스퀴즈 및 여기 네트워크

지 후[0000-0002-5150-1003] 리 셴[0000-0002-2283-4976] 사무엘 알바니[0000-0001-9736-5134]강 선[0000-0001-6913-6799] 엔슈아 우[0000-0002-2174-1428]

개요 - 컨볼루션 신경망(CNN)의 핵심 구성 요소는 컨볼루션 연산자로, 네트워크가 각 계층의 로컬 수신 필드 내에서 공간 및 채널별 정보를 모두 융합하여 유익한 특징을 구성할 수 있게 해줍니다. 광범위한 선행 연구에서 이 관계의 공간적 요소를 조사하여 특징 계층 전체에 걸쳐 공간 인코딩의 품질을 향상시킴으로써 CNN의 표현력을 강화하고자 했습니다. 이 연구에서는 채널 관계에 초점을 맞추고 채널 간의 상호 의존성을 명시적으로 모델링하여 채널별 특징 반응을 적응적으로 재보정하는 새로운 아키텍처 단위인 '스퀴즈 앤 익사이팅'(SE) 블록을 제안합니다. 이러한 블록을 함께 쌓아 서로 다른 데이터 세트에 걸쳐 매우 효과적으로 일반화할 수 있는 SENet 아키텍처를 형성할 수 있음을 보여줍니다. 또한 SE 블록이 약간의 추가 계산 비용으로 기존 최첨단 CNN의 성능을 크게 향상시킬 수 있음을 보여줍니다. 스퀴즈-엑세이션 네트워크는 1등을 차지하며 상위 5위 오차를 2.251%로 줄여 2016년 우승작을 약 25%나 능가한 ILSVRC 2017 분류 출품작의 토대를 형성했습니다. 모델과 코드는 https://github.com/hujie-frank/SENet 에서 확인할 수 있습니다.

색인 용어 - 압박과 흥분, 이미지 표현, 주의력, 컨볼루션 신경망.

1 소개

합성곱 신경망(CNN)은 다양한 시각적 작업을 처리하는 데 유용한 모델임이 입증되었습니다 [1], [2], [3], [4]. 네트워크 작업의 각 컨볼루션 레이어에서 필터 모음은 입력 채널을 따라 이웃 공간 연결 패턴을 표현하여 로컬 수신 필드 내에서 공간 및 채널별 정보를 함께 융합합니다. 일련의 컨볼루션 레이어와 비선형 활성화 함수 및 다운샘플링 연산자를 인터리빙함으로써 CNN은 계층적 패턴을 포착하고 글로벌 시지각 수용 필드를 달성하는 이미지 표현을 생성할 수 있습니다. 컴퓨터 비전 연구의 핵심 주제는 주어진 작업에 가장 두드러지는 이미지의 속성만을 포착하여 성능을 향상시킬 수 있는 더 강력한 표현을 찾는 것입니다. 비전 작업에 널리 사용되는 모델 제품군으로서 새로운 신경망 아키텍처 설계의 개발은 이제 이 검색의 핵심 영역이 되었습니다. 최근 연구에 따르면 특징 간의 공간적 상관관계를 포착하는 데 도움이 되는 학습 메커니즘을 네트워크에 통합함으로써 CNN이 생성하는 표현을 강화할 수 있는 것으로 나타났습니다. Inception 아키텍처 제품군[5], [6]에 의해 대중화된 이러한 접근 방식 중 하나는 멀티스케일 프로세스를 네트워크 모듈에 통합하여 향상된 성능을 달성하는 것입니다.

- *지 후와 엔후아 우는 중국 베이징, 100190, 중국 과학원 소프트웨어 연구소 컴퓨터 과학 국가 핵심 연구소에 소속되어 있습니다.*
  또한 중국과학원(중국 베이징, 100049)에도 소속되어 있습니다.
  지 후는 모멘타에서, 엔후아 우는 마카오대학교 과학기술학부 및 AI 센터에서 근무하고 있습니다.
  이메일: hujie@ios.ac.cn ehwu@umac.mo
- *강 선은 중국과학원 자동화연구소의 LIAMA-NLPR에서 근무하고 있습니다. 모멘타에서도 근무하고 있습니다.*
  이메일: sungang@momenta.ai
- *리 셴과 사무엘 알바니는 옥스퍼드 대학교의 시각 기하학 그룹에 소속되어 있습니다.*
  이메일 {lishen,albanie}@robots.ox.ac.uk

mance. 공간적 종속성을 더 잘 모델링하고[7], [8], 네트워크 구조에 공간적 주의를 통합하기 위한 추가 연구가 진행되었습니다[9].

이 백서에서는 네트워크 설계의 또 다른 측면인 채널 간의 관계에 대해 살펴봅니다. 네트워크의 특징 채널 간의 상호의존성을 명시적으로 모델링하여 네트워크가 생성하는 표현의 품질을 개선하기 위해 SE(*Squeeze-and- Excitation)* 블록이라는 새로운 아키텍처 유닛을 도입합니다. 이를 위해 네트워크가 특징 재보정을 수행할 수 있는 메커니즘을 제안하고, 이를 통해 글로벌 정보를 사용하여 유익한 특징을 선택적으로 강조하고 덜 유용한 특징을 억제하는 방법을 학습할 수 있습니다.

SE 빌딩 블록의 구조는 그림 1에 나와 있습니다. 주어진 변환 $\mathbf{F}_{tr}$ 에 대해 입력 $\mathbf{X}$ 를 특징 맵 $\mathbf{U}$ 에 매핑하는 경우, $\mathbf{U} \in \mathbb{R}^{H \times W \times C}$,
컨볼루션과 같은 경우, 특징 재보정을 수행하기 위해 해당 SE 블록을 구성할 수 있습니다. 먼저 특징 **U는** *스퀴즈* 연산을 통과하여 공간 차원($H \times W$)에 걸쳐 특징 맵을 집계하여 채널 설명자를 생성합니다. 이 기술자의 기능은 채널별 특징 응답의 글로벌 분포를 임베딩하여 네트워크의 글로벌 수신 필드의 정보를 모든 레이어에서 사용할 수 있도록 하는 것입니다. 집계 후에는 여기 연산이 이어지며, 여기 연산은 임베딩을 입력으로 받아 채널별 변조 가중치 모음을 생성하는 간단한 셀프 게이팅 메커니즘의 형태를 취합니다. 이러한 가중치는 피처 맵 **U에** 적용되어 네트워크의 후속 레이어에 직접 공급할 수 있는 SE 블록의 출력을 생성합니다.

SE 블록 모음을 쌓는 것만으로 SE 네트워크(SENet)를 구축할 수 있습니다. 또한 이러한 SE 블록은 네트워크 아키텍처의 다양한 깊이 범위에서 원본 블록을 드롭인 방식으로 대체할 수도 있습니다.

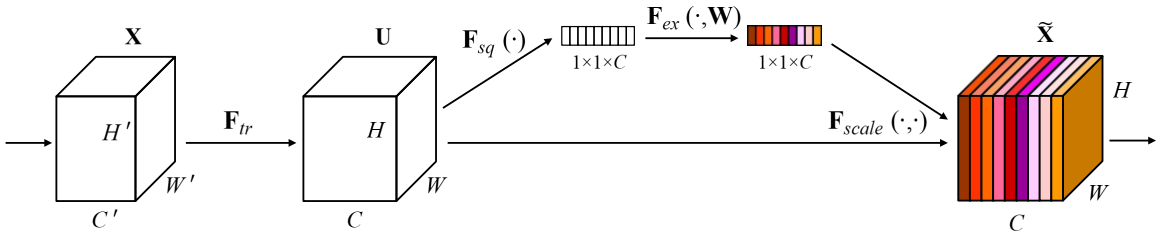


그림 1. 스퀴즈 및 여기 블록.

(섹션 6.4). 빌딩 블록의 템플릿은 일반적이지만, 각기 다른 깊이에서 수행하는 역할은 네트워크 전체에서 다릅니다. 초기 계층에서는 클래스에 구애받지 않는 방식으로 정보 기능을 자극하여 공유된 저수준 표현을 강화합니다. 이후 계층에서는 SE 블록이 점점 더 전문화되어 다양한 입력에 고도로 클래스별 방식으로 반응합니다(섹션 7.2). 결과적으로 SE 블록이 수행하는 특징 재보정의 이점은 네트워크를 통해 축적될 수 있습니다.

새로운 CNN 아키텍처의 설계 및 개발은 일반적으로 많은 새로운 하이퍼파라미터와 레이어 구성을 선택해야 하는 까다로운 엔지니어링 작업입니다. 반면, SE 블록의 구조는 간단하며 기존 최신 아키텍처에서 구성 요소를 SE 블록으로 교체하여 바로 사용할 수 있어 성능을 효과적으로 향상시킬 수 있습니다. 또한 SE 블록은 계산적으로 가볍기 때문에 모델 복잡성과 계산 부담이 약간만 증가합니다.

이러한 주장에 대한 증거를 제시하기 위해 여러 SENet을 개발하고 Ima- geNet 데이터 세트에 대한 광범위한 평가를 수행했습니다[10]. 또한 우리의 접근 방식이 특정 데이터 세트나 작업에 국한되지 않는다는 것을 보여주는 ImageNet 이외의 결과도 제시합니다. SENets을 활용하여 ILSVRC 2017 분류 대회에서 1위를 차지했습니다. 최고의 모델 앙상블은 테스트 세트에서 2.251%의 상위 5위 오류를 달성했습니다.[1]. 이는 전년도 우승작(상위 5위 오차 2.991%)과 비교했을 때 약 25%의 상대적 향상을 나타냅니다.

2 관련 작업

더 깊은 아키텍처. VGGNets [11]과 Inception 모델 [5]은 네트워크의 깊이를 늘리면 학습할 수 있는 표현의 품질이 크게 향상될 수 있음을 보여주었습니다. 각 레이어에 대한 입력의 분포를 조절함으로써 배치 정규화(BN)[6]는 심층 네트워크의 학습 프로세스에 안정성을 더하고 더 매끄러운 최적화 표면을 생성했습니다[12]. 이러한 연구를 바탕으로 ResNets는 신원 기반 스킵 연결[13], [14]을 사용하여 훨씬 더 깊고 강력한 네트워크를 학습할 수 있음을 입증했습니다. 고속도로 네트워크[15]는 지름길 연결을 따라 정보의 흐름을 조절하는 게이팅 메커니즘을 도입했습니다. 이러한 연구에 이어 네트워크 레이어 간의 연결에 대한 추가적인 공식화가 이루어졌습니다[16], [17],

1. http://image-net.org/challenges/LSVRC/2017/results

딥 네트워크의 학습 및 표현 속성에 대한 유망한 개선 사항을 보여줍니다.

이와 밀접하게 관련된 또 다른 연구 분야는 네트워크에 포함된 계산 요소의 기능적 형태를 개선하는 방법에 초점을 맞추고 있습니다. 그룹 컨볼루션은 학습된 변환의 카디널리티를 높이기 위한 인기 있는 접근법으로 입증되었습니다 [18], [19]. 그룹화 연산자의 자연스러운 확장으로 볼 수 있는 다중 분기 컨볼루션[5], [6], [20], [21]을 사용하면 연산자의 보다 유연한 구성을 달성할 수 있습니다. 이전 연구에서 채널 간 상관관계는 일반적으로 공간 구조와 독립적으로[22], [23] 또는 1 × 1 컨볼루션이 있는 표준 컨볼루션 필터[24]를 사용하여 공동으로 새로운 특징 조합으로 매핑됩니다. 이 연구의 대부분은 모델 및 계산 복잡성을 줄이는 목표에 집중되어 있으며, 이는 채널 관계가 국소 수신 필드를 가진 인스턴스 독립적 함수의 구성으로 공식화될 수 있다는 가정을 반영합니다. 이와는 대조적으로, 글로벌 인포메이션을 사용하여 채널 간의 동적 비선형 의존성을 명시적으로 모델링할 수 있는 메커니즘을 장치에 제공하면 학습 프로세스가 쉬워지고 네트워크의 표현력이 크게 향상될 수 있다고 주장합니다.

