**Xception Deep Learning with Depthwise Separable Convolutions**

Franc¸ois Chollet Google, Inc.

**Abstract**

We present an interpretation of Inception modules in convolutional neural networks as being an intermediate step in-between regular convolution and the depthwise separable convolution operation (a depthwise convolution followed by a pointwise convolution). In this light, a depthwise separable convolution can be understood as an Inception module with a maximally large number of towers. This observation leads us to propose a novel deep convolutional neural network architecture inspired by Inception, where Inception modules have been replaced with depthwise separable convolutions. We show that this architecture, dubbed Xception, slightly outperforms Inception V3 on the ImageNet dataset (which Inception V3 was designed for), and significantly outperforms Inception V3 on a larger image classification dataset comprising 350 million images and 17,000 classes. Since the Xception architecture has the same number of parameters as Inception V3, the performance gains are not due to increased capacity but rather to a more efficient use of model parameters.

우리는 컨볼루션 신경망의 인셉션 모듈에 대한 해석을 정규 컨볼루션과 깊이 분리 가능한 컨볼루션 연산 사이의 중간 단계로서 제시한다(깊이 컨볼루션 다음에 포인트 컨볼루션). 이러한 관점에서 깊이별로 분리 가능한 컨볼루션은 최대 수의 타워를 가진 인셉션 모듈로 이해할 수 있습니다. 이러한 관찰은 Inception 모듈이 깊이 구분 가능한 Convolution으로 대체된 Inception에서 영감을 받은 새로운 심층 컨볼루션 신경 네트워크 아키텍처를 제안하도록 유도합니다. Xception이라는 이 아키텍처는 ImageNet 데이터셋(Inception V3용으로 설계됨)에서 Inception V3를 약간 능가하며, 3억 5천만 개의 이미지와 17,000개의 클래스로 구성된 대규모 이미지 분류 데이터셋에서 Inception V3을 크게 능가합니다. Xception 아키텍처는 Inception V3과 동일한 수의 매개변수를 가지므로 성능 향상은 용량 증가가 아니라 모델 매개변수를 보다 효율적으로 사용하기 때문입니다.

1. **Introduction**

Convolutional neural networks have emerged as the master algorithm in computer vision in recent years, and developing recipes for designing them has been a subject of considerable attention. The history of convolutional neural network design started with LeNet-style models [10], which were simple stacks of convolutions for feature extraction and max-pooling operations for spatial sub-sampling. In 2012, these ideas were refined into the AlexNet architecture [9], where convolution operations were being repeated multiple times in-between max-pooling operations, allowing the network to learn richer features at every spatial scale. What followed was a trend to make this style of network increasingly deeper, mostly driven by the yearly ILSVRC competition; first with Zeiler and Fergus in 2013 [25] and then with the VGG architecture in 2014 [18].

최근 몇 년 사이 컴퓨터 비전의 마스터 알고리즘으로 떠오른 컨볼루션 신경망은 이를 설계하는 레시피 개발에도 상당한 관심이 집중되고 있다. Convolutional Neural Network 설계의 역사는 피쳐 추출과 공간 서브샘플링을 위한 최대 풀링 연산을 위한 단순한 Convolution 스택[10]인 LeNet-style 모델로 시작했다. 2012년에 이러한 아이디어는 Max-pooling 운영 간에 Convolution 운영이 여러 번 반복되어 모든 공간 규모에서 보다 풍부한 기능을 네트워크가 학습할 수 있도록 하는 AlexNet 아키텍처[9]로 세분화되었습니다. 이러한 유형의 네트워크는 2013년 Zeiler와 Fergus를 시작으로 2014년 VGG 아키텍처를 통해 매년 열리는 ILS VRC 경쟁에 의해 더욱 심화되고 있습니다[18].

At this point a new style of network emerged, the Inception architecture, introduced by Szegedy et al. in 2014 [20] as GoogLeNet (Inception V1), later refined as Inception V2[7], Inception V3 [21], and most recently Inception-ResNet [19]. Inception itself was inspired by the earlier NetworkIn-Network architecture [11]. Since its first introduction, Inception has been one of the best performing family of models on the ImageNet dataset [14], as well as internal datasets in use at Google, in particular JFT [5].

이 시점에서 2014년 Szegdy et al.[20]에 의해 GoogLeNet (Inception V1)으로 소개된 인셉션 아키텍처는 나중에 인셉션 V2[7], 인셉션 V3[21], 그리고 가장 최근의 인셉션-ResNet[19]로 개선되었다. Inception 자체는 이전의 Network-In-Network 아키텍처에서 영감을 받았습니다 [11]. 첫 도입 이후, Inception은 Google, 특히 JFT[5]에서 사용 중인 내부 데이터셋뿐만 아니라 ImageNet 데이터셋[14]에서 가장 성능이 우수한 모델 제품군 중 하나입니다.

The fundamental building block of Inception-style models is the Inception module, of which several different versions exist. In figure 1 we show the canonical form of an Inception module, as found in the Inception V3 architecture. An Inception model can be understood as a stack of such modules. This is a departure from earlier VGG-style networks which were stacks of simple convolution layers.

인셉션 스타일 모형의 기본 구성 요소는 인셉션 모듈이며, 그 중 몇 가지 다른 버전이 존재한다. 그림 1에서는 Inception V3 아키텍처에서 볼 수 있는 Inception 모듈의 표준 형식을 보여줍니다. 인셉션 모델은 그러한 모듈의 스택으로 이해될 수 있다. 이는 단순한 컨볼루션 레이어의 스택이었던 이전의 VGG 스타일 네트워크에서 벗어난 것입니다.

While Inception modules are conceptually similar to convolutions (they are convolutional feature extractors), they empirically appear to be capable of learning richer representations with less parameters. How do they work, and how do they differ from regular convolutions? What design strategies come after Inception?

인셉션 모듈은 개념적으로 컨볼루션(convolutional feature extractor)과 유사하지만, 경험적으로 더 적은 매개변수로 더 풍부한 표현을 배울 수 있는 것으로 보인다. 어떻게 작동하며, 일반 변환과 어떻게 다릅니까? Inception 다음에는 어떤 설계 전략이 있습니까?

* 1. **. The Inception hypothesis**

A convolution layer attempts to learn filters in a 3D space, with 2 spatial dimensions (width and height) and a channel dimension; thus a single convolution kernel is tasked with simultaneously mapping cross-channel correlations and spatial correlations.

콘볼루션 계층은 2개의 공간 치수(폭과 높이)와 채널 치수를 가진 3D 공간에서 필터를 학습하려고 시도합니다. 따라서 단일 콘볼루션 커널은 채널 간 상관 관계와 공간 상관 관계를 동시에 매핑하는 작업을 수행합니다.

This idea behind the Inception module is to make this process easier and more efficient by explicitly factoring it into a series of operations that would independently look at cross-channel correlations and at spatial correlations. More precisely, the typical Inception module first looks at crosschannel correlations via a set of 1x1 convolutions, mapping the input data into 3 or 4 separate spaces that are smaller than the original input space, and then maps all correlations in these smaller 3D spaces, via regular 3x3 or 5x5 convolutions. This is illustrated in figure 1. In effect, the fundamental hypothesis behind Inception is that cross-channel correlations and spatial correlations are sufficiently decoupled that it is preferable not to map them jointly 1.

