5. Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. 2016

Abstract

- Image Recognition Task에서 CNN은 큰 발전을 하였으며 Residual Connection과 적통적인 Architecture와의 결합은 SOTA 성능을 냄
- Residual Connection을 쓴 모델과 쓰지 않은 모델을 모두 소개함
- 적절한 Activation Scaling을 설정하는것은 안정적인 학습을 가능하게 했다는 사실을 발견함

Introduction

- Residual Connection과 Inception 두 가지 연구를 결합하여 모델을 만듬
- Inception의 구조는 매우 깊기 때문에 Inception의 Filter Concatenation Stage를 Residual Connection 구조로 대체
    - Inception-ResNet의 구조
        - 잔차 연결을 사용함으로써 Inception-ResNet은 기존 Inception 모델보다 계산 효율성을 향상시킬 수 있음
        - 잔차 연결은 학습 중 그래디언트가 네트워크를 더 잘 통과하게 함으로써 학습 과정이 안정화되고, 이는 전체적인 학습 시간을 단축 시킬 수 있는 결과를 가져옴
- Inception-v4 부터는 TensorFlow로 학습을 진행하여 구조를 크게 단순화함

Realted Work

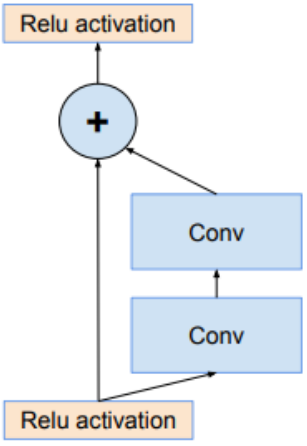


Figure 1. Residual connections as introduced in He et al. [5].

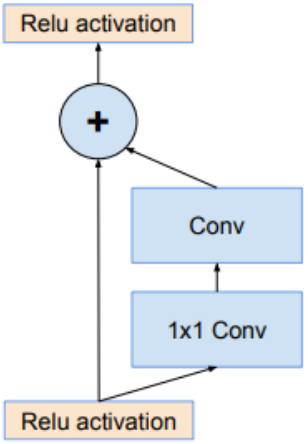
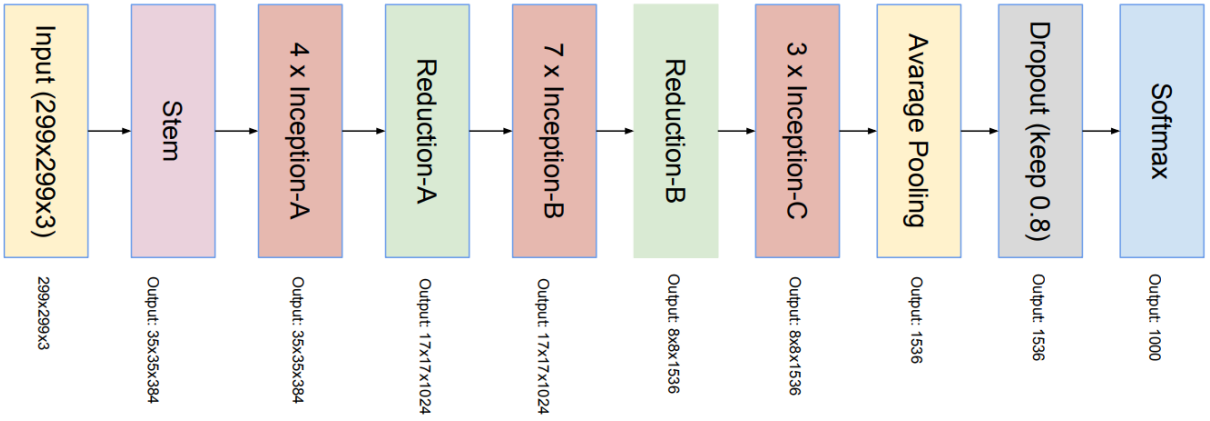