알고리즘 아키텍처 검색. 위에서 설명한 연구와 함께, 수동 아키텍처 설계를 포기하고 대신 네트워크 구조를 자동으로 학습하는 것을 목표로 하는 풍부한 연구 역사도 있습니다. 이 분야의 초기 연구 대부분은 신경 진화 커뮤니티에서 수행되었으며, 진화적 방법으로 네트워크 토폴로지를 검색하는 방법을 확립했습니다 [25], [26]. 진화적 검색은 종종 계산이 많이 필요하지만, 시퀀스 모델에 적합한 메모리 셀을 찾고[27], [28], 대규모 이미지 분류를 위한 정교한 아키텍처를 학습하는 등 주목할 만한 성공을 거두었습니다[29], [30], [31]. 이러한 방법의 계산 부담을 줄이기 위해 라마키안 상속[32]과 차별적 아키텍처 검색[33]을 기반으로 한 효율적인 대안이 제안되었습니다.

아키텍처 검색을 하이퍼파라미터 최적화로 공식화함으로써 무작위 검색[34] 및 기타 보다 정교한 모델 기반 최적화 기법[35], [36]을 사용하여 문제를 해결할 수도 있습니다. 가능한 설계 패브릭을 통과하는 경로로서의 토폴로지 선택[37] 및 직접 아키텍처 예측[38], [39]이 추가로 실행 가능한 아키텍처 검색 도구로 제안되었습니다. 특히 강화 학습 [40], [41], [42], [43], [44]의 기법을 사용하면 강력한 결과를 얻을 수 있습니다. SE 블록

는 이러한 검색 알고리즘의 원자적 구성 요소로 사용될 수 있으며, 동시 작업에서 이러한 능력이 매우 효과적이라는 것이 입증되었습니다 [45].

주의 및 게이팅 메커니즘. 주의는 사용 가능한 컴퓨팅 자원을 신호의 가장 유익한 구성 요소에 편향적으로 할당하는 수단으로 해석할 수 있습니다 [46], [47], [48], [49], [50], [51]. 주의 메커니즘은 시퀀스 학습 [52], [53], 이미지의 위치 파악 및 이해 [9], [54], 이미지 캡션 등 여러 작업에서 그 유용성이 입증되었습니다.

[55], [56] 및 입술 읽기 [57]. 이러한 애플리케이션에서는 다음 중 하나 이상의 연산자로 통합할 수 있습니다.

레이어는 양식 간 적응을 위한 더 높은 수준의 추상화를 나타냅니다. 일부 연구에서는 공간 주의와 채널 주의의 결합에 대한 흥미로운 연구를 제공합니다 [58], [59]. Wang 등[58]은 심층 잔류 네트워크의 중간 단계 사이에 삽입되는 모래시계 모듈[8]을 기반으로 하는 강력한 트렁크 앤 마스크 주의 메커니즘을 도입했습니다. 이와는 대조적으로, 저희가 제안한 SE 블록은 계산적으로 효율적인 방식으로 채널별 관계를 모델링하여 네트워크의 표현력을 향상시키는 데 중점을 둔 경량 게이팅 메커니즘으로 구성되어 있습니다.

3 스퀴즈 및 흥분 블록

스퀴즈-엑세이션 블록은 입력 $\mathbf{X} \in \mathrm{R}^{H' \times W' \times C'}$를 피처 맵 $\mathbf{U} \in \mathrm{R}^{H \times W \times C}$에 매핑하는 변환 $\mathbf{F}_{tr}$을 기반으로 구축할 수 있는 계산 단위입니다. 다음 표기법에서는 $\mathbf{F}_{tr}$를 컨볼루션 연산자로 간주하고 $\mathbf{V} = [\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_C]$를 사용하여 학습된 필터 커널 집합을 나타내며, 여기서 $\mathbf{v}_c$는 c *번째* 필터의 파라미터를 나타냅니다. 그런 다음 출력을 $\mathbf{U} = [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_C]$로 작성할 수 있습니다.

$$\mathbf{u}_c = \mathbf{v}_c * \mathbf{X} = \sum_{s=1}^{C'} \mathbf{v}_c^s * \mathbf{x}^s. \quad (1)$$

여기서 $*$는 컨볼루션을 나타내고 $\mathbf{v}_c = [\mathbf{v}_c^1, \mathbf{v}_c^2, \ldots, \mathbf{v}_c^{C'}]$, $\mathbf{X} = [\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^{C'}]$ 및 $\mathbf{u}_c \in \mathrm{R}^{H \times W}$. $\mathbf{v}_c^s$는 2D 공간 커널입니다

의 단일 채널을 나타내는 $\mathbf{v}_c$ 표기법을 간소화하기 위해 바이어스 항은 생략합니다. 출력은 모든 채널을 통한 합산으로 생성되므로 채널 종속성은 $\mathbf{v}_c$에 암시적으로 포함되지만 필터가 캡처한 로컬 공간 상관관계와 얽혀 있습니다. 컨볼루션으로 모델링된 채널 관계는 본질적으로 암시적이고 지역적입니다(최상위 레이어에 있는 관계 제외). 채널 상호 의존성을 명시적으로 모델링함으로써 컨볼루션 특징의 학습이 향상되어 네트워크가 후속 변환에서 활용할 수 있는 유익한 특징에 대한 민감도를 높일 수 있을 것으로 기대합니다. 따라서 글로벌 정보에 대한 액세스를 제공하고 필터 응답을 출력 기능. 학습된 각 필터는 로컬 수용 필드에서 작동하므로 변환 출력 **U의** 각 단위는 이 영역 외부의 컨텍스트 정보를 활용할 수 없습니다.

이 문제를 완화하기 위해 글로벌 공간 정보를 채널 설명자에 *압축하는* 방법을 제안합니다. 이는 글로벌 평균 풀링을 사용하여 채널별 통계를 생성함으로써 달성할 수 있습니다. 공식적으로, 통계 $\mathbf{z} \in \mathrm{R}^C$은 공간 차원 $H \times W$를 통해 **U를** 축소하여 생성되며, $\mathbf{z}$의 c번째 요소는 $z_c = \mathbf{F}_{sq}(\mathbf{u}_c) = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} u_c(i, j)$. (2)

토론. 변환 **U의** 출력은 전체 이미지에 대한 통계가 표현되는 로컬 디스크립터의 모음으로 해석할 수 있습니다. 이러한 정보를 활용하는 것은 이전의 특징 엔지니어링 작업에서 널리 사용되었습니다 [60], [61], [62]. 우리는 가장 간단한 집계 기법인 글로벌 평균 풀링을 선택했으며, 여기에서도 더 정교한 전략을 사용할 수 있다는 점에 주목했습니다.

3.2 여기: 적응형 재보정

스퀴즈 연산에서 수집된 정보를 활용하기 위해, 채널별 종속성을 완전히 파악하는 것을 목표로 하는 두 번째 연산이 이어집니다. 이 목표를 달성하기 위해 함수는 두 가지 기준을 충족해야 합니다. 첫째, 유연해야 하며(특히 채널 간의 비선형 상호작용을 학습할 수 있어야 함), 둘째, 한 가지 채널만 강조하는 것이 아니라 여러 채널을 강조할 수 있도록 하려면 상호 배타적이지 않은 관계를 학습해야 합니다. 이러한 기준을 충족하기 위해 시그모이드 활성화와 함께 간단한 게이팅 메커니즘을 사용하기로 결정했습니다:

$$\mathbf{s} = \mathbf{F}_{ex}(\mathbf{z}, \mathbf{W}) = \sigma(g(\mathbf{z}, \mathbf{W})) = \sigma(\mathbf{W}_2 \, \delta(\mathbf{W}_1 \, \mathbf{z})), \quad (3)$$

여기서 δ는 ReLU [63] 함수, $\mathbf{W}_1 \in \mathrm{R}^{\frac{C}{r} \times C}$ 및 $\mathbf{W}_2 \in \mathrm{R}^{C \times \frac{C}{r}}$. 모델 복잡성을 제한하고 일반적인

다음 변환에 적용하기 전에 스퀴즈와 *여기라는* 두 단계로 재보정하고자 합니다. SE 블록의 구조를 보여주는 다이어그램이 그림 1에 나와 있습니다.

3.1 스퀴즈: 글로벌 정보 임베딩

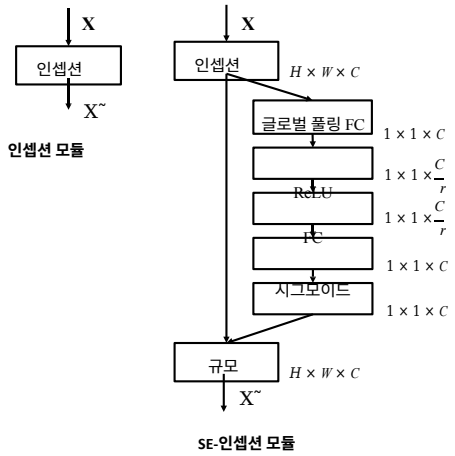
채널 종속성을 악용하는 문제를 해결하기 위해 먼저 각 채널에 대한 신호를 고려합니다.

비선형성 주위에 두 개의 완전 연결(FC) 레이어, 즉 감소 비율 r*을* 갖는 차원 감소 레이어(이 파라미터 선택은 섹션 6.1에서 설명함), ReLU, 그리고 변환 출력 **U의** 채널 차원으로 돌아가는 차원 증가 레이어로 병목 현상을 형성하여 게이팅 메커니즘을 파라미터화합니다. 블록의 최종 출력은 활성화 **s로 U를** 재조정하여 얻게 됩니다:

$$\tilde{\mathbf{x}}_c = \mathbf{F}_{scale}(\mathbf{u}_c, s_c) = s_c \mathbf{u}_c, \tag{4}$$

여기서 $\tilde{\mathbf{X}} = [\tilde{\mathbf{x}}_1, \tilde{\mathbf{x}}_2, \ldots, \tilde{\mathbf{x}}_C]$와 $\mathbf{F}_{scale}(\mathbf{u}_c, s_c)$는 스칼라 s_c 와 피처 맵 $\mathbf{u}_c \in \mathbb{R}^{H \times W}$ 사이의 채널별 곱셈을 나타냅니다.

토론. 여기 연산자는 입력별 기술자 **z를** 채널 가중치 집합에 매핑합니다. 이와 관련하여 SE 블록은 본질적으로 입력에 따른 역학을 도입하며, 이는 컨볼루션 필터가 반응하는 국소 수신 필드에 국한되지 않는 채널에 대한 자기 주의 기능으로 간주할 수 있습니다.



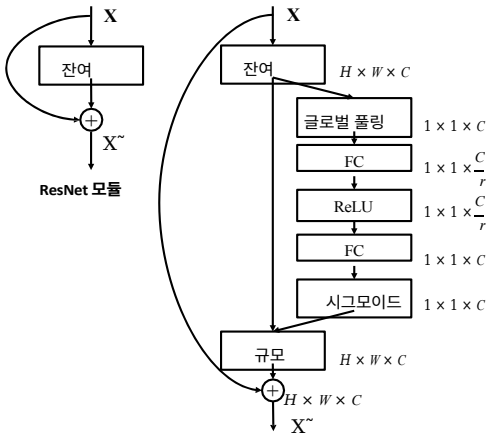


그림 2. 원본 Inception 모듈(왼쪽)과 SE- Inception 모듈(오른쪽)의 스키마.