인셉션 모듈의 이면에는 이 프로세스를 채널 간 상관 관계와 공간적 상관 관계를 독립적으로 살펴볼 수 있는 일련의 작업으로 명시적으로 팩터링하여 보다 쉽고 효율적으로 만들기 위한 아이디어가 있습니다. 보다 정확하게 말하면, 일반적인 Inception 모듈은 먼저 1x1 컨볼루션 세트를 통해 채널 간 상관 관계를 살펴보고, 입력 데이터를 원래 입력 공간보다 작은 3 또는 4개의 별도 공간으로 매핑한 다음, 일반 3x3 또는 5x5 컨볼루션을 통해 이러한 작은 3D 공간의 모든 상관 관계를 매핑합니다. 이는 그림 1에 설명되어 있습니다. 실제로 인셉션의 기본 가설은 교차 채널 상관관계와 공간 상관관계가 충분히 분리되어 있으므로 이들을 공동으로 매핑하지 않는 것이 바람직하다는 것이다1.

Consider a simplified version of an Inception module that only uses one size of convolution (e.g. 3x3) and does not include an average pooling tower (figure 2). This Inception module can be reformulated as a large 1x1 convolution followed by spatial convolutions that would operate on nonoverlapping segments of the output channels (figure 3). This observation naturally raises the question: what is the effect of the number of segments in the partition (and their size)? Would it be reasonable to make a much stronger hypothesis than the Inception hypothesis, and assume that cross-channel correlations and spatial correlations can be mapped completely separately?

하나의 크기의 컨볼루션(예: 3x3)만 사용하고 평균 풀링 타워(그림 2)는 포함하지 않는 인셉션 모듈의 단순화된 버전을 생각해 보자. 이 인셉션 모듈은 출력 채널의 겹치지 않는 세그먼트에서 동작하는 공간적 변환에 이어 큰 1x1 변환으로 재구성될 수 있습니다(그림 3). 이러한 관찰은 파티션의 세그먼트 수와 세그먼트의 크기에 미치는 영향은 무엇인가라는 질문을 자연스럽게 제기합니다. 인셉션 가설보다 훨씬 더 강력한 가설을 세우고 교차 채널 상관 관계와 공간 상관 관계를 완전히 별도로 매핑할 수 있다고 가정하는 것이 합리적일까요?

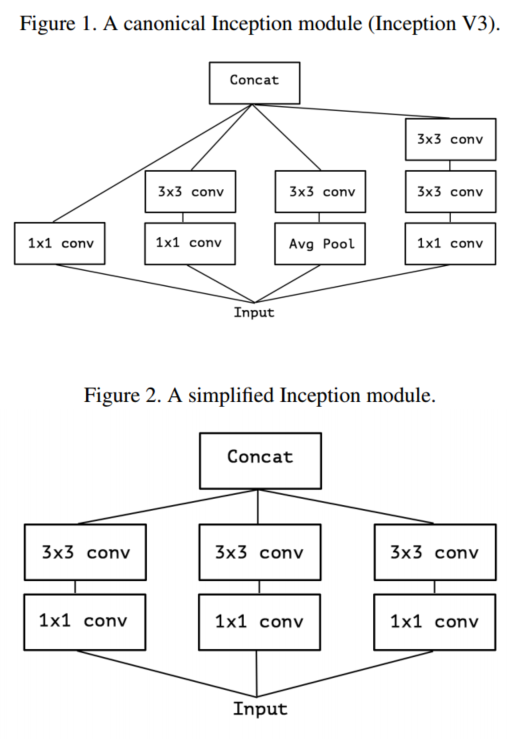


Figure 1. A canonical Inception module (Inception V3)

그림 1 표준 Inception 모듈(Inception V3)

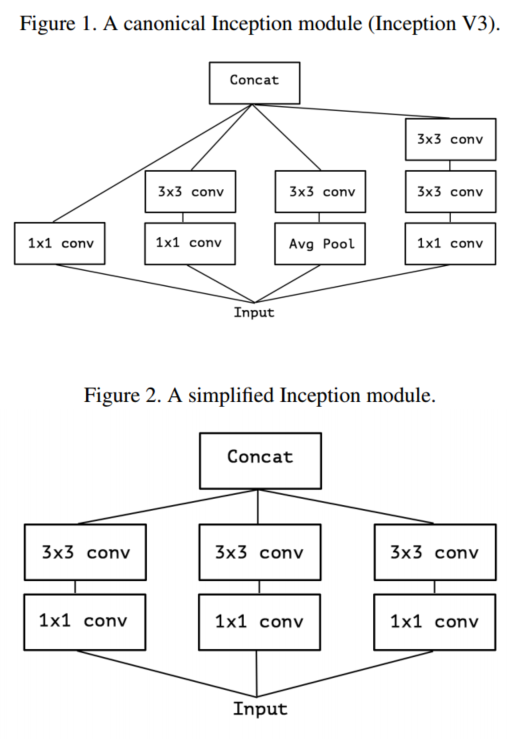


Figure 2. A simplified Inception module

그림 2 간단한 Inception 모듈.

**1.2. The continuum between convolutions and separable convolutions**

An “extreme” version of an Inception module, based on this stronger hypothesis, would first use a 1x1 convolution to map cross-channel correlations, and would then separately map the spatial correlations of every output channel. This is shown in figure 4. We remark that this extreme form of an Inception module is almost identical to a depthwise separable convolution, an operation that has been used in neural network design as early as 2014 [15] and has become more popular since its inclusion in the TensorFlow framework [1] in 2016.

이러한 더 강력한 가설에 기초한 인셉션 모듈의 "극한" 버전은 먼저 채널 간 상관 관계를 매핑하기 위해 1x1 컨볼루션을 사용한 다음 모든 출력 채널의 공간 상관 관계를 별도로 매핑합니다. 이것은 그림 4에 나와 있습니다. 우리는 이러한 극단적인 형태의 인셉션 모듈이 이르면 2014년[15]부터 신경망 설계에 사용되어 왔으며 2016년 TensorFlow 프레임워크 [1]에 포함된 이후 더욱 인기를 끌었던 연산인 깊이 분리 가능 컨볼루션과 거의 동일하다고 언급한다.

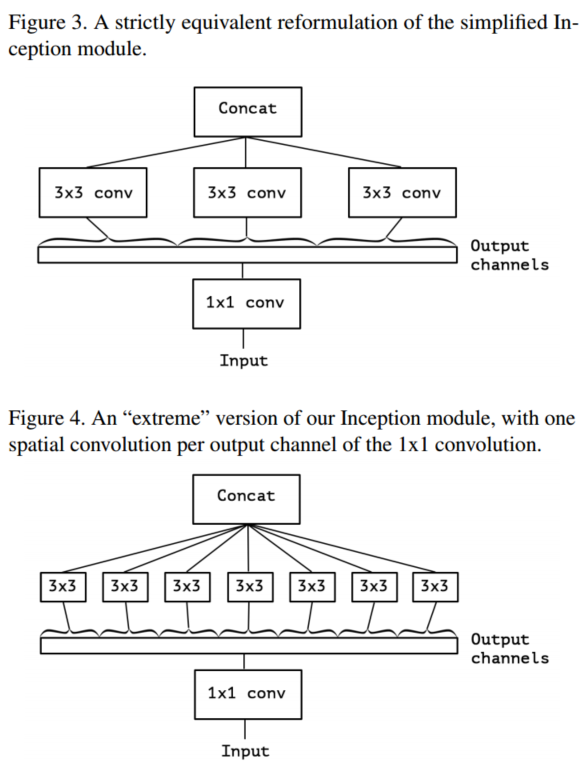


Figure 3. A strictly equivalent reformulation of the simplified Inception module.