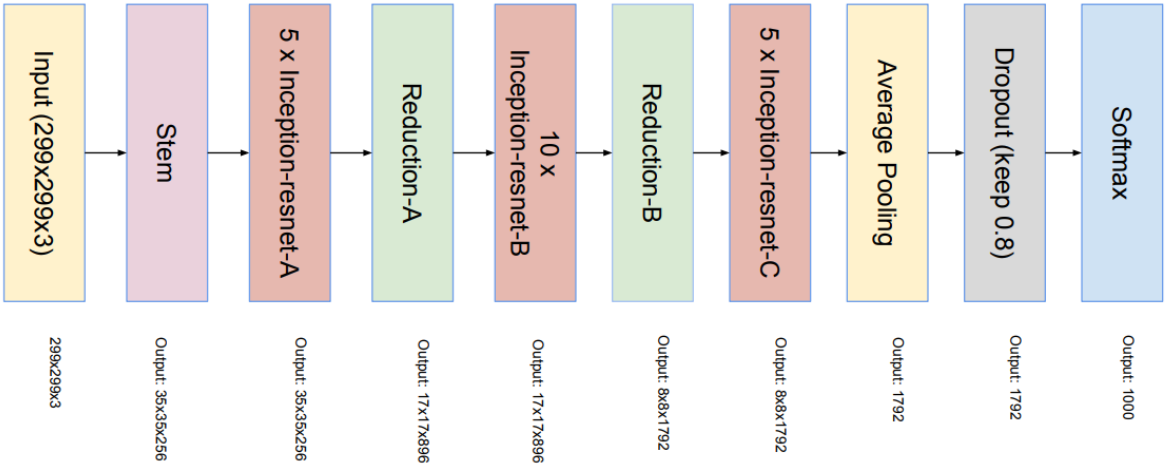
Figure 2. Optimized version of ResNet connections by [5] to shield computation.

- 좌측 그림은 Residual Connection의 기본적인 형태
- 우측 그림은 1 x 1 Convolution을 사용하여 연산량을 줄인 형태의 Residual Connection
- Residual Connection을 사용할 때 학습 속도가 크게 빨라져서 Inception구조와 결합

Architectural Choices



- Inception v4의 Architecture

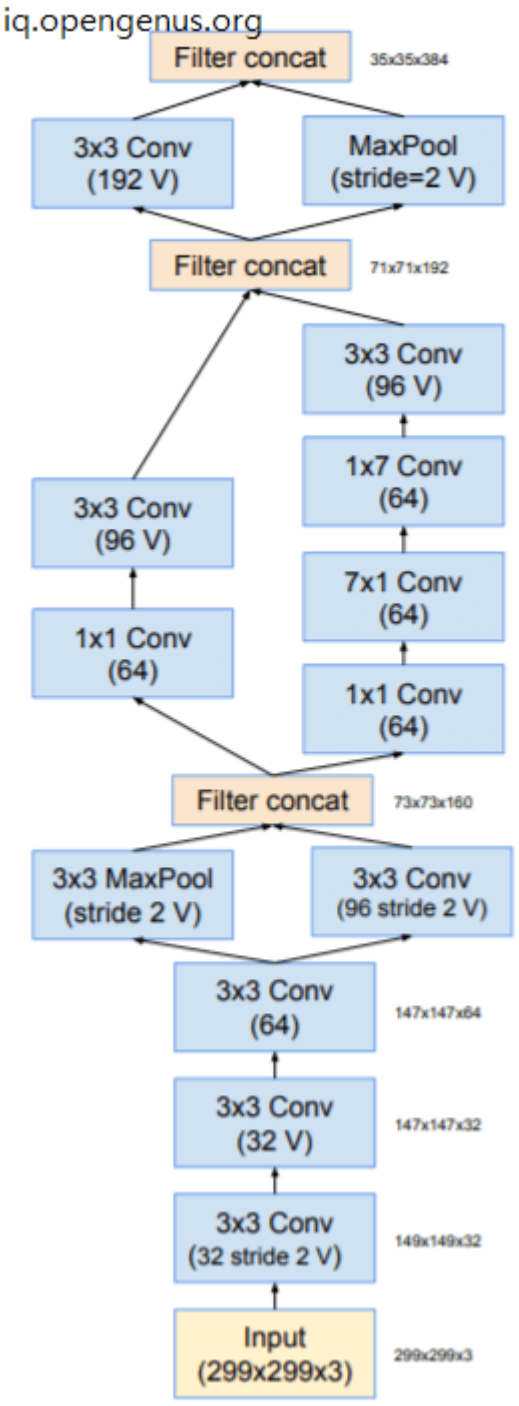


- Inception-resnet-v2의 Architecture

- Pure Inception Blocks
  - 이전의 Inception 모델은 메모리 제약 문제로 인해 전체 모델의 Replica를 여러 하위 네트워크로 분할하는 방식으로 학습했지만, TesorFlow의 도입으로 Replica분할 없이 학습이 가능
  - Inception Architecture를 변경하는 것에 대해 보수적인 측면이었지만, 네트워크를 단순화하기 위해서 Inception-v에서는 불필요한 부분을 과감하게 제거하여 하나의 Inception Block에서는 한가지 크기의 Grid Size를 사용하도록 함
  - 아래 나올 이미지에서 V로 표시된 부분은 Valid Padding이 적용되었고, V가 없는 부분은 Same Padding이 적용되었음
    - V표시된 부분은 Output Activation Map의 Grid Size가 감소하고 V표시가 없는 부분은 Input과 Output의 Grid Size가 동일함
    - Valid Padding
      - Padding을 추가하지 않은 형태, Valid Padding을 적용하고 필터를 통과시키면 입력사이즈보다 작게 나오게 됨
    - Same Padding
      - Padding을 추가하여 출력 크기를 입력 크기와 동일하게 유지