3.3 인스턴스화

SE 블록은 각 컨볼루션의 비선형성 뒤에 삽입하여 VGGNet [11]과 같은 표준 아키텍처에 통합할 수 있습니다. 또한 SE 블록의 유연성은 표준 컨볼루션을 넘어선 변환에도 직접 적용할 수 있다는 것을 의미합니다. 이 점을 설명하기 위해 아래에 설명하는 몇 가지 더 복잡한 아키텍처의 예에 SE 블록을 통합하여 SENet을 개발했습니다.

먼저 인셉션 네트워크를 위한 SE 블록의 구성을 고려합니다[5]. 여기서는 간단히 변환 $\mathbf{F}_{tr}$ 를 전체 Inception 모듈로 간주하고(그림 2 참조), 아키텍처의 각 모듈에 대해 이러한 변경을 수행하면 *SE-Inception* 네트워크를 얻을 수 있습니다. SE 블록은 잔여 네트워크와 함께 직접 사용할 수도 있습니다(그림 3은 *SE-ResNet* 모듈의 스키마를 보여줍니다). 여기서 SE 블록 변환 $\mathbf{F}_{tr}$ 는 잔여 모듈의 비아이덴티티 분기로 간주됩니다. 스퀴즈와 *여기는* 모두 아이덴티티 브랜치와의 합산 전에 작용합니다. SE 블록을 ResNeXt [19], Inception-ResNet [21], MobileNet [64] 및 ShuffleNet [65]과 통합하는 추가 변형은 유사한 체계를 따라 구성할 수 있습니다. SENet 아키텍처의 구체적인 예는 SE-ResNet-50과 SE-ResNeXt-50에 대한 자세한 설명이 표 1에 나와 있습니다.

SE 블록의 유연한 특성으로 인한 결과 중 하나는 이러한 아키텍처에 통합할 수 있는 몇 가지 실행 가능한 방법이 있다는 것입니다. 따라서 SE 블록을 네트워크 아키텍처에 통합하는 데 사용되는 통합 전략에 대한 민감도를 평가하기 위해 섹션 6.5에서 블록 포함을 위한 다양한 설계를 탐색하는 제거 실험도 제공합니다.

그림 3. 원래 잔여 모듈(왼쪽)과 SE- ResNet 모듈(오른쪽)의 스키마.

스퀴즈 단계와 *여기* 단계의 두 개의 작은 FC 레이어, 그리고 저렴한 채널별 스케일링 작업이 이어집니다. 종합적으로, 감소 비율 *r(*섹션 3.2에 소개됨)을 16으로 설정할 때 SE-ResNet-50에는 다음이 필요합니다.
~3.87 GFLOPs로, 기존 ResNet-50에 비해 0.26% 상대적 증가에 해당합니다. 이러한 약간의 추가 계산 부담을 감수하는 대신, SE-ResNet-50의 정확도는 ResNet-50의 정확도를 능가하며 실제로 ~7.58GFLOPs가 필요한 더 심층적인 ResNet-101 네트워크의 정 확 도 에 근접합니다(표 2).

실제로 ResNet-50을 통해 앞뒤로 한 번 통과하는 데는 190ms가 걸리는 반면, 256개의 이미지로 구성된 트레이닝 미니배치를 사용하는 SE-ResNet-50의 경우 209ms가 걸립니다(두 시간 모두 8개의 NVIDIA Titan X GPU가 탑재된 서버에서 수행됨). 이는 합리적인 런타임 오버헤드를 나타내며, 널리 사용되는 GPU 라이브러리에서 글로벌 풀링과 소규모 내부 제품 작업이 더욱 최적화됨에 따라 더 감소할 수 있습니다. 임베디드 디바이스 애플리케이션에서 중요하기 때문에 각 모델에 대한 CPU 추론 시간을 추가로 벤치마킹한 결과, 224×224픽셀 입력 이미지의 경우 ResNet-50은 164ms가 소요되는 반면 SE-ResNet-50은 167ms가 걸렸습니다. SE 블록으로 인해 발생하는 약간의 추가 계산 비용은 모델 성능에 대한 기여도를 고려할 때 정당화될 수 있다고 생각합니다.

4 모델 및 계산 복잡성

제안된 SE 블록 설계가 실용적으로 사용되려면 성능 향상과 모델 복잡성 증가 사이의 적절한 절충점을 제공해야 합니다. 모듈과 관련된 컴퓨팅 부담을 설명하기 위해 ResNet-50과 SE-ResNet-50의 비교를 예로 들어 보 겠 습 니 다 . ResNet-50은 224 × 224 픽셀 입력 이미지에 대해 단일 포워드 패스에서 ~3.86GFLOPs가 필요합니다. 각 SE 블록은 다음에서 글로벌 평균 풀링 연산을 사용합니다.

다음으로 제안된 SE 블록에 의해 도입된 추가 파라미터를 고려합니다. 이러한 추가 파라미터는 게이팅 메커니즘의 두 FC 레이어에서만 발생하므로 전체 네트워크 용량의 일부에 불과합니다. 구체적으로 이러한 FC 계층의 가중치 매개변수에 의해 도입된 총 개수는 다음과 같습니다:

$$\frac{2}{r}\sum_{s=1}^{S} N_s \cdot C_s^2, \tag{5}$$

여기서 r은 감소율을 나타내고, S는 스테이지 수(스테이지란 공통 공간 차원의 피처 맵에서 작동하는 블록의 집합을 의미), C_s 는 출력 채널의 차원, N_s 은 스테이지 s에 대해 반복되는 블록 수를 나타냅니다(FC 계층에서 바이어스 항을 사용하는 경우, 도입된 파라미터와 계산 비용은 일반적으로 무시할 수 있음). SE-ResNet-50은 다음과 같이 250만 개 이상의 파라미터를 추가로 도입합니다.

표 1

(왼쪽) ResNet-50 [13]. (가운데) SE-ResNet-50. (오른쪽) 32×4d 템플릿이 있는 SE-ResNeXt-50. 잔여 빌딩 블록의 특정 파라미터 설정이 있는 모양과 연산은 괄호 안에 나열되어 있으며 스테이지에 쌓인 블록의 수는 외부에 표시됩니다. 안쪽 괄호 안의 *fc*는 SE 모듈에서 완전히 연결된 두 레이어의 출력 치수를 나타냅니다.

출력 크기	ResNet-50	SE-ResNet-50	SE-ResNeXt-50(32 × 4d)
112 × 112	컨브, 7 × 7, 64, 보폭 2		
56 × 56	최대 풀, 3 × 3, 보폭 2		
	$\begin{bmatrix} conv, 1 \times 1, 64 \\ conv, 3 \times 3, 64 \\ conv, 1 \times 1, 256 \end{bmatrix} \times 3$	$\begin{bmatrix} conv, 1 \times 1, 64 \\ conv, 3 \times 3, 64 \\ conv, 1 \times 1, 256 \\ fc, [16, 256] \end{bmatrix} \times 3$	$\begin{bmatrix} conv, 1 \times 1, 128 \\ conv, 3 \times 3, 128 \quad C = 32 \\ conv, 1 \times 1, 256 \end{bmatrix} \times 3$ $fc, [16, 256]$
28 × 28	$\begin{bmatrix} conv, 1 \times 1, 128 \\ conv, 3 \times 3, 128 \\ conv, 1 \times 1, 512 \end{bmatrix} \times 4$	$\begin{bmatrix} conv, 1 \times 1, 128 \\ conv, 3 \times 3, 128 \\ conv, 1 \times 1, 512 \\ fc, [32, 512] \end{bmatrix} \times 4$	$\begin{bmatrix} conv, 1 \times 1, 256 \\ conv, 3 \times 3, 256 \quad C = 32 \\ conv, 1 \times 1, 512 \end{bmatrix} \times 4$ $fc, [32, 512]$
14 × 14	$\begin{bmatrix} conv, 1 \times 1, 256 \\ conv, 3 \times 3, 256 \\ conv, 1 \times 1, 1024 \end{bmatrix} \times 6$	$\begin{bmatrix} conv, 1 \times 1, 256 \\ conv, 3 \times 3, 256 \\ conv, 1 \times 1, 1024 \\ fc, [64, 1024] \end{bmatrix} \times 6$	$\begin{bmatrix} conv, 1 \times 1, 512 \\ conv, 3 \times 3, 512 \quad C = 32 \\ conv, 1 \times 1, 1024 \end{bmatrix} \times 6$ $fc, [64, 1024]$
7×7	$\begin{bmatrix} conv, 1 \times 1, 512 \\ conv, 3 \times 3, 512 \\ conv, 1 \times 1, 2048 \end{bmatrix} \times 3$	$\begin{bmatrix} conv, 1 \times 1, 512 \\ conv, 3 \times 3, 512 \\ conv, 1 \times 1, 2048 \\ fc, [128, 2048] \end{bmatrix} \times 3$	$\begin{bmatrix} conv, 1 \times 1, 1024 \\ conv, 3 \times 3, 1024 \quad C = 32 \\ conv, 1 \times 1, 2048 \\ fc, [128, 2048] \end{bmatrix} \times 3$
1 × 1	글로벌 평균 풀, 1000-D *FC*, 소프트맥스		

표 2

이미지넷 검증 세트의 단일 크롭 오류율(%) 및 복잡도 비교. *원본* 열은 원본 논문에서 보고된 결과를 나타냅니다(ResNets의 결과는 웹사이트 (https://github.com/Kaiminghe/deep-residual-networks)에서 확인할 수 있습니다). 공정한 비교를 위해 기준 모델을 재훈련하고 *재구현* 열에 점수를 보고합니다. *SENet* 열은 SE 블록이 추가된 해당 아키텍처를 나타냅니다. 괄호 안의 숫자는 기준 아키텍처보다 향상된 성능을 나타냅니다.

†다시 구현된 기준선. 는 모델이 유효성 검사 집합의 블랙리스트에 포함되지 않은 하위 집합에서 평가되었음을 나타내며(이에 대해서는 [21]에서 자세히 설명함), 이는 결과를 약간 개선할 수 있습니다. VGG-16과 SE-VGG-16은 일괄 정규화를 통해 훈련됩니다.

	원본		재구현			SENet		
	TOP-1 ERR.	상위 5위 오류.	TOP-1 ERR.	상위 5위 오류.	GFLOPs	TOP-1 ERR.	상위 5위 오류.	GFLOPs
ResNet-50 [13]	24.7	7.8	24.80	7.48	3.86	$23.29_{(1.51)}$	$6.62_{(0.86)}$	3.87
ResNet-101 [13]	23.6	7.1	23.17	6.52	7.58	$22.38_{(0.79)}$	$6.07_{(0.45)}$	7.60
ResNet-152 [13]	23.0	6.7	22.42	6.34	11.30	$21.57_{(0.85)}$	$5.73_{(0.61)}$	11.32
ResNeXt-50 [19]	22.2	-	22.11	5.90	4.24	$21.10_{(1.01)}$	$5.49_{(0.41)}$	4.25
ResNeXt-101 [19]	21.2	5.6	21.18	5.57	7.99	$20.70_{(0.48)}$	$5.01_{(0.56)}$	8.00
VGG-16 [11]	-	-	27.02	8.81	15.47	$25.22_{(1.80)}$	$7.70_{(1.11)}$	15.48
BN-인셉션 [6]	25.2	7.82	25.38	7.89	2.03	$24.23_{(1.15)}$	$7.14_{(0.75)}$	2.04
인셉션-ResNet-v2 [21]	19.9†	4.9†	20.37	5.21	11.75	$19.80_{(0.57)}$	$4.79_{(0.42)}$	11.76

~ResNet-50에 필요한 파라미터는 약 2,500만 개이며, 이는 약 10% 증가한 수치입니다. 실제로 이러한 파라미터의 대부분은 가장 많은 수의 채널에서 여기 연산이 수행되는 네트워크의 최종 단계에서 발생합니다. 그러나 상대적으로 비용이 많이 드는 이 SE 블록의 최종 단계를 약간의 성능 저하(이미지넷의 상위 5위 오차 0.1% *미만*)만으로 제거하면 상대적인 파라미터 증가를 ~4%로 줄일 수 있으며, 이는 파라미터 사용이 주요 고려 사항인 경우에 유용할 수 있습니다(자세한 내용은 6.4 및 7.2절 참조).

