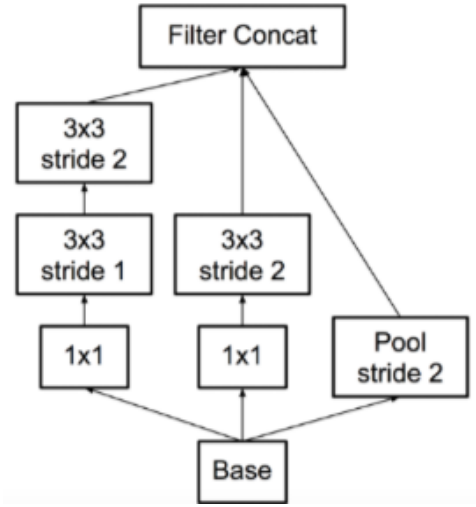
Avoid Representational Bottleneck

- Representational Bottleneck ?

  Pooling 을 먼저하고 Conv 연산을 하였을 때, Pooling을 먼저 함으로써 정보 손실이 발생하는 것을 의미

- Representational Bottleneck 이 발생하지 않으면서도, 적은 연산량으로 해결할 수 있는 구조를 마련



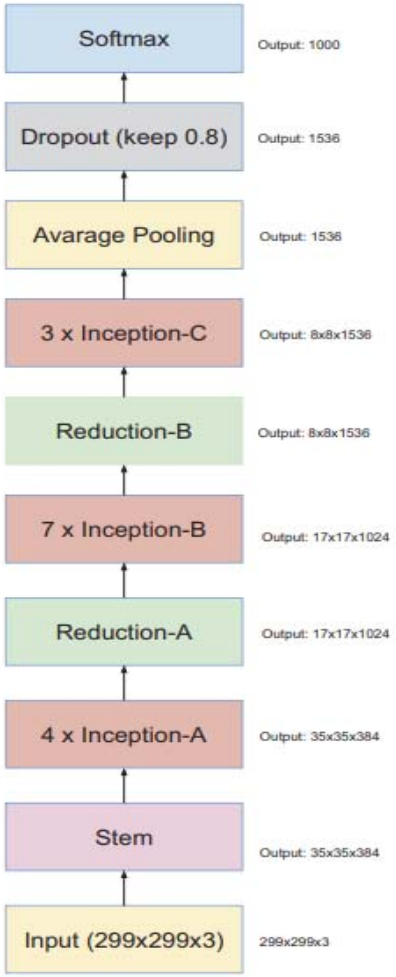


Reduction

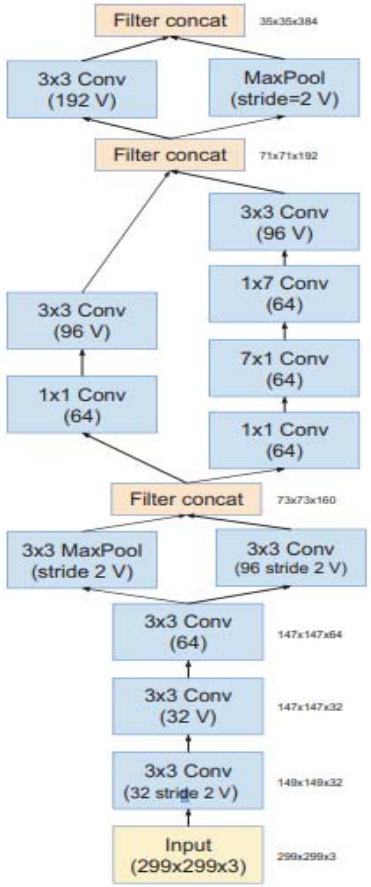
Inception-v4 Module - ①

Module 구성

- Input과 바로 연결되는 Stem block
- 3개의 Inception Block(Inception-A, Inception-B, Inception-C)
- feature map의 size가 절반으로 줄어드는
  2개의 Reduction Block(Reduction-A, Reduction-B)



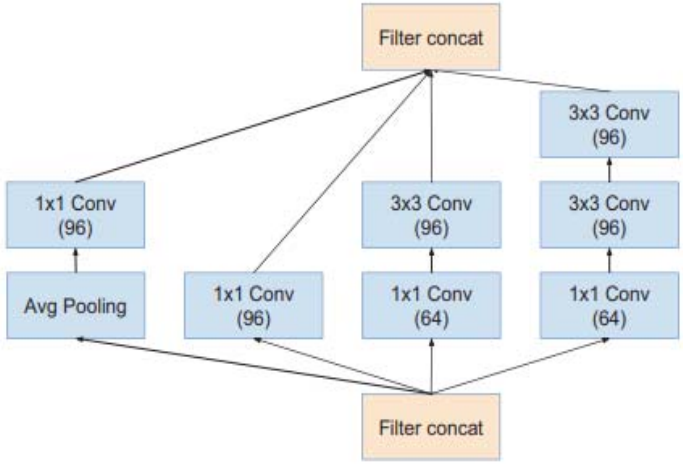
[Inception-v4 architecture]