- Residual Inception Blocks
  - Residual Version Network에서는 Inception-v4보다 연산량이 적은 Inception Block을 사용

- 각 Inception Block에는 Activation이 없는 1x1 Convolution Layer가 Filter-expansion Layer로 뒤따르는데 이는 Inception Block에서 감소한 차원을 다시 확장해주기 위함
- Inception-ResNet-v1의 경우는 Inception-v3의 계산 비용과 비슷하고, Inception-ResNet-v2의 경우는 Inception-v4의 계산 비용과 비슷함
- Inception-ResNet은 기존 Layer에만 Batch-Norm을 사용하고 Layer가 합쳐지는 부분에서는 사용하지 않았으며, Batch-Norm을 모든 Layer에서 사용하는 것이 좋지만 단일 GPU에서 학습을 하기 위함
- Inception-ResNet-v1에서는 독자적인 STEM을 사용하고, Inception-ResNet-v2는 Inception-v4와 같은 STEM을 사용



    - Inception-v4와 Inception-ResNet-v2 모델 앞단에 들어가는 STEM 구조
    - Inception-v3 모델에도 존재했던 앞단의 Convolution 부분을 더 성능이 좋도록 수정함
    - Grid Size를 줄이는 부분에서 병렬처리 확인이 가능

- Inception-ResNet-A

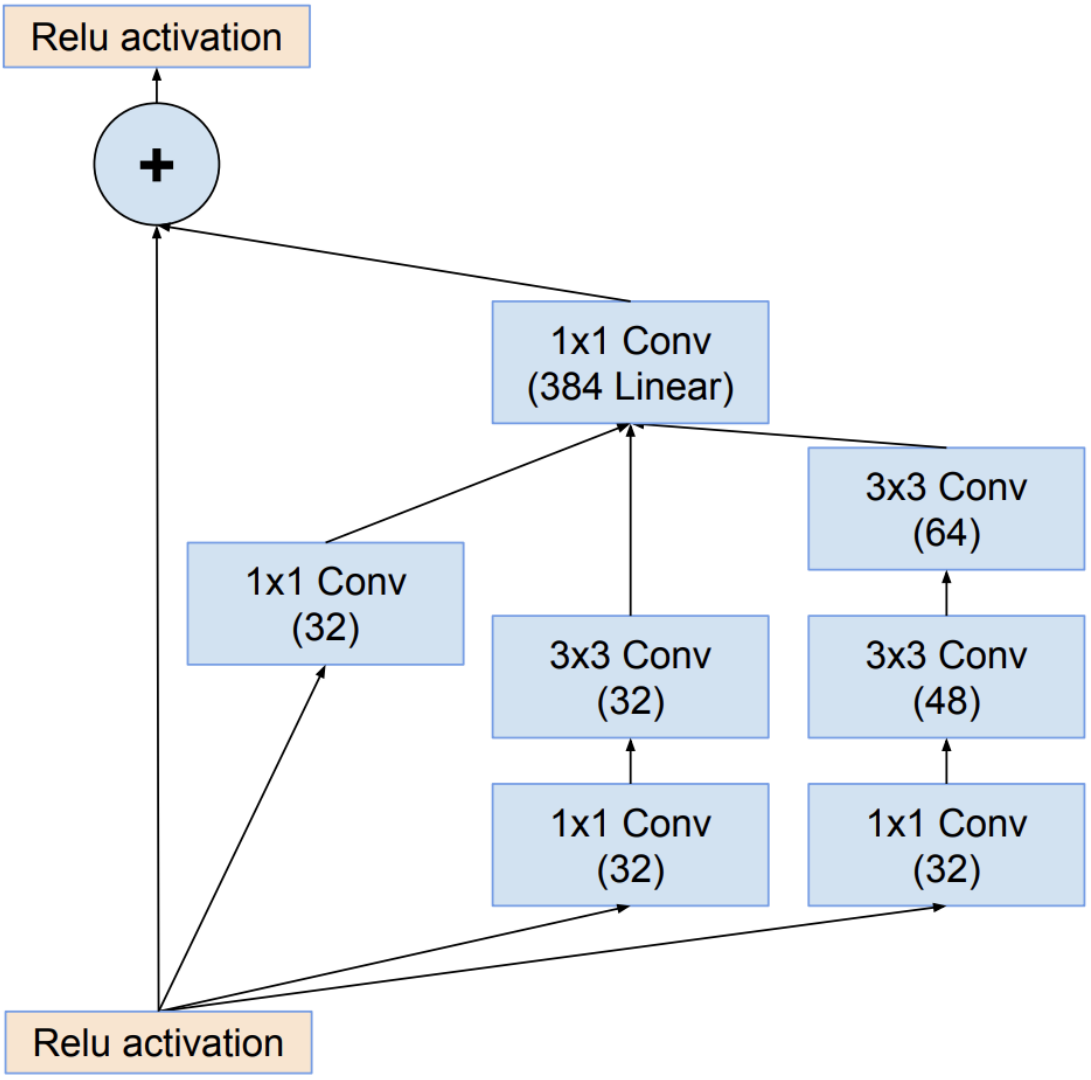


Figure 16. The schema for 35×35 grid (Inception-ResNet-A) module of the Inception-ResNet-v2 network.

- Inception-ResNet-B

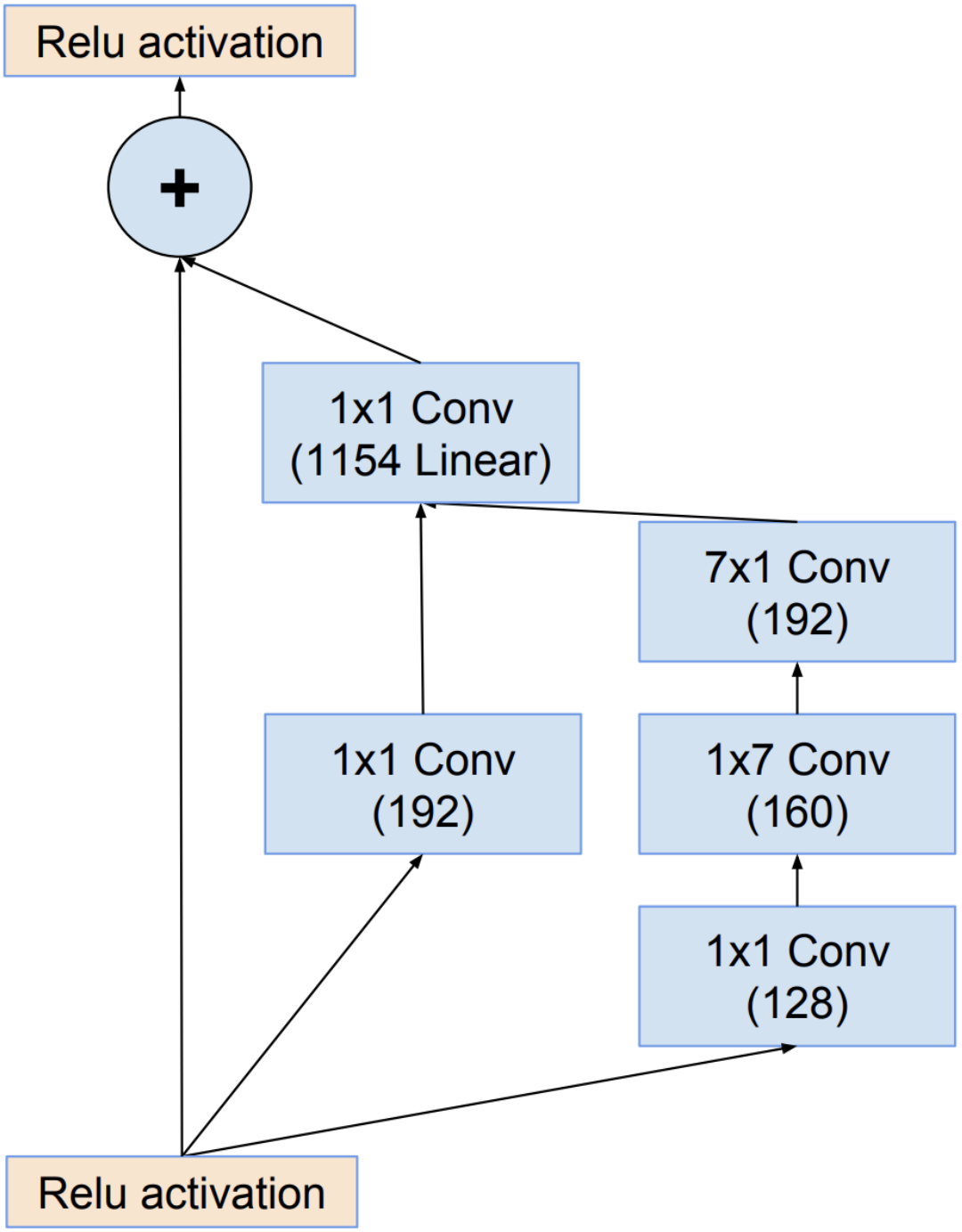


Figure 17. The schema for 17 × 17 grid (Inception-ResNet-B) module of the Inception-ResNet-v2 network.