5 실험

이 섹션에서는 다양한 작업, 데이터 세트 및 모델 아키텍처에서 SE 블록의 효과를 조사하기 위한 실험을 수행합니다.

5.1 이미지 분류

SE 블록의 영향을 평가하기 위해 먼저 128만 *개의* 훈련 이미지와 5만 개의 검증 이미지로 구성된 ImageNet 2012 데이터 세트[10]에서 실험을 수행합니다.

의 이미지로 구성됩니다. 훈련 세트에서 네트워크를 훈련하고 검증 세트에서 상위 1, 상위 5개의 오류를 보고합니다.

각 기준 네트워크 아키텍처와 그에 상응하는 SE는 동일한 최적화 체계로 훈련됩니다. 우리는 표준 관행을 따르고 스케일과 종횡비[5]를 사용하여 224×224픽셀 크기(Inception-ResNet-v2[21] 및 SE-Inception-ResNet-v2의 경우 299×299)로 무작위 자르기를 통해 데이터 증강을 수행하며 무작위 수평 뒤집기를 수행합니다. 각 입력 이미지는 평균 RGB 채널 차감을 통해 정규화됩니다. 모든 모델은 대규모 네트워크의 효율적인 병렬 학습을 처리하도록 설계된 분산 학습 시스템 *ROCS에서* 학습됩니다. 최적화는 모멘텀 0.9, 미니배치 크기 1024의 동기식 SGD를 사용하여 수행됩니다. 초기 학습률은 0.6으로 설정되어 있으며 30회 에포크마다 10씩 감소합니다. 모델은 [66]에 설명된 가중치 초기화 전략을 사용하여 처음부터 100개의 에포크 동안 훈련됩니다. *(*섹션 3.2의) 감소 비율 r*은* 기본적으로 16으로 설정됩니다(달리 명시된 경우 제외).

모델을 평가할 때 중앙 자르기를 적용하여 각 이미지에서 224 × 224 픽셀이 잘리도록 한 다음 다음과 같이 합니다.

표 3

이미지넷 검증 세트의 단일 크롭 오류율(%) 및 복잡도 비교. MobileNet은 [64]의 "1.0 MobileNet-224"를, ShuffleNet은 [65]의 "ShuffleNet $1 \times (g = 3)$"을 참조합니다. 괄호 안의 숫자는 재구현에 따른 성능 향상을 나타냅니다.

	원본		재구현				SENet			
	TOP-1 ERR.	상위 5위 오류.	TOP-1 ERR.	상위 5위 오류.	MFLOPs	매개변수	TOP-1 ERR.	상위 5위 오류.	MFLOPs	매개변수
모바일넷 [64]	29.4	-	28.4	9.4	569	4.2M	$25.3_{(3.1)}$	$7.7_{(1.7)}$	572	4.7M
셔플넷 [65]	32.6	-	32.6	12.5	140	1.8M	$31.0_{(1.6)}$	$11.1_{(1.4)}$	142	2.4M

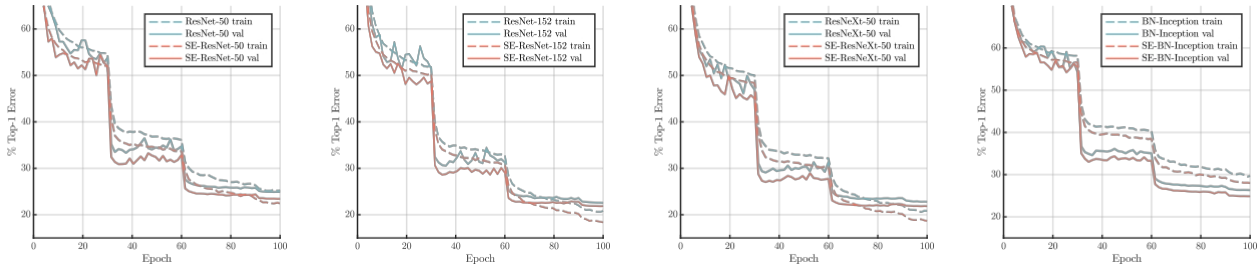


그림 4. 이미지넷에서 베이스라인 아키텍처와 해당 SENet 아키텍처를 트레이닝하는 모습. SENet은 향상된 최적화 특성을 나타내며 트레이닝 프로세스 전반에 걸쳐 일관된 성능 향상을 제공합니다.

의 짧은 에지는 먼저 256으로 크기가 조정됩니다 (Inception-ResNet-v2 및 SE-Inception-ResNet-v2의 경우 각 이미지에서 299 × 299의 짧은 에지가 먼저 352로 크기가 조정됨).

네트워크 깊이. 먼저 SE-ResNet을 다양한 깊이의 ResNet 아키텍처와 비교하여 그 결과를 표 2에 정리해 보았습니다. SE 블록은 계산 복잡성이 매우 적게 증가하면서 다양한 깊이에 걸쳐 일관되게 성능이 향상되는 것을 관찰할 수 있습니다. 다시 말해, SE-ResNet-50은 6.62%의 단일 크롭 상위 5개의 유효성 *검사* 오류를 달성하여 ResNet-50(7.48%)을 0.86% 초과하고, 전체 계산 부담의 절반(3.87GFLOPs 대 7.58GFLOPs)만으로 훨씬 더 깊은 ResNet-101 네트워크(6.52% 상위 5개의 오류)의 성능에 근접합니다. 이 패턴은 더 깊은 곳에서도 반복되는데, SE-ResNet-101(상위 5% 오차 6.07%)은 더 깊은 곳의 ResNet-152 네트워크(상위 5% *오차* 6.34%)와 일치할 뿐만 아니라 0.27% *더* 나은 성능을 보입니다. SE 블록 자체가 깊이를 추가하지만, 이는 매우 계산적으로 효율적인 방식으로 이루어지며, 기본 아키텍처의 깊이를 확장하면 수익률이 감소하는 시점에서도 좋은 수익률을 제공합니다. 또한, 다양한 네트워크 깊이 범위에서 이득이 일관되게 나타나는 것을 볼 수 있는데, 이는 SE 블록이 유도하는 개선이 단순히 기본 아키텍처의 깊이를 증가시킴으로써 얻을 수 있는 개선과 상호보완적일 수 있음을 시사합니다.

최신 아키텍처와의 통합. 다음으로 SE 블록을 두 개의 최신 아키텍처인 Inception-ResNet-v2 [21] 및 ResNeXt(32 × 4d 설정 사용) [19]와 통합하는 효과를 연구하는데, 이 두 아키텍처는 모두 기본 네트워크에 추가 계산 빌딩 블록을 도입합니다. 이러한 네트워크의 SENet 등가물인 SE-Inception-ResNet-v2 및 SE-ResNeXt(SE-ResNeXt-50의 구성은 표 1에 나와 있음)를 구축하고 표 2에 결과를 보고합니다. 이전 실험과 마찬가지로, 두 아키텍처 모두에 SE 블록을 도입함으로써 상당한 성능 향상이 관찰되었습니다. 특히, SE-ResNeXt-50의 상위 5위 오차는 5.49%로 직접적으로 대응하는 ResNeXt-50(5.90%)보다 우수합니다.

상위 5퍼센트 오차)와 더 심층적인 모델인 ResNeXt-101(상위 5퍼센트 오차 5.57%)을 비교했는데, 이 모델은 총 매개변수 수와 계산 오버헤드가 거의 두 배에 달합니다. Inception-ResNet-v2를 다시 구현한 결과와 [21]에서 보고된 결과 사이에 약간의 성능 차이가 있음을 알 수 있습니다. 그러나 SE 블록의 효과와 관련해서는 비슷한 추세를 관찰할 수 있었으며, SE 블록(4.79% 상위 5% 오차)이 재구현된 Inception-ResNet-v2 기준선(5.21% 상위 5% 오차)과 [21]에서 보고된 결과보다 0.42% *더* 나은 성능을 보였습니다.

또한, *비잔류* 네트워크에서 작동할 때 SE 블록의 효과를 평가하기 위해 VGG-16 [11] 및 BN-Inception 아키텍처 [6]로 실험을 수행합니다. VGG-16을 처음부터 쉽게 훈련할 수 있도록 각 컨볼루션 후에 배치 정규화 레이어를 추가합니다. VGG-16과 SE-VGG-16 모두에 대해 동일한 훈련 방식을 사용합니다. 비교 결과는 표 2에 나와 있습니다. 잔여 기준 아키텍처에 대해 보고된 결과와 유사하게, SE 블록이 비잔여 설정에서 성능 향상을 가져오는 것을 관찰할 수 있습니다.

이러한 모델의 최적화에 대한 SE 블록의 영향에 대한 인사이트를 제공하기 위해 기준 아키텍처와 해당 SE 아키텍처의 실행에 대한 훈련 곡선 예시를 그림 4에 표시했습니다. SE 블록은 최적화 절차 전반에 걸쳐 꾸준히 개선되는 것을 관찰할 수 있습니다. 또한 이러한 추세는 기준선으로 간주되는 다양한 네트워크 아키텍처에서 상당히 일관되게 나타납니다.

모바일 설정. 마지막으로, 모바일에 최적화된 네트워크 클래스의 대표적인 두 가지 아키텍처인 MobileNet [64]과 ShuffleNet [65]을 고려합니다. 이 실험에서는 256개의 미니배치 크기와 [65]에서와 같이 약간 덜 공격적인 데이터 증강 및 정규화를 사용했습니다. 모멘텀(0.9로 설정)과 초기 학습률 0.1의 SGD를 사용하여 8개의 GPU에서 모델을 훈련했으며, 검증 손실이 정체될 때마다 10배씩 감소시켰습니다. 총 훈련 과정에는 400 에포크가 필요했습니다([65]의 기준 성능을 재현할 수 있음). 표 3에 보고된 결과는 SE 블록이 지속적으로 개선되었음을 보여줍니다.

표 4
CIFAR-10의 분류 오류(%).

	원본	SENet
ResNet-110 [14]	6.37	5.21
ResNet-164 [14]	5.46	4.39
WRN-16-8 [67]	4.27	3.88
쉐이크셰이크 26 2x96d [68] + 컷아웃 [69]	2.56	2.12

표 5
CIFAR-100의 분류 오류(%).

	원본	SENet
ResNet-110 [14]	26.88	23.85
ResNet-164 [14]	24.33	21.31
WRN-16-8 [67]	20.43	19.14
쉐이크-이븐 29 2x4x64d [68] + 컷아웃 [69]	15.85	15.41

최소한의 계산 비용 증가로 정확도를 큰 폭으로 향상시킬 수 있습니다.