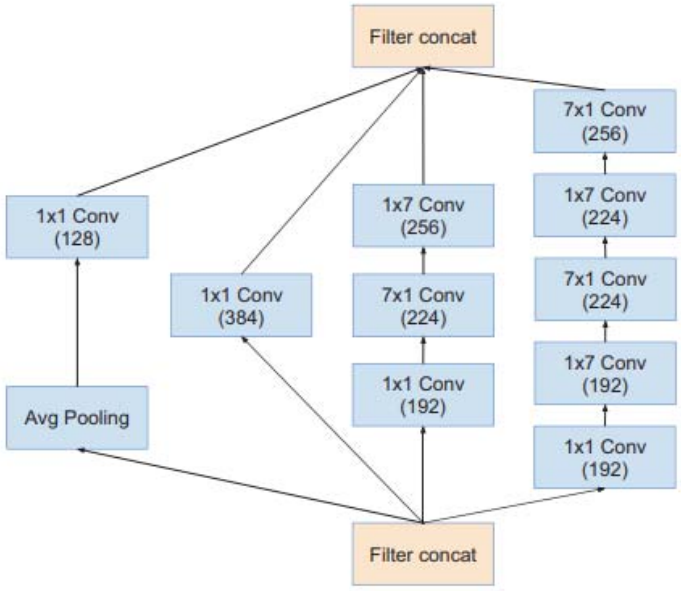
[Stem]

Inception-v4 Module - ②

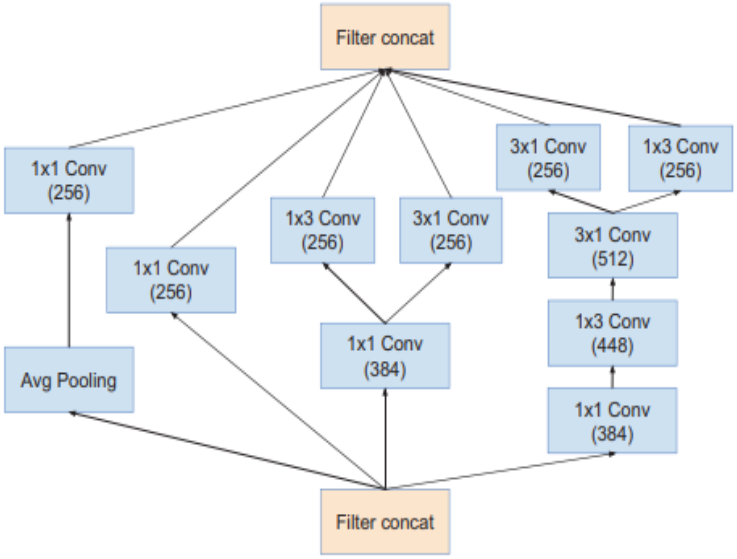
Inception – A, B, C



[Inception-A] (35 x 35)



[Inception-B] (17 x 17)

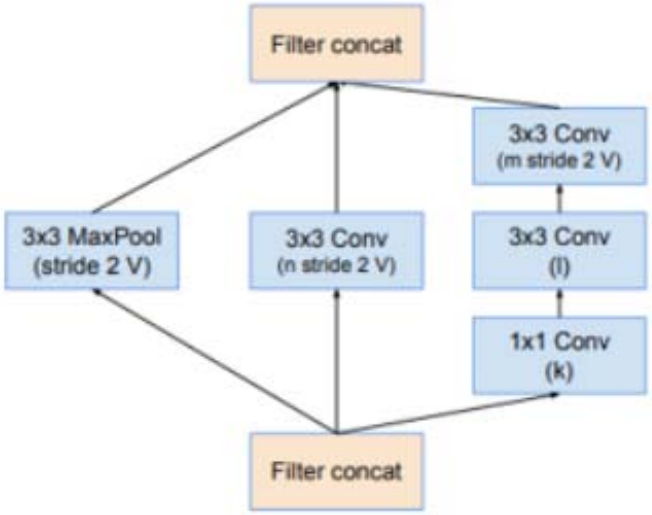


[Inception-C] (8 x 8)