그림 3 단순화된 Inception 모듈을 완전히 동일한 방식으로 재구성한다.

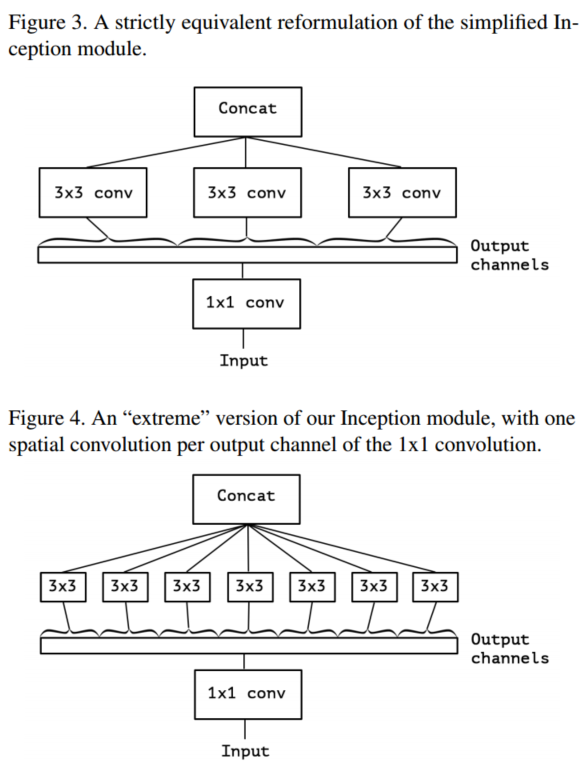


Figure 4. An “extreme” version of our Inception module, with one spatial convolution per output channel of the 1x1 convolution.

그림 4 Inception 모듈의 "익스트림" 버전으로, 1x1 회전의 출력 채널당 1개의 공간 회전이다.

A depthwise separable convolution, commonly called “separable convolution” in deep learning frameworks such as TensorFlow and Keras, consists in a depthwise convolution, i.e. a spatial convolution performed independently over each channel of an input, followed by a pointwise convolution, i.e. a 1x1 convolution, projecting the channels output by the depthwise convolution onto a new channel space. This is not to be confused with a spatially separable convolution, which is also commonly called “separable convolution” in the image processing community.

TensorFlow 및 Keras와 같은 딥러닝 프레임워크에서 일반적으로 "분리 가능한 컨볼루션"이라고 불리는 깊이별 분리형 컨볼루션(즉, 입력의 각 채널에 대해 독립적으로 수행되는 공간 컨볼루션)으로 구성되며, 이어서 점별 컨볼루션(즉, 1x1 컨볼루션)이 출력 채널을 투영합니다. 심도별 변환을 새 채널 공간에 적용합니다. 이는 이미지 처리 커뮤니티에서 일반적으로 "분리 가능한 컨볼루션"이라고도 하는 공간적으로 분리 가능한 컨볼루션과 혼동해서는 안 됩니다.

Two minor differences between and “extreme” version of an Inception module and a depthwise separable convolution would be:

• The order of the operations: depthwise separable convolutions as usually implemented (e.g. in TensorFlow) perform first channel-wise spatial convolution and then perform 1x1 convolution, whereas Inception performs the 1x1 convolution first.

• The presence or absence of a non-linearity after the first operation. In Inception, both operations are followed by a ReLU non-linearity, however depthwiseseparable convolutions are usually implemented without non-linearities.

인셉션 모듈의 "극한" 버전과 깊이 있게 분리할 수 있는 개념의 두 가지 사소한 차이점은 다음과 같다.

• 연산 순서: 일반적으로 구현되는 깊이 분리 가능한 변환(예: TensorFlow)은 채널별 공간 변환을 먼저 수행한 후 1x1 변환을 수행하는 반면, 인셉션은 1x1 변환을 먼저 수행합니다.

• 첫 번째 연산 후 비선형성의 유무 인셉션에서, 두 운영 모두 ReLU 비선형성이 뒤따르지만, 깊이 측면에서는 분리 가능한 컨볼루션은 일반적으로 비선형성이 없이 구현된다.

We argue that the first difference is unimportant, in particular because these operations are meant to be used in a stacked setting. The second difference might matter, and we investigate it in the experimental section (in particular see figure 10).

특히 이러한 연산은 누적된 환경에서 사용되기 때문에 첫 번째 차이는 중요하지 않다고 주장합니다. 두 번째 차이는 문제가 될 수 있으며, 실험 부분에서 조사합니다(특히 그림 10 참조).

We also note that other intermediate formulations of Inception modules that lie in between regular Inception modules and depthwise separable convolutions are also possible: in effect, there is a discrete spectrum between regular convolutions and depthwise separable convolutions, parametrized by the number of independent channel-space segments used for performing spatial convolutions. A regular convolution (preceded by a 1x1 convolution), at one extreme of this spectrum, corresponds to the single-segment case; a depthwise separable convolution corresponds to the other extreme where there is one segment per channel; Inception modules lie in between, dividing a few hundreds of channels into 3 or 4 segments. The properties of such intermediate modules appear not to have been explored yet.

우리는 또한 정규 인셉션 모듈과 깊이 분리 가능한 컨볼루션 사이에 있는 인셉션 모듈의 다른 중간 공식화도 가능하다는 점에 주목한다. 사실상 정규 컨볼루션과 깊이 분리 가능한 컨볼루션 사이에는 사용된 독립적인 채널-공간 세그먼트의 수로 매개 변수화되는 이산 스펙트럼이 존재한다. 공간적 변환을 수행하는 데 사용됩니다. 이 스펙트럼의 한 극단에서 정규 컨볼루션(1x1 컨볼루션 앞에 있음)은 단일 세그먼트 사례에 해당합니다. 깊이별로 분리 가능한 컨볼루션(convolution)은 채널당 한 세그먼트가 있는 다른 극단과 일치합니다. 인셉션 모듈은 수백 개의 채널을 3개 또는 4개의 세그먼트로 나눕니다. 이러한 중간 모듈의 속성은 아직 조사되지 않은 것으로 보입니다.

Having made these observations, we suggest that it may be possible to improve upon the Inception family of architectures by replacing Inception modules with depthwise separable convolutions, i.e. by building models that would be stacks of depthwise separable convolutions. This is made practical by the efficient depthwise convolution implementation available in TensorFlow. In what follows, we present a convolutional neural network architecture based on this idea, with a similar number of parameters as Inception V3, and we evaluate its performance against Inception V3 on two large-scale image classification task.

이러한 관찰을 통해, 우리는 인셉션 모듈을 깊이별 분리 가능 컨버전스로 대체함으로써, 즉 깊이별 분리 가능 컨버전스의 스택이 될 모델을 구축함으로써 인셉션 아키텍처 제품군을 개선할 수 있을 것이라고 제안한다. 이것은 TensorFlow에서 사용할 수 있는 효율적인 깊이별 컨볼루션 구현에 의해 실용화됩니다. 이어서 우리는 인셉션 V3와 유사한 수의 매개변수를 가진 이 아이디어에 기초한 컨볼루션 신경망 아키텍처를 제시하고, 두 개의 대규모 이미지 분류 작업에서 인셉션 V3에 대한 성능을 평가한다.