추가 데이터 세트. 다음으로 SE 블록의 적합성이 ImageNet 이외의 데이터 세트에도 일반화되는지 여부를 조사합니다. 몇 가지 널리 사용되는 기준 아키텍처와 기법(ResNet-110 [14], ResNet-164 [14], WideResNet-16-8 [67], Shake-Shake [68], Cutout [69])으로 CIFAR-10 및 CIFAR-100 데이터 세트 [70]에 대한 실험을 수행합니다. 이는 각각 10개와 100개의 클래스로 레이블이 지정된 50만 개의 훈련 및 1만 개의 테스트 32×32픽셀 RGB 이미지 모음으로 구성됩니다. 이러한 네트워크에 SE 블록을 통합하는 것은 3.3절에서 설명한 것과 동일한 접근 방식을 따릅니다. 각 기준선과 해당 SENet은 표준 데이터 증강 전략으로 훈련됩니다 [24], [71]. 훈련 중에 이미지는 무작위로 수평으로 뒤집히고 각 면에 4픽셀씩 제로 패딩된 후 무작위로 32×32 자르기를 수행합니다. 평균 및 표준편차 정규화도 적용됩니다. 훈련 하이퍼파라미터의 설정(예: 미니배치 크기, 초기 학습 속도, 가중치 감쇠)은 원본 논문에서 제안한 것과 일치합니다. 표 4에서는 각 기준선과 해당 SENet의 CIFAR-10에 대한 성능을, 표 5에서는 CIFAR-100에 대한 성능을 보고합니다. 모든 비교에서 SENet이 기준 아키텍처를 능가하는 것으로 나타났는데, 이는 SE 블록의 이점이 ImageNet 데이터 세트에만 국한되지 않음을 시사합니다.

5.2 장면 분류

또한 장면 분류를 위해 Places365-Challenge 데이터 세트 [73]에서 실험을 진행했습니다. 이 데이터 세트는 다음과 같이 구성됩니다.

8백만 개의 훈련 이미지와 365개 카테고리에 걸친 3만 6,500개의 검증 이미지. 분류에 비해 장면 이해 작업은 일반화를 잘하고 추상화를 처리하는 모델의 능력을 평가하는 또 다른 방법입니다. 이는 종종 모델이 더 복잡한 데이터 연결을 처리하고 더 큰 수준의 외관 변화에 대해 견고해야 하기 때문입니다.

저희는 SE 블록의 효과를 평가하기 위한 강력한 기준으로 ResNet-152를 사용하고 [72], [74]에 설명된 훈련 및 평가 프로토콜을 따르기로 결정했습니다. 이 실험에서는 모델을 처음부터 학습시켰습니다. 그 결과는 표 6에서 확인할 수 있으며, 이전 연구와도 비교됩니다. SE-ResNet-152(11.01% 상위 5% 오류)가 ResNet-152(11.61% 상위 5% 오류)보다 낮은 검증 오류를 달성하는 것을 관찰할 수 있습니다,

표 6
Places365 유효성 검사 세트의 단일 자르기 오류율(%).

	TOP-1 ERR.	상위 5위 오류.
장소-365-CNN [72]	41.07	11.48
ResNet-152(당사)	41.15	11.61
SE-ResNet-152	**40.37**	**11.01**

표 7
COCO *미니벌* 세트에서 더 빠른 R-CNN 객체 감지 결과(%).

	AP@IoU=0.5	AP
ResNet-50	57.9	38.0
SE-ResNet-50	61.0	40.4
ResNet-101	60.1	39.9
SE-ResNet-101	62.7	41.9

SE 블록이 장면 분류를 개선할 수 있다는 증거를 제공합니다. 이 SENet은 이 작업에서 상위 *5위* 오차가 11.48%에 달하는 이전의 최신 모델인 Places-365-CNN[72]을 능가합니다.

5.3 COCO에서 물체 감지

또한 COCO 데이터 세트 [75]를 사용하여 물체 감지 작업에 대한 SE 블록의 일반화를 평가합니다. 이전 작업[19]에서와 마찬가지로 *미니벌* 프로토콜, 즉 80만 개의 훈련 세트와 3만 5천 개의 가치 하위 집합의 조합으로 모델을 훈련하고 나머지 5천 개의 가치 하위 집합으로 평가하는 방식을 사용합니다. 가중치는 ImageNet 데이터 세트에서 훈련된 모델의 파라미터로 초기화됩니다. 우리는 모델 평가의 기초로 Faster R-CNN [4] 탐지 프레임워크를 사용하고 [76]에 설명된 하이퍼파라미터 설정을 따릅니다(즉, '2배' 학습 스케줄로 엔드투엔드 트레이닝). 우리의 목표는 물체 감지기의 트렁크 아키텍처(ResNet)를 SE-ResNet으로 대체했을 때의 효과를 평가하여 천공률의 변화가 더 나은 표현에 기인할 수 있도록 하는 것입니다. 표 7은 ResNet-50, ResNet-101 및 이에 상응하는 SE를 트렁크 아키텍처로 사용하는 객체 감지기의 검증 세트 성능을 보여줍니다. SE-ResNet-50은 COCO의 표준 AP 메트릭에서 2.4%(상대적 6.3% 개선), AP@IoU=0.5 에서 3.1% *더* 나은 성능을 보였습니다. SE 블록은 또한 더 심층적인 ResNet-101 아키텍처의 이점을 활용하여 AP 지표에서 2.0% 개선(상대적 5.0% 개선)을 달성했습니다. 요약하면, 이 일련의 실험은 SE 블록의 일반화 가능성을 보여줍니다. 유도된 개선은 광범위한 아키텍처, 작업 및 데이터 세트에서 실현될 수 있습니다.

5.4 ILSVRC 2017 분류 경쟁

SENet은 ILSVRC 대회에 제출하여 1등을 차지할 수 있었던 기반이 되었습니다. 우승작은 테스트 세트에서 2.251%의 상위 5위 *오차를* 얻기 위해 표준 멀티스케일 및 멀티크롭 융합 전략을 채택한 소규모 SENet 앙상블로 구성되었습니다. 이번 제출의 일환으로, 저희는 SE 블록을 수정된 ResNeXt [19]와 통합하여 추가 모델인 *SENet-154를* 구축했습니다(아키텍처에 대한 자세한 내용은 부록에 제공됨). 이 모델을 표준 크롭을 사용하여 표 8의 ImageNet 검증 세트에 대한 이전 작업과 비교합니다.

표 8
크롭 크기가 224 × 224 및 320 × 320 / 299 × 299인 ImageNet 유효성 검사 세트의 최신 CNN의 단일 크롭 오류율(%).

	224 × 224		320 × 320 / 299 × 299	
	top-1 err.	상위 5위 err.	top-1 err.	상위 5위 err.
ResNet-152 [13]	23.0	6.7	21.3	5.5
ResNet-200 [14]	21.7	5.8	20.1	4.8
인셉션-v3 [20]	-	-	21.2	5.6
인셉션-v4 [21]	-	-	20.0	5.0
인셉션-ResNet-v2 [21]	-	-	19.9	4.9
ResNeXt-101(64 × 4d) [19] [19]	20.4	5.3	19.1	4.4
DenseNet-264 [17]	22.15	6.12	-	-
주의-92 [58]	-	-	19.5	4.8
피라미드넷-200 [77]	20.1	5.4	19.2	4.7
DPN-131 [16]	19.93	5.12	18.55	4.16
SENet-154	**18.68**	**4.47**	**17.28**	**3.79**

표 9
더 큰 크롭 크기/추가 훈련 데이터를 사용한 ImageNet 검증 세트의 최신 CNN과의 비교(%). †모델은 320 × 320의 크롭 크기로 훈련되었습니다.

	추가 데이터	자르기 크기	TOP-1 ERR.	상위 5위 오류.
매우 깊은 폴리넷 [78]	-	331	18.71	4.25
NASNet-A(6 @ 4032) [42] [42]	-	331	17.3	3.8
PNASNet-5(N=4,F=216) [35] [35]	-	331	17.1	3.8
SENet-154†	-	320	16.88	3.58
아메바넷-C [79]	-	331	16.5	3.5
ResNeXt-101 32 × 48d [80]	C	224	14.6	2.4

크기(224 × 224 및 320 × 320). 보고된 가장 강력한 결과인 224 × 224 중심 작물 평가에서 SENet-154는 상위 1% *오차* 18.68%, 상위 5% 오차 4.47%를 달성한 것으로 나타났습니다.

챌린지 이후 이미지넷 벤치마크는 많은 진전이 있었습니다. 비교를 위해 현재 알려진 가장 강력한 결과를 표 9에 포함시켰습니다. 이미지넷 데이터만을 사용한 최고의 성능은 최근 [79]에 의해 보고되었습니다. 이 방법은 강화 학습을 사용하여 훈련 중에 데이터 보강을 위한 새로운 정책을 개발하여 [31]에서 검색한 아키텍처의 성능을 향상시킵니다. 전반적으로 가장 우수한 성능은 [80]에서 ResNeXt-101 *32×48d* 아키텍처를 사용하여 보고했습니다. 이는 약 10억 개의 라벨이 약한 이미지에 대해 모델을 사전 훈련하고 ImageNet에서 미세 조정을 통해 달성했습니다. 보다 정교한 데이터 증강[79]과 광범위한 사전 훈련[80]을 통해 얻은 개선 사항은 네트워크 아키텍처에 대한 제안된 변경 사항을 보완할 수 있습니다.

6 수술 연구

이 섹션에서는 SE 블록의 구성 요소에 대해 서로 다른 구성을 사용하는 것이 미치는 영향을 더 잘 이해하기 위해 제거 실험을 수행합니다. 모든 제거 실험은 단일 머신(GPU 8개)의 ImageNet 데이터 세트에서 수행됩니다. ResNet-50이 백본 아키텍처로 사용됩니다. ResNet 아키텍처에서 여기 작업에서 FC 레이어의 편향을 제거하면 채널 종속성 모델링이 용이해진다는 것을 경험적으로 확인했으며, 다음에서 이 구성을 사용합니다.

표 10
다양한 축소 비율에서 이미지넷의 단일 크롭 오류율(%) 및 SE-ResNet-50의 매개변수 크기입니다. 여기서 *원본은* ResNet-50을 의미합니다.

비율 r	TOP-1 ERR.	상위 5위 오류.	매개변수
2	22.29	6.00	45.7M
4	22.25	6.09	35.7M
8	22.26	5.99	30.7M
16	22.28	6.03	28.1M
32	22.72	6.20	26.9M
원본	23.30	6.55	25.6M

실험. 데이터 증강 전략은 섹션 5.1에 설명된 접근 방식을 따릅니다. 각 변형에 대한 성능의 상한을 연구할 수 있도록 학습 속도를 0.1로 초기화하고 검증 손실이 정점에 도달할 때까지 학습을 계속합니다.[2] (총 ~300 에포크)까지 학습을 계속합니다. 그런 다음 학습 속도를 10배로 낮춘 다음 이 과정을 반복합니다(총 3회). 훈련 중에는 라벨 평활 정규화 [20]가 사용됩니다.

6.1 감소 비율

방정식 5에 도입된 감소율 r*은* 네트워크에서 SE 블록의 용량과 컴퓨팅 비용을 변화시킬 수 있는 하이퍼파라미터입니다. 이 하이퍼파라미터에 의해 매개되는 성능과 컴퓨팅 비용 간의 트레이드 오프를 조사하기 위해 다양한 범위의 r 값에 대해 SE-ResNet-50으로 실험을 수행했습니다. 표 10의 비교를 보면 다양한 감소율에 대해 성능이 견고하다는 것을 알 수 있습니다. 복잡성이 증가해도 성능이 단조롭게 개선되지 않는 반면, 감소율이 작아지면 모델의 파라미터 크기가 급격히 증가합니다. $r = 16$으로 설정하면 정확도와 복잡성 간에 좋은 균형을 이룰 수 있습니다. 실제로는 네트워크 전체에 동일한 비율을 사용하는 것이 최적이 아닐 수 있으므로(각 레이어가 수행하는 역할이 다르기 때문에), 주어진 기본 아키텍처의 요구 사항에 맞게 비율을 조정하여 추가 개선을 달성할 수 있습니다.