- Inception-ResNet-C

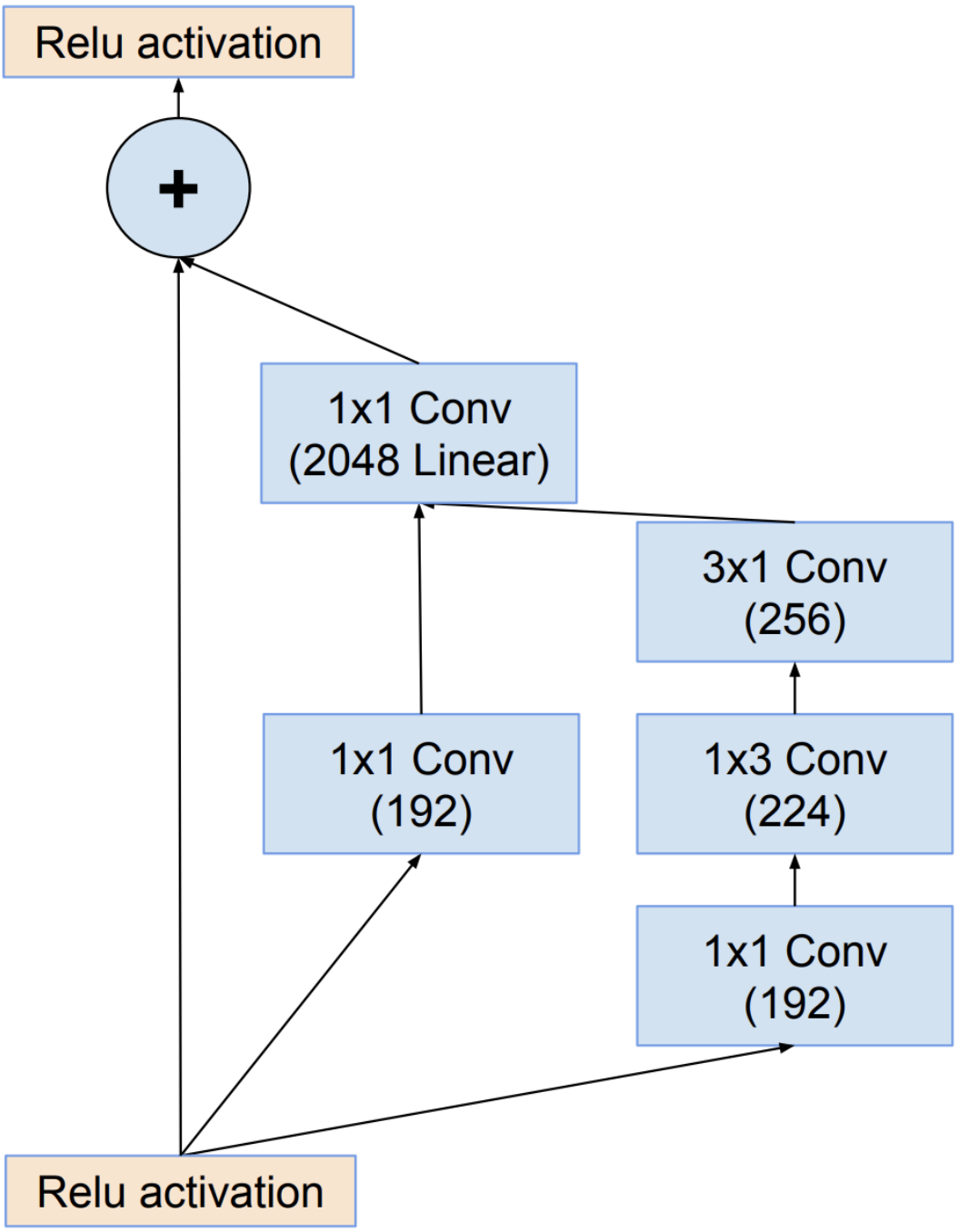


Figure 19. The schema for 8×8 grid (Inception-ResNet-C) module of the Inception-ResNet-v2 network.