Inception-v4 Module - ③

Reduction – A

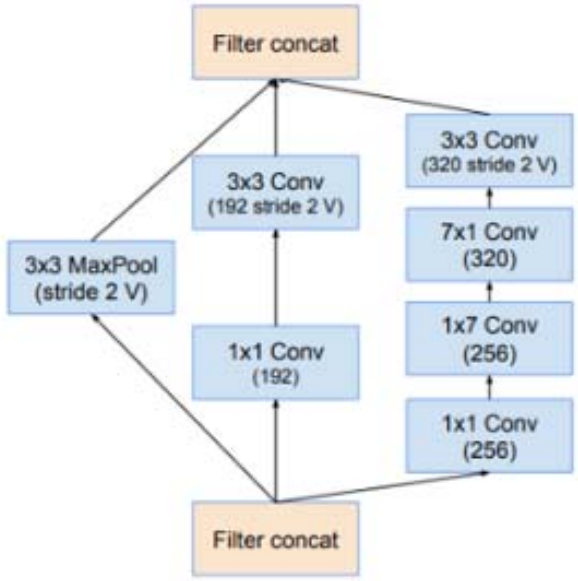
Convert (35 x 35) to (17 x 17)



[Reduction-A]

Reduction – B

Convert (17 x 17) to (8 x 8)



[Reduction-B]

DATASET CIFAR-10

32x32사이즈의 RGB 이미지와

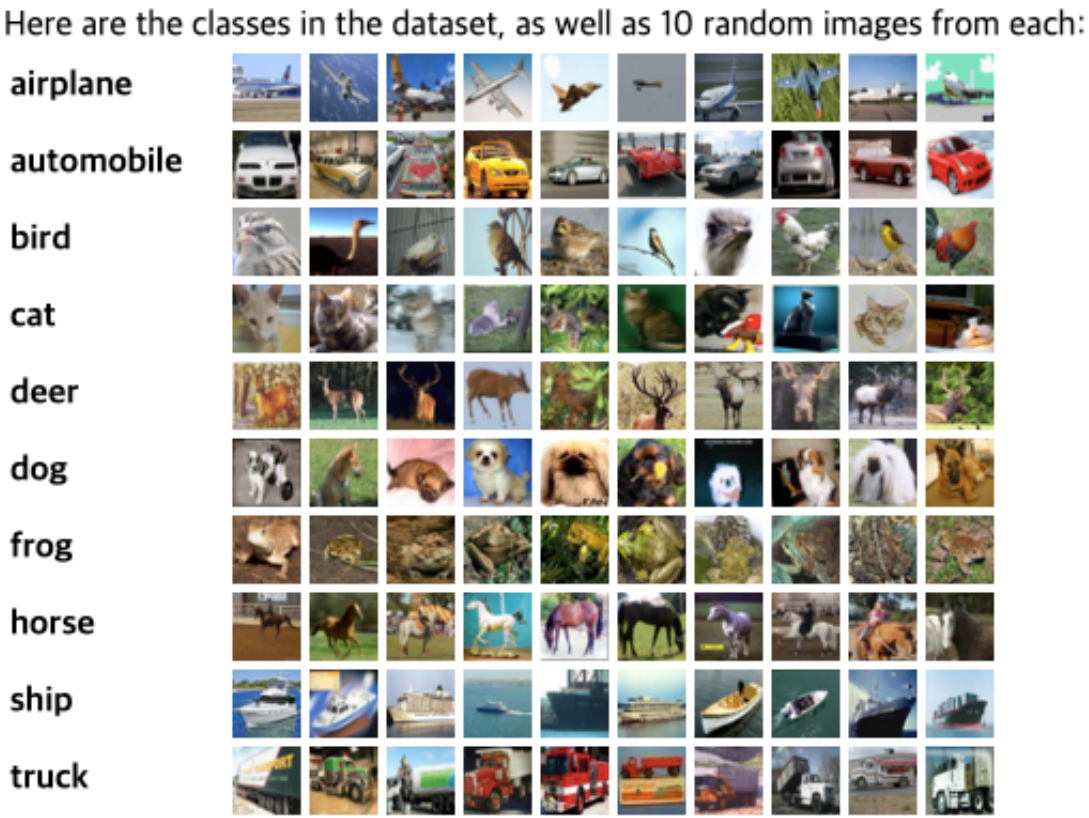
10가지의 레이블로 구성 $((32, 32, 3),\ (10))$

Inception-v4 Input Size가 (299, 299, 3) 이므로

CIFAR-10 의 데이터를 (299, 299, 3) 으로 RESIZE

INPUT DATA : 50000

TEST DATA : 10000



실험 결과

		실제 정답	
		True	False
분류 결과	True	True Positive	False Positive
	False	False Negative	True Negative

<Confusion Matrix>

정밀도(Precision): 88.06%

재현율(Recall): 87.90%

정확도(Accuracy): 87.95%

정밀도(Precision): $\frac{TP}{TP+FP}$

재현율(Recall): $\frac{TP}{TP+FN}$

정확도(Accuracy): $\frac{TP+TN}{TP+FP+FN+TN}$