6.2 스퀴즈 연산자

스퀴즈 연산자 선택 시 글로벌 최대 풀링이 아닌 글로벌 평균 풀링을 사용하는 것이 얼마나 중요한지 살펴봅니다(이 방법이 잘 작동했기 때문에 더 정교한 대안은 고려하지 않았습니다). 결과는 표 11에 나와 있습니다. 최대 풀링과 평균 풀링 모두 효과적이지만, 평균 풀링이 약간 더 나은 성능을 달성하여 스퀴즈 연산의 기초로 선택된 것을 정당화합니다. 그러나 SE 블록의 성능은 특정 집계 연산자의 선택에 따라 상당히 견고하다는 점에 유의해야 합니다.

6.3 여기 연산자

다음으로 여기 메커니즘에 대한 비선형성 선택을 평가합니다. 두 가지 옵션을 추가로 고려합니다: ReLU와 tanh를 고려하고, 시그모이드를 다음과 같이 대체하는 실험을 해봅니다.

2. 참고로 270에포크 고정 스케줄(125, 200, 250에포크에서 학습률을 낮춤)로 훈련할 경우 ResNet-50과 *SE-ResNet-50의* 상위 1% 및 상위 5% 오류율은 각각 (23.21%, 6.53%), (22.20%, 6.00%)에 달합니다.

표 11
SE-ResNet-50에서 다양한 스퀴즈 연산자를 사용하는 것이 ImageNet에 미치는 영향(오류율 %).

스퀴즈	TOP-1 ERR.	상위 5위 오류.
최대	22.57	6.09
평균	**22.28**	**6.03**

표 12
SE-ResNet-50에서 여기 연산자에 대해 서로 다른 비선형성을 사용한 결과(오류율 %)가 ImageNet에 미치는 영향.

흥분	TOP-1 ERR.	상위 5위 오류.
ReLU	23.47	6.98
Tanh	23.00	6.38
시그모이드	**22.28**	**6.03**

대체 비선형성. 결과는 표 12에 나와 있습니다. 시그모이드를 tanh로 교체하면 성능이 약간 악화되는 반면, ReLU를 사용하면 성능이 크게 악화되어 실제로 SE-ResNet-50의 성능이 ResNet-50 기준선 아래로 떨어지는 것을 볼 수 있습니다. 이는 SE 블록이 효과적이기 위해서는 여기 연산자를 신중하게 구성하는 것이 중요하다는 것을 시사합니다.

6.4 다양한 단계

한 번에 한 단계씩 SE 블록을 ResNet-50에 통합하여 각 단계별 SE 블록의 영향을 살펴봅니다. 구체적으로 중간 단계인_ 2단계,_ 3단계,_ 4단계에 SE 블록을 추가하고 그 결과를 표 13에 보고합니다. SE 블록을 아키텍처의 각 단계에 도입하면 성능 이점을 얻을 수 있음을 관찰했습니다. 또한, 서로 다른 단계에서 SE 블록에 의해 유도되는 이득은 네트워크 성능을 더욱 강화하기 위해 효과적으로 결합될 수 있다는 점에서 상호 보완적입니다.

6.5 통합 전략

마지막으로, SE 블록을 기존 아키텍처에 통합할 때 SE 블록의 위치가 미치는 영향을 평가하기 위해 제거 연구를 수행합니다. 제안된 SE 설계 외에도 (1) SE 블록이 잔여 유닛보다 먼저 이동하는 SE-PRE 블록, (2) SE 유닛이 아이덴티티 브랜치와의 합산 이후(ReLU 이후) 이동하는 SE-POST 블록, (3) SE 유닛이 잔여 유닛과 병렬로 아이덴티티 연결에 배치되는 SE-Identity 블록 등 3가지 변형을 고려합니다. 이러한 변형은 그림 5에 설명되어 있으며 각 변형의 성능은 표 14에 나와 있습니다. SE-PRE, SE-Identity, 제안된 SE 블록의 성능은 각각 비슷하게 우수하지만, 사용량은 다음과 같습니다.

표 13
이미지넷의 여러 단계에서 SE 블록을 ResNet-50과 통합했을 때의 효과(오류율 %).

무대	TOP-1 ERR.	상위 5위 오류.	GFLOPs	매개변수
ResNet-50	23.30	6.55	3.86	25.6M
SE_2단계 _	23.03	6.48	3.86	25.6M
SE_3단계 _	23.04	6.32	3.86	25.7M
SE 4단계	22.68	6.22	3.86	26.4M
SE 모두	22.28	6.03	3.87	28.1M

표 14
ResNet-50을 사용한 다양한 SE 블록 통합 전략이 ImageNet에 미치는 영향(오류율 %).

디자인	TOP-1 ERR.	상위 5위 오류.
SE	22.28	6.03
SE-PRE	22.23	6.00
SE-POST	22.78	6.35
SE-아이덴티티	22.20	6.15

표 15
ResNet-50의 각 잔여 분기의 3x3 컨볼루션 레이어에서 SE 블록을 통합하는 것이 ImageNet에 미치는 영향(오류율 %).

디자인	TOP-1 ERR.	상위 5위 오류.	GFLOPs	매개변수
SE	22.28	6.03	3.87	28.1M
SE 3×3	22.48	6.02	3.86	25.8M

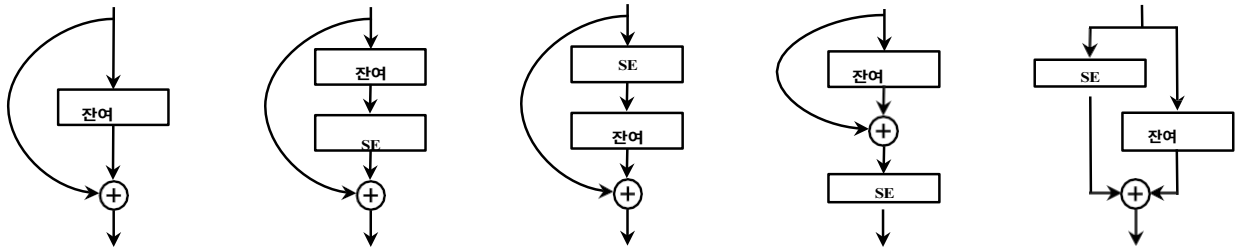
을 초과하면 성능이 저하됩니다. 이 실험은 SE 유닛에 의한 성능 향상이 브랜치 집계 전에 적용된다면 그 위치에 따라 상당히 견고하다는 것을 시사합니다. 위의 실험에서 각 SE 블록은 잔여 유닛의 구조 외부에 배치되었습니다. 또한 SE 블록을 잔여 유닛 내부로 이동시켜 3×3 컨볼루션 레이어 바로 뒤에 배치하는 설계의 변형도 구축했습니다. 3×3 컨볼루션 레이어는 채널 수가 적기 때문에 해당 SE 블록에 의해 도입되는 파라미터의 수도 줄어듭니다. 표 15의 비교는 SE_3×3 변형이 표준 SE 블록보다 더 적은 수의 파라미터로 비슷한 수준의 분류 정확도를 달성한다는 것을 보여줍니다. 이 작업의 범위를 벗어나지만, 다음과 같이 추가적인 효율성 향상을 달성할 수 있을 것으로 예상됩니다. 특정 아키텍처에 맞게 SE 블록 사용량을 맞춤 설정하여

7 SE 블록의 역할

제안된 SE 블록은 여러 시각 작업에서 네트워크 성능을 입증하는 것으로 나타났지만, 우리는 스퀴즈 작업의 상대적 중요성과 여기 메커니즘이 실제로 어떻게 작동하는지 이해하고자 합니다. 심층 신경망에 의해 학습된 표현에 대한 엄격한 이론적 분석은 여전히 어려운 과제이므로, 우리는 실제 기능에 대한 원론적인 이해를 목표로 SE 블록이 수행하는 역할을 조사하는 경험적 접근 방식을 취합니다.

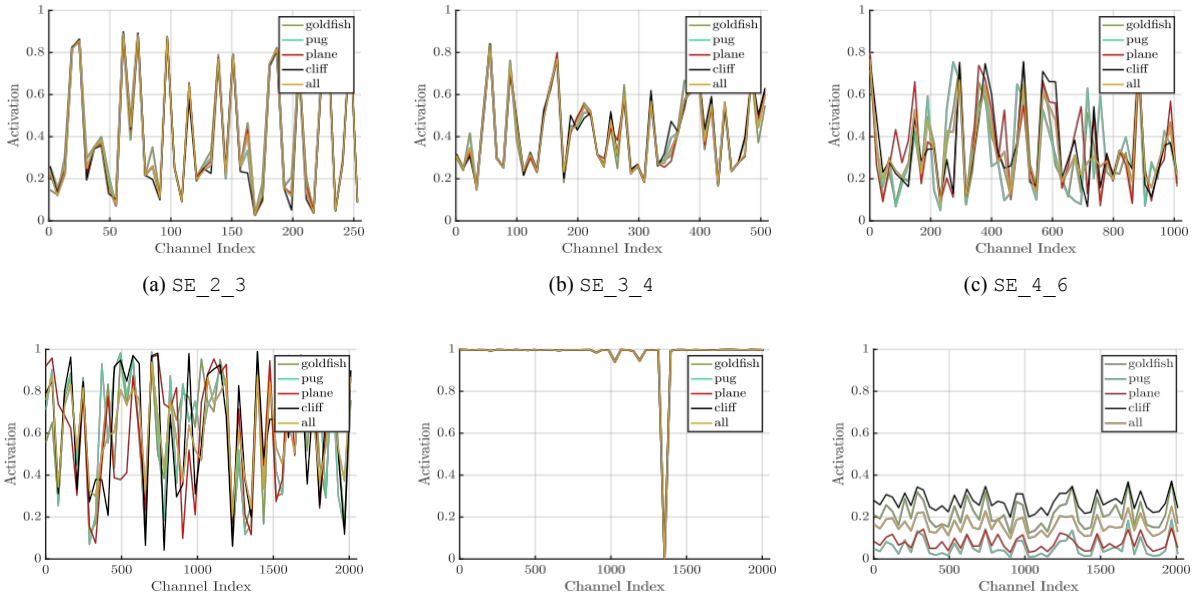
7.1 스퀴즈 효과

스퀴즈 연산에 의해 생성된 글로벌 임베딩이 성능에 중요한 역할을 하는지 평가하기 위해, 동일한 수의 파라미터를 추가하지만 글로벌 평균 풀링을 수행하지 않는 SE 블록의 변형을 실험해 보았습니다. 구체적으로 풀링 연산을 제거하고 두 개의 FC 레이어를 여기 연산자에서 동일한 채널 치수를 가진 대응하는 1×1 컨볼루션으로 대체하여, 여기 출력에서 공간 치수가 입력으로 유지되는 *NoSqueeze를 구현합니다*. SE 블록과 달리 이러한 점 단위 컨볼루션은 로컬 연산자 출력의 함수로만 채널을 다시 매핑할 수 있습니다. 실제로는 딥 네트워크의 후반 레이어는 일반적으로 (이론적으로) 전역적



(a) 잔여 블록 (b) 표준 SE 블록 (c) SE-PRE 블록 (d) SE-POST 블록 (e) SE-아이덴티티 블록

그림 5. 절제 연구에서 살펴본 SE 블록 통합 설계.



(a) SE_2_3 (b) SE_3_4 (c) SE_4_6

(d) SE_5_1 (e) SE_5_2 (f) SE_5_3

그림 6. 이미지넷의 SE-ResNet-50에서 다양한 깊이에서 여기 연산자에 의해 유도된 활성화. 각 활성화 세트는 다음 체계에 따라 이름이 지정됩니다: SE_stageID_blockID. SE_5_2에서의 비정상적인 동작을 제외하고, 활성화는 깊이가 증가함에 따라 점점 더 클래스별로 달라집니다.