- Reduction-A

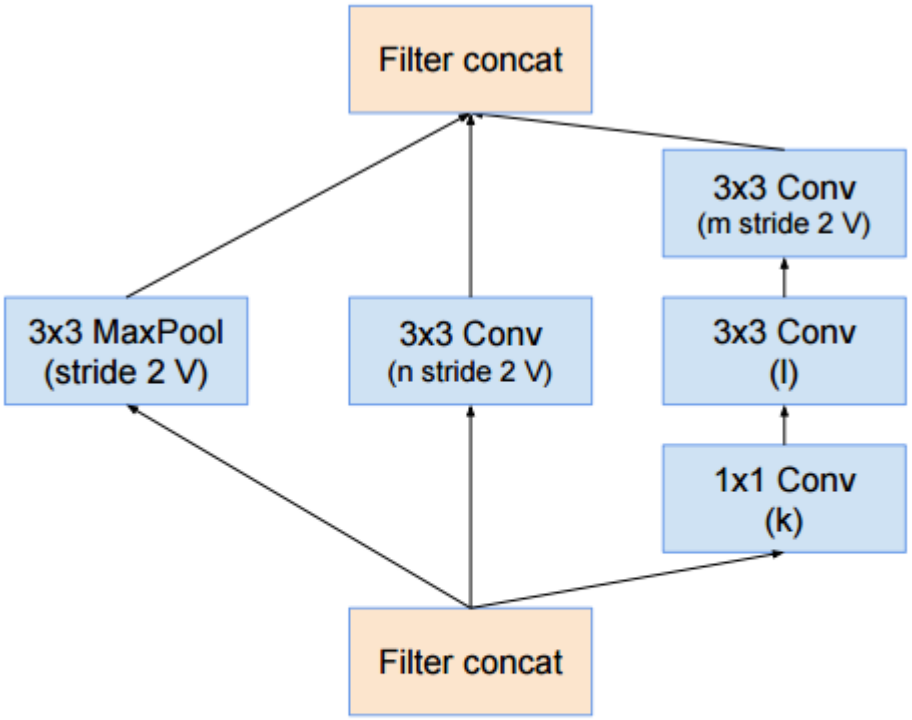


Figure 7. The schema for 35×35 to 17×17 reduction module. Different variants of this blocks (with various number of filters) are used in Figure 9, and 15 in each of the new Inception(-v4, -ResNet-v1, -ResNet-v2) variants presented in this paper. The k, l, m, n numbers represent filter bank sizes which can be looked up in Table 1.

- Reduction-B

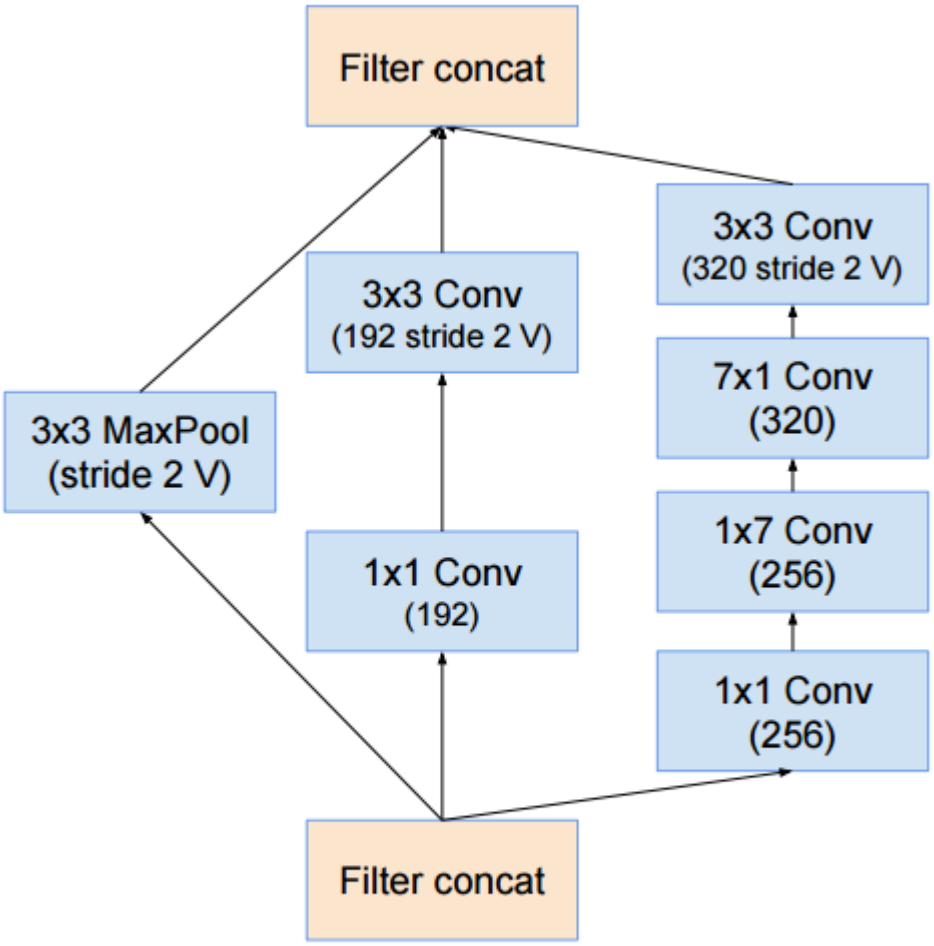


Figure 8. The schema for 17×17 to 8×8 grid-reduction module. This is the reduction module used by the pure Inception-v4 network in Figure 9.