1. **Prior Work**

The present work relies heavily on prior efforts in the following areas:

• Convolutional neural networks [10, 9, 25], in particular the VGG-16 architecture [18], which is schematically similar to our proposed architecture in a few respects.

• The Inception architecture family of convolutional neural networks [20, 7, 21, 19], which first demonstrated the advantages of factoring convolutions into multiple branches operating successively on channels and then on space.

• Depthwise separable convolutions, which our proposed architecture is entirely based upon. While the use of spatially separable convolutions in neural networks has a long history, going back to at least 2012 [12] (but likely even earlier), the depthwise version is more recent. Laurent Sifre developed depthwise separable convolutions during an internship at Google Brain in 2013, and used them in AlexNet to obtain small gains in accuracy and large gains in convergence speed, as well as a significant reduction in model size. An overview of his work was first made public in a presentation at ICLR 2014 [23]. Detailed experimental results are reported in Sifre’s thesis, section 6.2 [15]. This initial work on depthwise separable convolutions was inspired by prior research from Sifre and Mallat on transformation-invariant scattering [16, 15]. Later, a depthwise separable convolution was used as the first layer of Inception V1 and Inception V2 [20, 7]. Within Google, Andrew Howard [6] has introduced efficient mobile models called MobileNets using depthwise separable convolutions. Jin et al. in 2014 [8] and Wang et al. in 2016 [24] also did related work aiming at reducing the size and computational cost of convolutional neural networks using separable convolutions. Additionally, our work is only possible due to the inclusion of an efficient implementation of depthwise separable convolutions in the TensorFlow framework [1].

• Residual connections, introduced by He et al. in [4], which our proposed architecture uses extensively.

본 연구는 다음 분야의 이전 노력에 크게 의존하고 있습니다.

• 컨볼루션 신경망[10, 9, 25], 특히 VGG-16 아키텍처[18]는 몇 가지 측면에서 우리가 제안한 아키텍처와 도식적으로 유사합니다.

• 인셉션 아키텍처의 컨볼루션 신경 네트워크 제품군[20, 7, 21, 19]은 먼저 채널에서 연속 작동한 후 공간에서 작동하는 여러 갈래로 컨볼루션의 장점을 입증했습니다.

• 깊이 있게 분리할 수 있는 컨버전스, 우리가 제안한 아키텍처가 전적으로 기반이 됩니다. 신경 네트워크에서 공간적으로 분리 가능한 합성곱의 사용은 적어도 2012년[12]으로 거슬러 올라가 오랜 역사를 가지고 있지만, 깊이 버전은 더 최근 버전이다. Laurent Sifre는 2013년 Google Brain에서 인턴십을 하는 동안 깊이별로 분리할 수 있는 컨버전스(convolutions)를 개발했으며, 이를 AlexNet에서 사용하여 정확도가 약간 향상되고 수렴 속도가 크게 향상될 뿐만 아니라 모델 크기를 크게 줄였습니다. 그의 작업에 대한 개요는 ICLR 2014에서 열린 프레젠테이션에서 처음 공개되었습니다 [23]. 자세한 실험 결과는 Sifre의 논문, 섹션 6.2[15]에 보고됩니다. 깊이별로 분리 가능한 컨버전스에 대한 이러한 초기 연구는 Sifre와 Mallaton의 변환-불변 산란에서의 이전 연구에서 영감을 받았습니다 [16, 15]. 이후 깊이별로 분리 가능한 컨볼루션을 인셉션 V1과 인셉션 V2의 첫 번째 계층으로 사용하였다[20, 7]. Google에서 Andrew Howard[6]는 깊이 구분 가능한 컨버전스를 사용하여 MobileNets라는 효율적인 모바일 모델을 도입했습니다. 2014년 진 외 연구진[8]과 2016년 왕 외 연구진[24]도 분리 가능한 컨볼루션(convolution)을 사용하여 컨볼루션(convolutional neural network)의 크기와 계산 비용을 줄이는 것을 목표로 관련 연구를 수행했다. 또한, 우리의 작업은 오직 깊이별 분리 가능 컨볼루션을 TensorFlow 프레임워크에 효율적으로 구현하기 때문에 가능합니다 [1].

• He 등이 [4]에 소개한 잔류 연결 방식 제안 아키텍처가 광범위하게 사용합니다.

1. **The Xception architecture**

We propose a convolutional neural network architecture based entirely on depthwise separable convolution layers. In effect, we make the following hypothesis: that the mapping of cross-channels correlations and spatial correlations in the feature maps of convolutional neural networks can be entirely decoupled. Because this hypothesis is a stronger version of the hypothesis underlying the Inception architecture, we name our proposed architecture Xception, which stands for “Extreme Inception”.

우리는 전적으로 깊이별로 분리 가능한 컨볼루션 레이어를 기반으로 하는 컨볼루션 신경 네트워크 아키텍처를 제안합니다. 사실상, 우리는 다음과 같은 가설을 세운다: 컨볼루션 신경망의 특성 맵에서 교차 채널 상관 관계와 공간 상관 관계 매핑은 완전히 분리될 수 있다. 이 가설은 인셉션 아키텍처의 기초가 되는 가설의 강력한 버전이기 때문에, 우리는 제안된 아키텍처의 이름을 "Exception"(극한 인셉션)으로 지정합니다.

A complete description of the specifications of the network is given in figure 5. The Xception architecture has 36 convolutional layers forming the feature extraction base of the network. In our experimental evaluation we will exclusively investigate image classification and therefore our convolutional base will be followed by a logistic regression layer. Optionally one may insert fully-connected layers before the logistic regression layer, which is explored in the experimental evaluation section (in particular, see figures 7 and 8). The 36 convolutional layers are structured into 14 modules, all of which have linear residual connections around them, except for the first and last modules.

네트워크 사양에 대한 전체 설명은 그림 5에 나와 있습니다. Xception 아키텍처에는 네트워크의 피쳐 추출 기반을 형성하는 36개의 컨볼루션 레이어가 있습니다. 우리의 실험 평가에서 우리는 이미지 분류를 독점적으로 조사할 것이며 따라서 우리의 컨볼루션 베이스에는 로지스틱 회귀 레이어가 뒤따를 것입니다. 선택적으로 로지스틱 회귀 분석 도면층 앞에 완전히 연결된 도면층을 삽입할 수 있습니다(특히 그림 7 및 8 참조). 36개의 컨볼루션 레이어는 14개의 모듈로 구성되어 있으며, 첫 번째 모듈과 마지막 모듈을 제외하고 모두 주변에 선형 잔여 연결부가 있습니다.

In short, the Xception architecture is a linear stack of depthwise separable convolution layers with residual connections. This makes the architecture very easy to define and modify; it takes only 30 to 40 lines of code using a highlevel library such as Keras [2] or TensorFlow-Slim [17], not unlike an architecture such as VGG-16 [18], but rather unlike architectures such as Inception V2 or V3 which are far more complex to define. An open-source implementation of Xception using Keras and TensorFlow is provided as part of the Keras Applications module2, under the MIT license.

간단히 말해, Xception 아키텍처는 깊이별로 분리 가능한 컨볼루션 레이어의 선형 스택이며 잔존 연결입니다. 따라서 정의 및 수정이 매우 용이합니다. Keras [2] 또는 TensorFlow-Slim [17]과 같은 고급 라이브러리를 사용하는 코드 행은 VGG-16[18]과 같은 아키텍처와 달리 정의하기가 훨씬 더 복잡한 Inception V2 또는 V3과 같은 아키텍처와는 달리 30~40개에 불과합니다. Keras 및 TensorFlow를 사용하는 Xception의 오픈 소스 구현은 Keras Applications 모듈 2의 일부로 MIT 라이센스에 따라 제공됩니다.