표 16
스퀴즈 연산자가 이미지넷에 미치는 영향(오류율 %).

	TOP-1 ERR.	상위 5위 오류.	GFLOPs	매개변수
ResNet-50	23.30	6.55	3.86	25.6M
NoSqueeze	22.93	6.39	4.27	28.1M
SE	**22.28**	**6.03**	3.87	28.1M

수신 필드에서 글로벌 임베딩은 더 이상 네트워크 전체에서 직접 액세스할 수 없습니다. 두 모델의 정확도와 계산 복잡도는 표 16에서 표준 ResNet-50 모델과 비교됩니다. 글로벌 정보의 사용이 모델 성능에 상당한 영향을 미치는 것을 관찰할 수 있으며, 이는 스퀴즈 연산의 중요성을 강조합니다. 또한 *노스퀴즈* 설계와 비교했을 때 SE 블록은 이 글로벌 정보를 계산적으로 간결한 방식으로 사용할 수 있습니다.

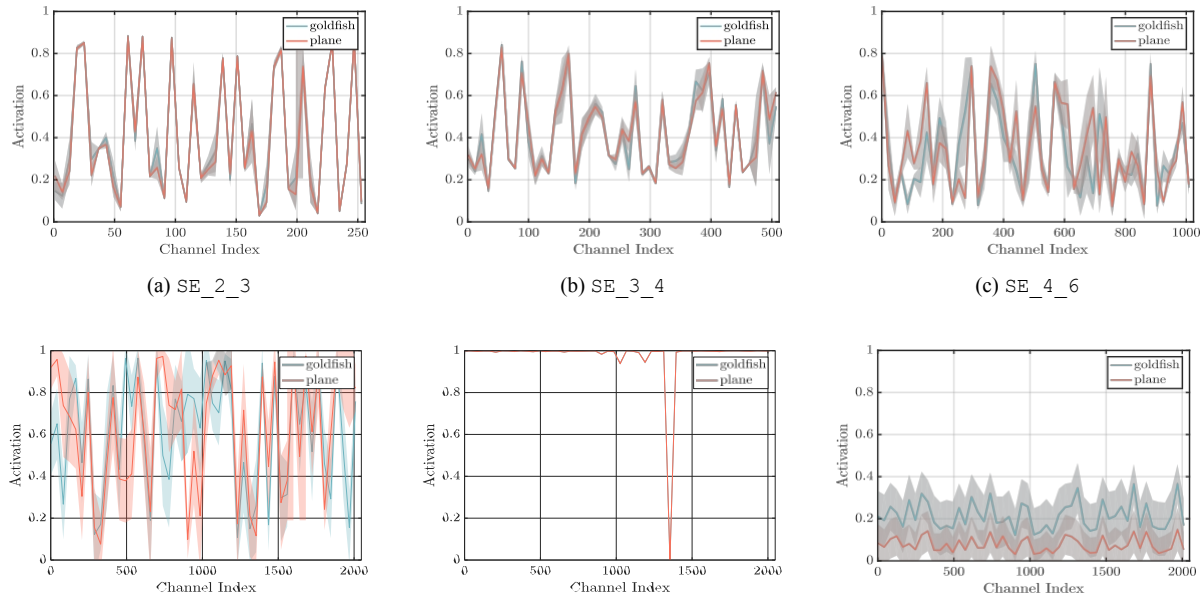
7.2 여기의 역할

SE 블록에서 여기 연산자의 기능을 보다 명확하게 파악하기 위해 이 섹션에서는 다음과 같은 예제를 살펴봅니다.

활성화의 분포를 네트워크의 다양한 깊이에서 다양한 클래스 및 다양한 입력 이미지에 대해 조사합니다. 특히 다른 클래스의 이미지와 한 클래스 내의 이미지에 따라 여기가 어떻게 달라지는지 이해하고자 합니다.

먼저 서로 다른 클래스에 대한 여기 분포를 고려합니다. 구체적으로, 의미적 및 외형적 다양성을 나타내는 네 가지 클래스, 즉 *금붕어*, *퍼그*, *평면*, *절벽을* ImageNet 데이터 세트에서 샘플링합니다*(* 이러한 클래스의 이미지 예는 부록에 나와 있습니다). 그런 다음 검증 세트에서 각 클래스에 대해 50개의 샘플을 추출하고 각 단계의 마지막 SE 블록(다운샘플링 직전)에서 균일하게 샘플링된 50개의 채널에 대한 평균 활성화를 계산하여 그림 6에 그 분포를 표시합니다. 참고로, 1000개의 클래스 전체에 대한 평균 활성화 분포도 함께 플롯합니다.

여기 연산의 역할에 대해 다음 세 가지를 관찰했습니다. 첫째, 네트워크의 초기 계층에서 서로 다른 클래스 간의 분포가 매우 유사합니다(예: SE 2__ 3). 이는 특징 채널의 중요도가 초기 단계에서 여러 클래스에 의해 공유될 가능성이 높다는 것을 시사합니다. 두 번째 관찰은



(a) SE_2_3 (b) SE_3_4 (c) SE_4_6

(d) SE_5_1 (e) SE_5_2 (f) SE_5_3

그림 7. 이미지넷의 금붕어 및 평면 클래스의 이미지 샘플에 대해 SE-ResNet-50의 여러 모듈에서 여기로 유도된 활성화. 모듈의 이름은 "SE_stageID_blockID"입니다.

깊이에 따라 각 채널의 값은 클래스마다 특징의 변별력에 대한 선호도가 다르기 때문에 훨씬 더 클래스별로 달라집니다 (예: SE 4__ 6 및 SE 5__ 1). 이러한 관찰은 이전 연구 [81], [82]의 결과, 즉 초기 레이어 특징은 일반적으로 더 일반적(예: 분류 작업의 맥락에서 클래스 불가지론적)인 반면 후기 레이어 특징은 더 높은 수준의 특이성을 나타낸다는 결과와 일치합니다 [83].

다음으로, 네트워크의 마지막 단계에서 다소 다른 현상을 관찰합니다. SE 5__ 2는 대부분의 활성화가 1에 가까워지는 포화 상태로 향하는 흥미로운 경향을 보입니다. 모든 활성화가 값 1을 취하는 지점에서 SE 블록은 신원 연산자로 축소됩니다. SE 5__ 3의 네트워크 끝에서(분류기 이전의 글로벌 풀링이 바로 뒤따름) 비슷한 패턴이 여러 클래스에 걸쳐 나타나며, 규모에 약간의 변화(분류기에 의해 조정될 수 있음)까지 나타납니다. 이는 SE 5__ 2 및 SE 5__ 3이 네트워크에 재보정을 제공하는 데 있어 이전 블록보다 덜 중요하다는 것을 시사합니다. 이 결과는 4절의 실증적 조사 결과와 일치하며, 마지막 단계의 SE 블록을 제거하면 약간의 성능 손실만으로 추가 파라미터 수를 크게 줄일 수 있음을 보여주었습니다.

마지막으로, 그림 7에는 두 개의 샘플 클래스(금붕어와 *비행기*)에 대해 동일한 클래스 내의 이미지 인스턴스에 대한 활성화의 평균과 표준편차가 표시되어 있습니다. 클래스 간 시각화와 일치하는 추세를 관찰할 수 있는데, 이는 SE 블록의 동적 동작이 클래스와 클래스 내의 인스턴스 모두에 걸쳐 다양하다는 것을 나타냅니다. 특히 단일 클래스 내에서 고려할 수 있는 표현의 다양성이 있는 네트워크의 후기 계층에서 네트워크는 특징 재보정을 활용하여 판별 성능을 개선하는 방법을 학습합니다 [84]. 요약하면, SE 블록은 다음과 같은 인스턴스별 응답을 생성합니다.

그럼에도 불구하고 아키텍처의 여러 계층에서 모델의 점점 더 많은 클래스별 요구 사항을 지원하는 기능을 수행합니다.

8 결론

이 백서에서는 동적 채널별 특징 재보정을 수행하여 네트워크의 표현력을 향상시키도록 설계된 아키텍처 단위인 SE 블록을 제안했습니다. 다양한 실험을 통해 여러 데이터 세트와 작업에서 최첨단 성능을 달성하는 SE넷의 효율성을 보여줍니다. 또한, SE 블록은 이전 아키텍처가 채널별 특징 지연을 적절하게 모델링하지 못한다는 사실을 밝혀냈습니다. 이러한 인사이트가 강력한 변별력이 필요한 다른 작업에도 유용하게 활용될 수 있기를 바랍니다. 마지막으로, SE 블록에서 생성된 특징 중요도 값은 모델 압축을 위한 네트워크 가지치기와 같은 다른 작업에도 유용하게 사용될 수 있습니다.

감사

저자들은 훈련 시스템 최적화와 CIFAR 데이터 세트 실험에 기여한 Momenta의 Chao Li와 Guangyuan Wang에게 감사의 말을 전합니다. 또한 많은 도움이 되는 토론을 해준 Andrew Zisserman, Aravindh Ma- hendran, Andrea Vedaldi에게도 감사의 말씀을 전합니다. 이 연구는 부분적으로 NSFC 보조금(61632003, 61620106003, 61672502, 61571439), 국가 핵심 연구 개발 프로
중국 그램(2017YFB1002701), 마카오 FDCT 보조금(068/2015/A2)을 받았습니다. 사무엘 알바니는 EPSRC AIMS CDT EP/L015897/1의 지원을 받고 있습니다.

부록: 부록: SENET-154 세부 정보

SENet-154는 64×4d ResNeXt-152의 수정된 버전에 SE 블록을 통합하여 구성되며, 이는 다음과 같이 확장됩니다.

(a) 금붕어 (b) 퍼그 (c) 비행기 (d) 절벽

그림 8. 7.2절에 설명된 실험에 사용된 네 가지 이미지넷 클래스의 샘플 이미지.

ResNet-152[13]의 블록 스택 전략을 채택하여 기존 ResNeXt-101[19]을 개선했습니다. 이 모델의 설계 및 훈련에 대한 추가 차이점(SE 블록 사용 외)은 다음과 같습니다: (a) 성능 저하를 최소화하면서 모델의 계산 비용을 줄이기 위해 각 병목 빌딩 블록에 대한 첫 번째 1 × 1 컨볼루션 채널 수를 절반으로 줄였습니다. (b) 첫 번째 7 × 7 컨볼루션 레이어가 연속된 3 × 3 컨볼루션 레이어 3개로 대체되었습니다. (c) 보폭 2 컨볼루션을 사용한 1 × 1 다운 샘플링 투영은 정보를 보존하기 위해 3 × 3 보폭 2 컨볼루션으로 대체되었습니다. (d) 과적합을 줄이기 위해 분류 레이어 앞에 드롭아웃 레이어(드롭아웃 비율 0.2)를 삽입했습니다. (e) 훈련 중에 라벨 평활 정규화([20]에 소개된 바와 같이)를 사용했습니다. (f) 훈련과 테스트 간의 일관성을 보장하기 위해 모든 BN 레이어의 파라미터는 마지막 몇 번의 훈련 기간 동안 동결되었습니다. (g) 훈련은 대규모 배치 크기(2048개)를 지원하기 위해 8대의 서버(64개의 GPU)를 병렬로 연결하여 수행했습니다. 초기 학습 속도는 1.0으로 설정되었습니다.

참고 자료

[1] A. Krizhevsky, I. Sutskever, 및 G. E. Hinton, "심층 컨볼루션 신경망을 이용한 이미지넷 분류", *신경 정보 처리 시스템 컨퍼런스*, 2012.