- Reduction 계층에서는 Grid Size가 절반으로 줄어듬
  - Inception-v4, Inception-ResNet-v1, Inception-v2는 모두 동일한 Reduction-A구조를 사용함

- Scaling of the Residuals
  - Filter의 수가 1000개를 초과할 때 Residual Variants가 불안정해지고, 학습 도중에 네트워크가 죽어버리는 문제가 발생
  - 수만번의 Iteration이 지나면 Average Pooling 이전에 0 값만을 반환하기 때문인데, 본 논문에서는 활성화 함수를 적용하기 이전에 잔차를 Scaling Down하여 학습과정을 완화함

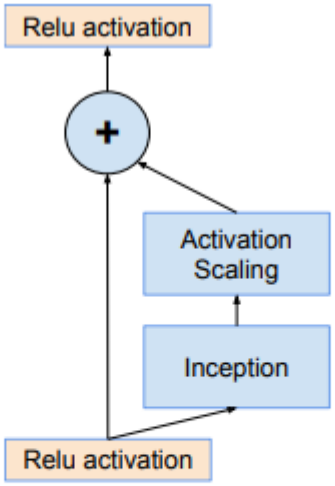


Figure 20. The general schema for scaling combined Inception-resnet moduels. We expect that the same idea is useful in the general resnet case, where instead of the Inception block an arbitrary subnetwork is used. The scaling block just scales the last linear activations by a suitable constant, typically around 0.1.

- 2단계로 학습이 진행되며 0.1 ~ 0.3 사이의 Scaling Factor를 사용함

- Results
  - Inception-v3와 Inception-Resnet-v1 학습 곡선 비교
    - 둘의 연산량이 비슷하기 때문에 비교

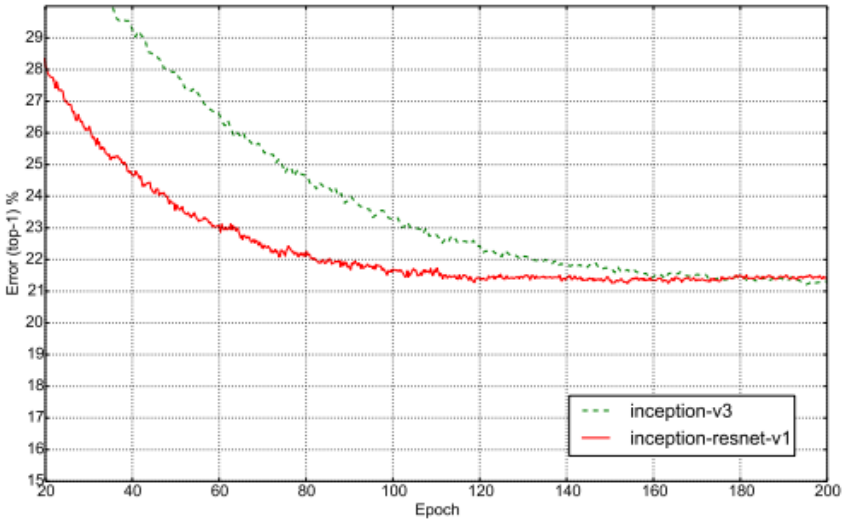


Figure 21. Top-1 error evolution during training of pure Inception-v3 vs a residual network of similar computational cost. The evaluation is measured on a single crop on the non-blacklist images of the ILSVRC-2012 validation set. The residual model was training much faster, but reached slightly worse final accuracy than the traditional Inception-v3.