1. **Experimental evaluation**

We choose to compare Xception to the Inception V3 architecture, due to their similarity of scale: Xception and Inception V3 have nearly the same number of parameters (table 3), and thus any performance gap could not be attributed to a difference in network capacity. We conduct our comparison on two image classification tasks: one is the well-known 1000-class single-label classification task on the ImageNet dataset [14], and the other is a 17,000-class multi-label classification task on the large-scale JFT dataset.

우리는 Exception과 Inception V3의 규모 유사성 때문에 Exception을 Inception V3 아키텍처와 비교하기로 선택했다. 즉, Xception과 Inception V3의 매개변수 수는 거의 같으며(표 3), 따라서 성능 격차는 네트워크 용량의 차이에 기인할 수 없다. 두 가지 이미지 분류 작업을 비교합니다. 하나는 ImageNet 데이터 집합의 잘 알려진 1000 클래스 단일 레이블 분류 작업이고 다른 하나는 대규모 JFT 데이터 집합의 17,000 클래스 다중 레이블 분류 작업입니다.

**4.1. The JFT dataset**

JFT is an internal Google dataset for large-scale image classification dataset, first introduced by Hinton et al. in [5], which comprises over 350 million high-resolution images annotated with labels from a set of 17,000 classes. To evaluate the performance of a model trained on JFT, we use an auxiliary dataset, FastEval14k.

JFT는 [5]에서 Hinton 등에 의해 처음 소개된 대규모 이미지 분류 데이터셋용 내부 Google 데이터셋으로, 17,000개 클래스 집합의 레이블이 달린 3억 5천만 개 이상의 고해상도 이미지로 구성됩니다. JFT에 대해 교육된 모델의 성능을 평가하기 위해 보조 데이터 세트인 FastEval14k를 사용합니다.

FastEval14k is a dataset of 14,000 images with dense annotations from about 6,000 classes (36.5 labels per image on average). On this dataset we evaluate performance using Mean Average Precision for top 100 predictions (MAP@100), and we weight the contribution of each class to MAP@100 with a score estimating how common (and therefore important) the class is among social media images. This evaluation procedure is meant to capture performance on frequently occurring labels from social media, which is crucial for production models at Google.

FastEval14k는 약 6,000개의 클래스(이미지당 평균 36.5개의 레이블)에서 고밀도 주석을 가진 14,000개의 이미지 데이터 집합입니다. 이 데이터 집합에서는 상위 100개 예측(MAP@100)에 대해 Mean Average Precision을 사용하여 성능을 평가하고, 소셜 미디어 이미지에서 클래스가 얼마나 공통(따라서 중요한)지를 추정하는 점수로 MAP@100에 대한 각 클래스의 기여도를 평가합니다. 이 평가절차는 구글에서 생산모델에 중요한 소셜미디어에서 자주 발생하는 라벨의 성능을 포착하기 위한 것이다.

**4.2. Optimization configuration**

A different optimization configuration was used for ImageNet and JFT:

• On ImageNet: Optimizer: SGD, Momentum: 0.9, Initial learning rate: 0.045, Learning rate decay: decay of rate 0.94 every 2 epochs

• On JFT: Optimizer: RMSprop [22], Momentum: 0.9, Initial learning rate: 0.001, Learning rate decay: decay of rate 0.9 every 3,000,000 samples

생략

For both datasets, the same exact same optimization configuration was used for both Xception and Inception V3. Note that this configuration was tuned for best performance with Inception V3; we did not attempt to tune optimization hyperparameters for Xception. Since the networks have different training profiles (figure 6), this may be suboptimal, especially on the ImageNet dataset, on which the optimization configuration used had been carefully tuned for Inception V3.

Additionally, all models were evaluated using Polyak averaging [13] at inference time.

두 데이터셋에 대해 동일한 최적화 구성이 Xception 및 Inception V3 모두에 사용되었습니다. 이 구성은 Inception V3에서 최상의 성능을 발휘하도록 조정되었습니다. Xception에 대한 최적화 하이퍼 매개 변수를 조정하려고 시도하지 않았습니다. 네트워크의 교육 프로필이 다르기 때문에(그림 6), 특히 Inception V3에 맞게 최적화 구성을 세심하게 조정한 ImageNet 데이터셋에서는 이 방법이 차선책일 수 있습니다.

또한 모든 모델은 추론 시 Polyak 평균[13]을 사용하여 평가되었습니다.

**4.3. Regularizaiton configuration**

**Weight decay:** The Inception V3 model uses a weight decay (L2 regularization) rate of 4e − 5, which has been carefully tuned for performance on ImageNet. We found this rate to be quite suboptimal for Xception and instead settled for 1e − 5. We did not perform an extensive search for the optimal weight decay rate. The same weight decay rates were used both for the ImageNet experiments and the JFT experiments.

무게 감소: Inception V3 모델은 ImageNet에서 성능을 주의하여 조정한 무게 감소(L2 정규화) 속도 4e - 5를 사용합니다. 우리는 이 비율이 Xception에 매우 적합하지 않다는 것을 알게 되었고 대신 1e - 5로 정해졌다. 최적의 중량 감소율에 대한 광범위한 검색을 수행하지 않았습니다. ImageNet 실험과 JFT 실험 모두에 동일한 무게 감소율이 사용되었습니다.

**Dropout:** For the ImageNet experiments, both models include a dropout layer of rate 0.5 before the logistic regression layer. For the JFT experiments, no dropout was included due to the large size of the dataset which made overfitting unlikely in any reasonable amount of time.

자퇴: ImageNet 실험의 경우 두 모형 모두 로지스틱 회귀 분석 레이어 앞에 비율 0.5의 드롭아웃 레이어를 포함합니다. JFT 실험의 경우 데이터 집합의 크기가 커서 적당한 시간 내에 오버핏이 불가능했기 때문에 탈락자가 포함되지 않았습니다.

**Auxiliary loss tower:** The Inception V3 architecture may optionally include an auxiliary tower which backpropagates the classification loss earlier in the network, serving as an additional regularization mechanism. For simplicity, we choose not to include this auxiliary tower in any of our models.

보조 손실 타워: 인셉션 V3 아키텍처는 선택적으로 네트워크 초기에 분류손실을 역전파하는 보조탑을 포함시켜 추가적인 정규화 메커니즘의 역할을 할 수 있다. 단순성을 위해, 저희는 어떤 모델에도 이 보조 타워를 포함하지 않기로 했습니다.

**4.4. Training infrastructure**

All networks were implemented using the TensorFlow framework [1] and trained on 60 NVIDIA K80 GPUs each. For the ImageNet experiments, we used data parallelism with synchronous gradient descent to achieve the best classification performance, while for JFT we used asynchronous gradient descent so as to speed up training. The ImageNet experiments took approximately 3 days each, while the JFT experiments took over one month each. The JFT models were not trained to full convergence, which would have taken over three month per experiment.

모든 네트워크는 TensorFlow 프레임워크[1]를 사용하여 구현되었으며 각각 60개의 NVIDIA K80 GPU에서 교육되었습니다. ImageNet 실험에서는 최상의 분류 성능을 달성하기 위해 동기식 구배 강하를 사용하는 데이터 병렬 방식을 사용했으며, JFT에서는 교육 속도를 높이기 위해 비동기식 구배 강하를 사용했습니다. ImageNet 실험은 각각 약 3일이 소요된 반면 JFT 실험은 각각 1개월 이상이 소요되었습니다. JFT 모델은 실험당 3개월 이상이 걸릴 정도로 완전히 융합되도록 훈련되지 않았습니다.

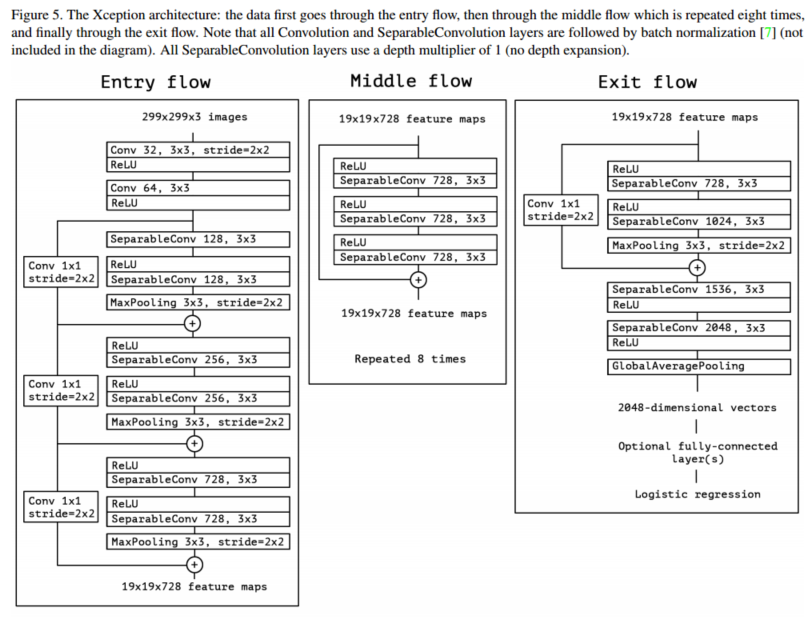


Figure 5. The Xception architecture: the data first goes through the entry flow, then through the middle flow which is repeated eight times, and finally through the exit flow. Note that all Convolution and SeparableConvolution layers are followed by batch normalization (not included in the diagram). All SeparableConvolution layers use a depth multiplier of 1 (no depth expansion).

그림 5 Xception 아키텍처: 데이터는 처음에 엔트리 플로우를 통과한 후 8회 반복되는 중간 플로우를 통과한 후 최종적으로 출구 플로우를 통과합니다. 모든 Convolution 레이어 및 Separable Convolution 레이어 뒤에 배치 정규화(그림에 포함되지 않음)가 계속된다는 점에 유의하십시오. 모든 SeparableConvolution 레이어는 1의 깊이 승수를 사용합니다(깊이 확장 없음).

**4.5. Comparison with Inception V3**

**4.5.1 Classification performance**

All evaluations were run with a single crop of the inputs images and a single model. ImageNet results are reported on the validation set rather than the test set (i.e. on the non-blacklisted images from the validation set of ILSVRC 2012). JFT results are reported after 30 million iterations (one month of training) rather than after full convergence. Results are provided in table 1 and table 2, as well as figure 6, figure 7, figure 8. On JFT, we tested both versions of our networks that did not include any fully-connected layers, and versions that included two fully-connected layers of 4096 units each before the logistic regression layer.

모든 평가는 입력 이미지의 단일 자르기 및 단일 모델로 실행되었습니다. ImageNet 결과는 테스트 세트가 아니라 검증 세트에 보고됩니다(즉, ILSVRC 2012의 검증 세트에 있는 블랙리스트에 없는 이미지에 보고됨). JFT 결과는 완전 융합이 아닌 3000만회(연수 1개월)를 거쳐 보고된다. 결과는 그림 6, 그림 7, 그림 8뿐만 아니라 표 1과 표 2에도 나와 있습니다. JFT에서는 완전히 연결된 계층이 포함되지 않은 네트워크 버전과 로지스틱 회귀 계층 이전에 각각 4096개의 완전히 연결된 두 개의 계층이 포함된 버전을 모두 테스트했습니다.

On ImageNet, Xception shows marginally better results than Inception V3. On JFT, Xception shows a 4.3% relative improvement on the FastEval14k MAP@100 metric. We also note that Xception outperforms ImageNet results reported by He et al. for ResNet-50, ResNet-101 and ResNet-152 [4].

ImageNet에서 Xception은 Inception V3보다 약간 더 나은 결과를 표시합니다. JFT에서 Xception은 FastEval14k MAP@100 메트릭에서 상대적인 4.3% 향상되었습니다. 우리는 또한 Xception이 ResNet-50, ResNet-101 및 ResNet-152 [4]에 대해 He 등이 보고한 ImageNet 결과를 능가한다는 점에 주목합니다.

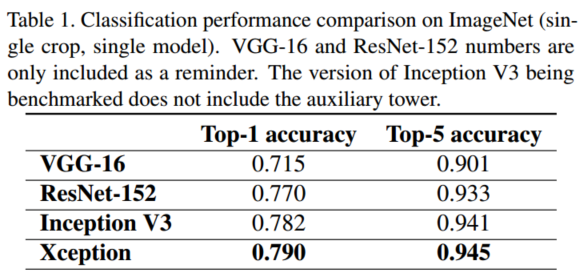


Table 1. Classification performance comparison on ImageNet (single crop, single model). VGG-16 and ResNet-152 numbers are only included as a reminder. The version of Inception V3 being benchmarked does not include the auxiliary tower.

표 1. ImageNet(싱글 크롭, 싱글 모델)에서의 분류 퍼포먼스 비교. VGG-16 및 ResNet-152 번호는 주의사항으로만 포함되어 있습니다. 벤치마킹 대상 Inception V3 버전에는 보조 타워가 포함되어 있지 않습니다.

The Xception architecture shows a much larger performance improvement on the JFT dataset compared to the ImageNet dataset. We believe this may be due to the fact that Inception V3 was developed with a focus on ImageNet and may thus be by design over-fit to this specific task. On the other hand, neither architecture was tuned for JFT. It is likely that a search for better hyperparameters for Xception on ImageNet (in particular optimization parameters and regularization parameters) would yield significant additional improvement.

Xception 아키텍처는 ImageNet 데이터셋에 비해 JFT 데이터셋의 성능이 크게 향상되었습니다. 이는 Inception V3가 ImageNet에 중점을 두고 개발되었기 때문에 이 특정 작업에 과도하게 적합하게 설계되었기 때문일 수 있습니다. 반면에, 두 아키텍처 모두 JFT에 맞게 조정되지 않았습니다. ImageNet에서 Xception에 대한 더 나은 하이퍼 파라미터(특히 최적화 파라미터 및 정규화 파라미터)를 검색하면 상당한 추가 개선 효과를 얻을 수 있습니다.

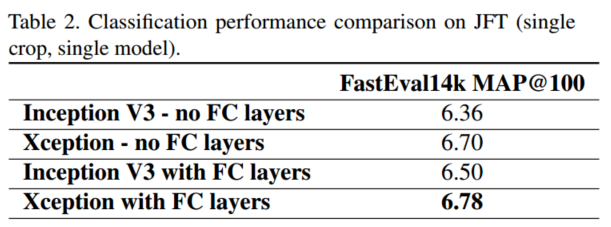


Table 2. Classification performance comparison on JFT (single crop, single model).