- Inception-v4와 Inception-Resnet-v2 학습 곡선 비교
  - 둘의 연산량이 비슷하기 때문에 비교

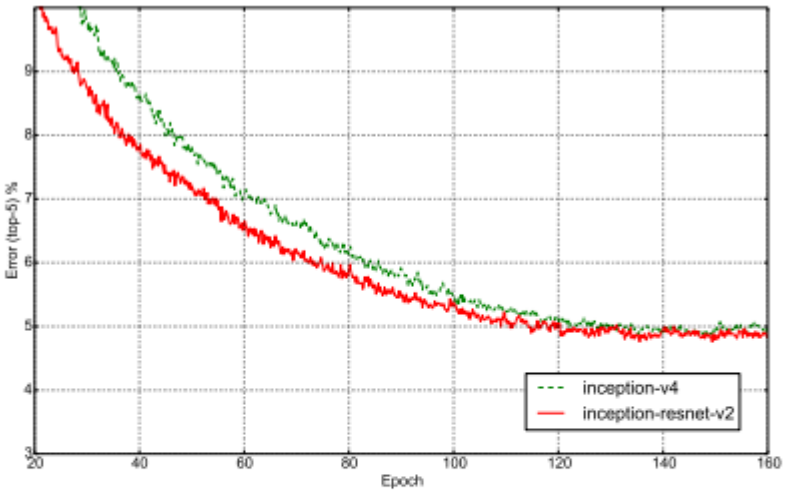


Figure 24. Top-5 error evolution during training of pure Inception-v4 vs a residual Inception of similar computational cost. The evaluation is measured on a single crop on the non-blacklist images of the ILSVRC-2012 validation set. The residual version trained faster and reached slightly better final recall on the validation set.

- 성능비교

Network	Top-1 Error	Top-5 Error
BN-Inception [6]	25.2%	7.8%
Inception-v3 [15]	21.2%	5.6%
Inception-ResNet-v1	21.3%	5.5%
Inception-v4	20.0%	5.0%
Inception-ResNet-v2	19.9%	4.9%

Table 2. Single crop - single model experimental results. Reported on the non-blacklisted subset of the validation set of ILSVRC 2012.

총정리

- Reduction Block

    - 신경망의 특정 단계에서 특징맵의 크기를 감소시키면서 동시에 채널 수를 증가시키는 것
    - 네트워크가 더 넓은 수용 필드를 갖게 하여 더 큰 컨텍스트 정보를 학습할 수 있도록 함
    - 입력 데이터의 공간적 차원은 줄어들지만, 깊이는 증가하여 데이터의 추상적인 특징을 더 잘 포착할 수 있게 됨
- Inception-ResNet Block
    - 다양한 크기의 컨볼루션 필터를 병렬로 적용하여 이미지의 다양한 특징을 동시에 포착
    - 각각의 경로는 서로 다른 크기의 컨볼루션 연산을 수행하고, 그 결과를 합치게 함
    - 다양한 스케일에서 이미지의 특징을 포착할 수 있는 능력을 네트워크에 부여함
    - 각 블록의 출력에 입력을 더하는 잔여 연결이 포함되어 있어 네트워크가 더 깊어질 때 발생할 수 있는 기울기 소실 문제를 완

- STEM
    - STEM레이어는 네트워크의 입력부에 위치하며, 입력 이미지의 크기를 줄이고 특징을 추출하는 역할
    - 일반적으로 여러 개의 합성곱(Convolution)층과 최대 풀링(Max Pooling) 층으로 구성되어 있음
    - 이미지의 초기 고차원 특징들을 감지하고 차원 축소를 진행하여 효율적인 연산을 가능하게 함
- Inception-ResNet-A
    - 여러가지 크기의 커널을 가진 합성곱 층을 병렬로 배치하여, 다양한 스케일의 특징을 추출함
    - 이를 결합하기 전에 잔여 연결(Residual Connection)을 통해 입력을 출략에 직접적으로 더하여 기울기 소실 문제를 줄임
- Reduction-A
    - Inception-ResNet-A 블록 이후에 위치하며, 네트워크의 차원을 줄이는 역할
    - 일반적으로 최대 풀링 층과 함께 컨볼루션 연산을 통해 특징 맵의 차원을 줄이며, 이는 다음 층에서 더 큰 컨택스트를 포착할 수 있도록 도움
- Inception-ResNet-B

    - Inception-ResNet-A보다 더 깊은 층으로, 더 복잡한 특징을 추출함
    - 다양한 크기의 커널을 병렬로 사용하고, 잔여 연결을 통해 입력을 직접적으로 출력에 더해줌, 이는 더 깊은 층에서도 학습이 잘 이루어질 수 있도록 도움
- Reduction-B
    - Inception-ResNet-B 블록 이후에 위치하며, 특징 맵의 차원을 줄임
    - 더 깊은 층으로의 전환을 위해 특징 맵의 크기를 조정하는 중요한 역할
- Inception-ResNet-C
    - 더욱 세분화된 특징을 처리하기 위해 설계되었으며, 가장 깊은 층에서 미세한 정보들을 잡아내는데 중요하고, 다른 Inception-ResNet블록과 마찬가지로 다양한 크기의 커널과 잔여 연결을 사용
- Average Pooling
    - 네트워크의 마지막에 위치하며, 각 특징 맵의 평균 값을 계산하는 풀링층
    - 최종적으로 특징을 요약하여 각 클래스에 대한 예측을 생성하기 위한 분류기로 전달되는 특징 벡터를 만드는데 도움을 줌