표 2 JFT(단품종, 단품종)의 분류 성능 비교.

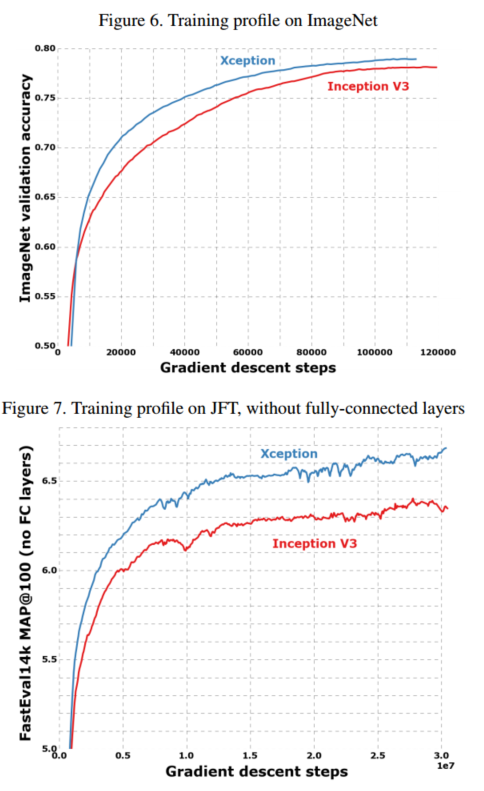


Figure 6. Training profile on ImageNet

그림 6.ImageNet 교육 프로필

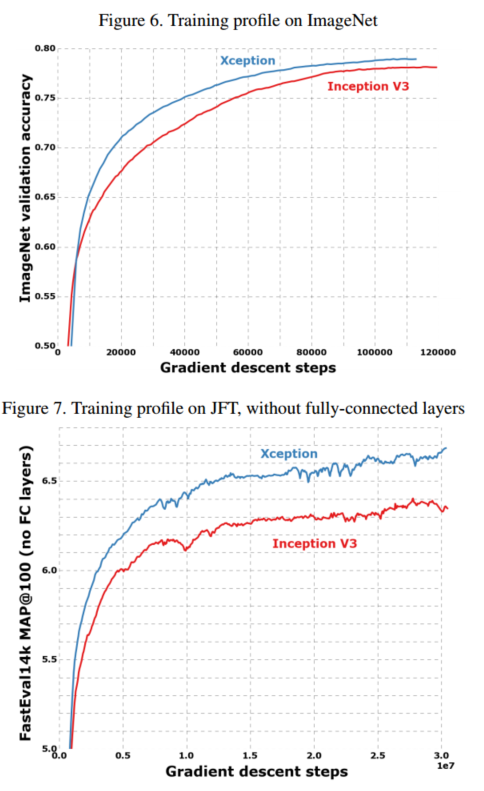


Figure 7. Training profile on JFT, without fully-connected layers

그림 7 JFT에서의 트레이닝 프로파일(완전 접속 레이어 없음)

**4.5.2 Size and speed**

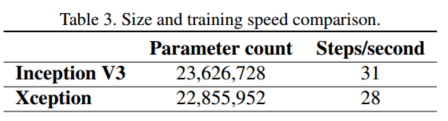


Table 3. Size and training speed comparison.

표 3 규모와 트레이닝 속도 비교.

In table 3 we compare the size and speed of Inception V3 and Xception. Parameter count is reported on ImageNet (1000 classes, no fully-connected layers) and the number of training steps (gradient updates) per second is reported on ImageNet with 60 K80 GPUs running synchronous gradient descent. Both architectures have approximately the same size (within 3.5%), and Xception is marginally slower. We expect that engineering optimizations at the level of the depthwise convolution operations can make Xception faster than Inception V3 in the near future. The fact that both architectures have almost the same number of parameters indicates that the improvement seen on ImageNet and JFT does not come from added capacity but rather from a more efficient use of the model parameters.

표 3에서는 Inception V3와 Xception의 크기와 속도를 비교합니다. 매개 변수 수는 ImageNet(1000 클래스, 완전히 연결된 계층 없음)에 보고되고 초당 교육 단계 수(그라데이션 업데이트)는 동기식 그라데이션 강하를 실행하는 60개의 K80 GPU에 보고됩니다. 두 아키텍처의 크기는 거의 동일(3.5% 이내)하며 Xception은 약간 느립니다. 깊이별 Convolution 작업 수준의 엔지니어링 최적화를 통해 가까운 시일 내에 Xception이 Inception V3보다 빨라질 수 있을 것으로 기대합니다. 두 아키텍처의 매개 변수 수가 거의 동일하다는 것은 ImageNet과 JFT에서 나타나는 개선 사항이 용량을 추가한 것이 아니라 모델 매개 변수를 보다 효율적으로 사용함으로써 나타난다는 것을 나타냅니다.

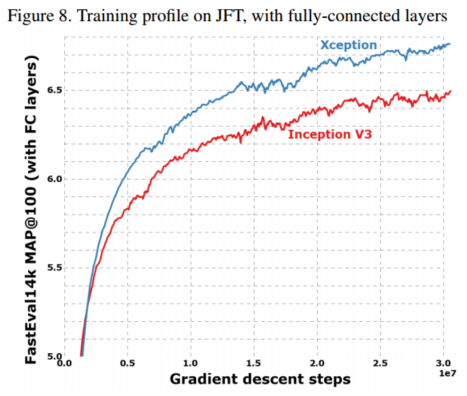


Figure 8. Training profile on JFT, with fully-connected layers

그림 8 JFT에서의 트레이닝 프로파일(완전 접속 레이어 포함)

**4.6. Effect of the residual connections**

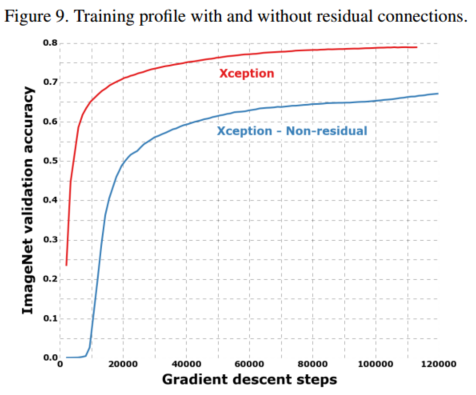


Figure 9. Training profile with and without residual connections

그림 9.잔류 연줄 없이와 뛰면 때론 프로필입니다.

To quantify the benefits of residual connections in the Xception architecture, we benchmarked on ImageNet a modified version of Xception that does not include any residual connections. Results are shown in figure 9. Residual connections are clearly essential in helping with convergence, both in terms of speed and final classification performance. However we will note that benchmarking the non-residual model with the same optimization configuration as the residual model may be uncharitable and that better optimization configurations might yield more competitive results.

Xception 아키텍처에서 잔여 연결의 이점을 수량화하기 위해 ImageNet에서 잔여 연결이 포함되지 않은 수정된 Xception 버전을 벤치마킹했습니다. 결과는 그림 9에 나와 있습니다. 잔류 연결은 속도와 최종 분류 성과 면에서 정합화를 돕는 데 있어 분명히 필수적이다. 그러나 잔류 모델과 동일한 최적화 구성으로 비잔여 모델을 벤치마킹하는 것은 자비롭지 않을 수 있으며, 더 나은 최적화 구성은 더 경쟁력 있는 결과를 낳을 수 있습니다.

Additionally, let us note that this result merely shows the importance of residual connections for this specific architecture, and that residual connections are in no way required in order to build models that are stacks of depthwise separable convolutions. We also obtained excellent results with non-residual VGG-style models where all convolution layers were replaced with depthwise separable convolutions (with a depth multiplier of 1), superior to Inception V3 on JFT at equal parameter count.

또한, 이 결과는 이 특정 아키텍처에 대한 잔여 연결의 중요성을 보여줄 뿐이며 깊이별로 분리 가능한 컨볼루션의 스택 모델을 구축하기 위해 잔여 연결이 필요한 것은 아닙니다. 또한 모든 컨볼루션 레이어가 동일한 파라미터 카운트에서 JFT의 Inception V3보다 우수한 깊이별 분리 가능 컨볼루션(깊이 승수 1)으로 대체된 비잔류 VG 스타일 모델을 사용하여 우수한 결과를 얻었다.

**4.7. Effect of an intermediate activation after pointwise convolutions**

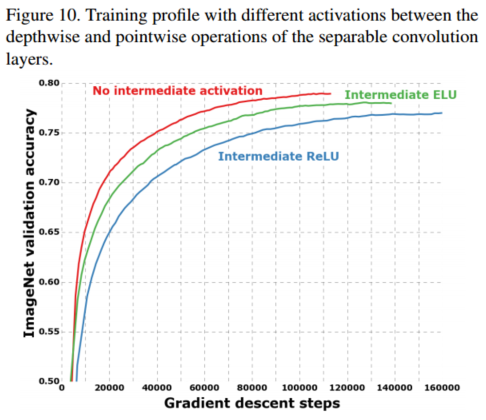


Figure 10. Training profile with different activations between the depthwise and pointwise operations of the separable convolution layers.

그림 10. 분리 가능한 컨볼루션 층의 깊이별 연산과 점별 연산 간에 서로 다른 액티베이션이 있는 교육 프로파일.

We mentioned earlier that the analogy between depthwise separable convolutions and Inception modules suggests that depthwise separable convolutions should potentially include a non-linearity between the depthwise and pointwise operations. In the experiments reported so far, no such nonlinearity was included. However we also experimentally tested the inclusion of either ReLU or ELU [3] as intermediate non-linearity. Results are reported on ImageNet in figure 10, and show that the absence of any non-linearity leads to both faster convergence and better final performance.

우리는 심층적으로 분리할 수 있는 컨볼루션과 인셉션 모듈 사이의 유추에 따르면 심층적으로 분리할 수 있는 컨볼루션은 잠재적으로 심층적으로 운용하는 것과 포인트적으로 운용하는 것 사이의 비선형성을 포함해야 한다고 앞에서 언급하였다. 지금까지 보고된 실험에서는 그러한 비선형성이 포함되지 않았습니다. 그러나 우리는 또한 중간 비선형성으로 ReLU 또는 ELU [3]를 포함하는 것을 실험적으로 테스트했다. 그림 10의 ImageNet에 그 결과가 보고되었으며, 비선형성의 부재로 인해 수렴 속도가 빨라지고 최종 성능이 향상된다는 것을 알 수 있습니다.

This is a remarkable observation, since Szegedy et al. report the opposite result in [21] for Inception modules. It may be that the depth of the intermediate feature spaces on which spatial convolutions are applied is critical to the usefulness of the non-linearity: for deep feature spaces (e.g. those found in Inception modules) the non-linearity is helpful, but for shallow ones (e.g. the 1-channel deep feature spaces of depthwise separable convolutions) it becomes harmful, possibly due to a loss of information.

이는 Szegedy 등이 Inception 모듈에 대해 [21]에서 반대 결과를 보고하기 때문에 주목할 만한 관측치이다. 공간 변환이 적용되는 중간 피쳐 공간의 깊이는 비선형성의 유용성에 매우 중요한 것일 수 있습니다. 깊은 피쳐 공간(예: 인셉션 모듈에서 발견되는 공간)의 경우 비선형성이 유용하지만, 얕은 피쳐 공간(예: 깊이 분리 가능한 컨버전스의 1채널 깊이 피쳐 공간)의 경우에는 도움이 됩니다. 정보의 손실 때문일 수 있습니다.

1. **Future Directions**

We noted earlier the existence of a discrete spectrum between regular convolutions and depthwise separable convolutions, parametrized by the number of independent channelspace segments used for performing spatial convolutions. Inception modules are one point on this spectrum. We showed in our empirical evaluation that the extreme formulation of an Inception module, the depthwise separable convolution, may have advantages over regular a regular Inception module. However, there is no reason to believe that depthwise separable convolutions are optimal. It may be that intermediate points on the spectrum, lying between regular Inception modules and depthwise separable convolutions, hold further advantages. This question is left for future investigation.

앞에서 우리는 공간적 변환을 수행하는 데 사용되는 독립 채널 공간 세그먼트의 수로 파라미터화된 정규 변환과 깊이 분리 가능한 변환 사이에 이산 스펙트럼이 존재한다는 점에 주목했다. 인셉션 모듈은 이 스펙트럼의 한 지점입니다. 우리는 경험적 평가에서 인셉션 모듈의 극단적인 공식화, 즉 깊이별로 분리 가능한 컨볼루션이 정규 인셉션 모듈보다 장점이 있을 수 있음을 보여주었다. 그러나 깊이별로 분리 가능한 합성곱이 최적이라고 믿을 이유는 없습니다. 정기적인 인셉션 모듈과 깊이별로 분리할 수 있는 컨볼루션 사이에 있는 스펙트럼의 중간 지점들이 추가적인 이점을 가질 수 있다. 이 문제는 향후 조사를 위해 남겨두었습니다.

1. **Conclusions**

We showed how convolutions and depthwise separable convolutions lie at both extremes of a discrete spectrum, with Inception modules being an intermediate point in between. This observation has led to us to propose replacing Inception modules with depthwise separable convolutions in neural computer vision architectures. We presented a novel architecture based on this idea, named Xception, which has a similar parameter count as Inception V3. Compared to Inception V3, Xception shows small gains in classification performance on the ImageNet dataset and large gains on the JFT dataset. We expect depthwise separable convolutions to become a cornerstone of convolutional neural network architecture design in the future, since they offer similar properties as Inception modules, yet are as easy to use as regular convolution layers.

우리는 인셉션 모듈이 중간 지점인 가운데 이산 스펙트럼의 양 극단에 있는 컨볼루션과 깊이 분리 가능한 컨볼루션을 보여주었다. 이러한 관찰을 통해 우리는 인셉션 모듈을 신경 컴퓨터 비전 아키텍처에서 깊이별로 분리 가능한 컨버전스로 대체할 것을 제안하게 되었습니다. 우리는 이 아이디어를 바탕으로 Inception V3와 유사한 매개변수 수를 가진 Xception이라는 새로운 아키텍처를 제시했습니다. Inception V3에 비해, Xception은 ImageNet 데이터셋에서 분류 성능이 약간 향상되고 JFT 데이터셋에서 큰 폭으로 향상됩니다. 깊이 있게 분리 가능한 컨버전스는 인셉션 모듈과 유사한 특성을 제공하면서도 일반 컨버전스 레이어처럼 사용하기 쉽기 때문에 향후 컨버전스 네트워크 아키텍처 설계의 초석이 될 것으로 예상합니다.