[2] A. Toshev와 C. Szegedy, "DeepPose: 심층 신경망을 통한 인간 자세 추정", *CVPR*, 2014.

[3] J. Long, E. Shelhamer, 및 T. Darrell, "의미론적 세분화를 위한 완전 컨볼루션 네트워크," *CVPR*, 2015.

[4] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: 영역 제안 네트워크를 통한 실시간 객체 감지를 향하여," *신경 정보 처리 시스템 컨퍼런스*, 2015.

[5] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, 및 A. Rabinovich, " 컨볼루션으로 더 깊이 들어가기", *CVPR*, 2015.

[6] S. Ioffe와 C. Szegedy, "일괄 정규화: 내부 공변량 이동을 줄임으로써 심층 네트워크 학습 가속화", *ICML*, 2015.

[7] S. Bell, C. L. Zitnick, K. Bala, R. Girshick, "인사이드-아웃사이드 네트워크: 스킵 풀링과 반복 신경망 네트워크를 사용하여 컨텍스트에서 객체 감지하기," in *CVPR*, 2016.

[8] A. Newell, K. Yang, J. Deng, "인간 자세 추정을 위한 스택형 모래시계 네트워크", ECCV, 2016.

[9] M. Jaderberg, K. Simonyan, A. Zisserman, 및 K. Kavukcuoglu, "공간 트랜스포머 네트워크", *신경 정보 컨퍼런스 처리 시스템*, 2015.

[10] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. 황, A. 카르파시, A. 코슬라, M. 번스타인, A. C. 버그, 및 L. 페이 페이, "이미지넷 대규모 시각 인식 챌린지," *국제 컴퓨터 비전 저널*, 2015.

[11] K. Simonyan과 A. Zisserman, "대규모 이미지 인식을 위한 매우 심층적인 컨볼루션 네트워크 작업", *ICLR*, 2015.

[12] S. Santurkar, D. Tsipras, A. Ilyas, A. Madry, "배치 정규화는 최적화에 어떻게 도움이 되나요? (아니요, 내부 공변량 *이동에* 관한 것이 아닙니다.)", *신경 정보 처리 컨퍼런스 시스템*, 2018.

[13] K. He, X. Zhang, S. Ren, J. Sun, "이미지 인식을 위한 심층 잔여 학습", *CVPR*, 2016.

[14] K. He, X. Zhang, S. Ren, J. Sun, "심층 잔여 네트워크의 신원 매핑", *ECCV*, 2016.

[15] R. K. 스리바스타바, K. 그레프, J. 슈미드후버, "매우 심층적인 네트워크 훈련", *신경 정보 처리 시스템 컨퍼런스*, 2015.

[16] Y. Chen, J. Li, H. Xiao, X. Jin, S. Yan, J. Feng, "이중 경로 네트워크", *신경 정보 처리 시스템 컨퍼런스(신경 정보 처리 시스템 컨퍼런스)*, 2017.

[17] G. Huang, Z. Liu, K. Q. 와인버거, L. 마텐, "밀도 높은 연결 컨볼루션 네트워크", *CVPR*, 2017.

[18] Y. Ioannou, D. Robertson, R. Cipolla, 및 A. Criminisi, "Deep roots: 계층적 필터 그룹으로 CNN 효율성 향상," in *CVPR*, 2017.

[19] S. Xie, R. Girshick, P. Dolla´r, Z. Tu, K. He, "심층 신경망을 위한 집계된 잔여 변환," in *CVPR*, 2017.

[20] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna, "컴퓨터 비전을 위한 인셉션 아키텍처에 대한 재검토," in *CVPR*, 2016.

[21] C. Szegedy, S. Ioffe, V. Vanhoucke, A. Alemi, "Inception- v4, inception-resnet 및 잔여 연결이 학습에 미치는 영향", *AAAI 인공 지능 컨퍼런스*, 2016.

[22] M. Jaderberg, A. Vedaldi, 및 A. Zisserman, "낮은 순위 확장을 통한 콘볼루션 신경망의 속도 향상", in *BMVC*, 2014.

[23] F. Chollet, "Xception: 깊이별로 분리 가능한 컨볼루션을 사용한 딥러닝," in *CVPR*, 2017.

[24] M. Lin, Q. Chen, 및 S. Yan, "네트워크 내 네트워크", *ICLR*, 2014.

[25] G. F. Miller, P. M. Todd, S. U. Hegde, "유전 알고리즘을 이용한 신경망 설계." in *ICGA*, 1989.

[26] K. O. Stanley와 R. Miikkulainen, "증강 토폴로지를 통한 신경망의 진화", *진화적 계산*, 2002.

[27] J. Bayer, D. Wierstra, J. Togelius, J. Schmidhuber, "진화하는 시퀀스 학습을 위한 메모리 셀 구조", *ICANN*, 2009.

[28] R. Jozefowicz, W. Zaremba, 및 I. Sutskever, "An empirical expl- ration of recurrent network architectures," in *ICML*, 2015.

[29] L. Xie와 A. L. Yuille, "유전적 CNN", *ICCV*, 2017.

[30] E. Real, S. Moore, A. Selle, S. Saxena, Y. L. Suematsu, J. Tan, Q. Le, A. Kurakin, "대규모 이미지 분류기 진화," in *ICML*, 2017.

[31] E. Real, A. Aggarwal, Y. Huang, 및 Q. V. Le, "이미지 분류기 아키텍처 검색을 위한 정규화된 진화," *arXiv preprint arXiv:1802.01548*, 2018.

[32] T. Elsken, J. H. Metzen, 및 F. Hutter, "라마키안 진화를 통한 효율적인 다중 목표 신경 구조 검색", *arXiv preprint arXiv:1804.09081*, 2018.

[33] H. Liu, K. Simonyan, 및 Y. Yang, "DARTS: 차별적인 아키텍처 검색," *arXiv preprint arXiv:1806.09055*, 2018.

[34] J. Bergstra와 Y. Bengio, "하이퍼 파라미터 최적화를 위한 무작위 검색", *JMLR*, 2012.

[35] C. Liu, B. Zoph, J. Shlens, W. Hua, L.-J. Li, L. Fei-Fei, A. Yuille, J. Huang, 및 K. Murphy, "프로그레시브 신경 아키텍처 검색", *ECCV*, 2018.

[36] R. Negrinho and G. Gordon, "Deeparchitect: 딥 아키텍처의 자동 설계 및 학습," *arXiv 사전 인쇄본 arXiv:1704.08792*, 2017.

[37] S. Saxena와 J. Verbeek, "*컨볼루션* 신경 구조", *신경 정보 처리 시스템에 관한 연구*, 2016.

[38] A. Brock, T. Lim, J. M. Ritchie, N. Weston, "SMASH: 하이퍼네트워크를 통한 원샷 모델 아키텍처 검색," in *ICLR*, 2018.

[39] B. Baker, O. 굽타, R. 라스카, N. 나익, "성능 예측을 이용한 신경 아키텍처 검색 가속화", *ICLR Work- shop,* 2018.

[40] B. Baker, O. Gupta, N. Naik, R. Raskar, "강화 학습을 이용한 신경망 아키텍처 설계", *ICLR*, 2017.

[41] B. Zoph와 Q. V. Le, "강화 학습을 통한 신경 구조 검색", *ICLR*, 2017.

[42] B. Zoph, V. Vasudevan, J. Shlens, 및 Q. V. Le, "확장 가능한 이미지 인식을 위한 학습 전송 가능 아키텍처," in *CVPR*, 2018.

[43] H. Liu, K . Simonyan, O . Vinyals, C . Fernando, 및 K. 카부쿠오글루, "효율적인 아키텍처 검색을 위한 계층적 표현", *ICLR*, 2018.

[44] H. Pham, M. Y. Guan, B. Zoph, Q. V. Le, J. Dean, "파라미터 공유를 통한 효율적인 신경망 아키텍처 검색," in *ICML*, 2018.

[45] M. Tan, B. Chen, R. Pang, V. Vasudevan, 및 Q. V. Le, "Mnas- net: 모바일을 위한 플랫폼 인식 신경 아키텍처 검색," *arXiv preprint arXiv:1807.11626*, 2018.

[46] B. A. Olshausen, C. H. Anderson 및 D. C. V. Essen, "정보의 동적 라우팅에 기반한 시각적 주의 및 불변 패턴 인식의 신경생물학적 논리 모델", *신경과학 저널*, 1993.

[47] L. Itti, C. Koch, E. Niebur, "신속한 장면 분석을 위한 주목도 기반 시각적 주의 모델", *IEEE 트랜잭션: 패턴 분석 및 기계 지능*, 1998.

[48] L. Itti와 C. Koch, "시각적 주의의 계산적 모델링", "시각적 주의의 계산적 모델링,"
네이처 리뷰 신경과학, 2001.

[49] H. Larochelle과 G. E. Hinton, "학습을 통해 동공과 3차 볼츠만 머신을 결합하는 방법", *신경 정보 처리 시스템 컨퍼런스*, 2010.

[50] V. Mnih, N. Heess, A. Graves, K. Kavukcuoglu, "시각적 주의의 반복 모델", *신경 정보 처리 시스템 컨퍼런스,* 2014.

[51] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, 및 I. Polosukhin, "주의력만 있으면 됩니다.", *신경 정보 처리 시스템 컨퍼런스*, 2017.

[52] T. Bluche, "종단 간 필기 단락 인식을 위한 공동 선 분할 및 전사", *신경 정보 처리 시스템 컨퍼런스*, 2016.

[53] A. Miech, I. Laptev, J. Sivic, "비디오 분류를 위한 컨텍스트 게이팅을 통한 학습 가능한 풀링", *arXiv:1706.06905*, 2017.

[54] C. Cao, X. Liu, Y. Yang, Y. Yu, J. Wang, Z. Wang, Y. Huang, L. Wang, C. Huang, W. Xu, D. Ramanan, T. S. Huang, "보고 두 번 생각하기: 피드백을 통한 하향식 시각적 주의 포착 컨볼루션 신경망," in *ICCV*, 2015.

[55] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, 및 Y. Bengio, "보여주고, 듣고, 말하기: 시각적 주의를 기울인 신경 이미지 캡션 생성", *ICML*, 2015.

[56] L. Chen, H. Zhang, J. Xiao, L. Nie, J. Shao, W. Liu, and T. Chua, "SCA-CNN: 이미지 캡션을 위한 컨볼루션 네트워크의 공간 및 채널별 주의," in *CVPR*, 2017.

[57] J. S. Chung, A. Senior, O. Vinyals, A. Zisserman, "입술로 읽는 야생의 문장," in *CVPR*, 2017.

[58] F. Wang, M. Jiang, C. Qian, S. Yang, C. Li, H. Zhang, X. Wang, 및 X. Tang, "이미지 분류를 위한 잔여 관심 네트워크", 에서 *CVPR*, 2017.

[59] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "CBAM: Convolutional 블록 주의 모듈," in *ECCV*, 2018.

[60] J. Yang, K. Yu, Y. Gong, T. Huang, "이미지 분류를 위한 스파스 코딩을 이용한 선형 공간 피라미드 매칭," in *CVPR*, 2009.

[61] J. Sanchez, F. Perronnin, T. Mensink, J. Verbeek, "피셔 벡터를 사용한 이미지 분류: 이론과 실제," *국제 컴퓨터 비전 저널*, 2013.

[62] L. Shen, G. Sun, Q. Huang, S. Wang, Z. Lin, E. Wu, "대 규 모 이미지 분류에 적용되는 다수준 판별 사전 학습," *IEEE TIP,* 2015.

[63] V. Nair와 G. E. Hinton, "정류된 선형 유닛으로 제한된 볼츠만 기계 개선", *ICML*, 2010.

[64] A. G. 하워드, M. 주, B. 첸, D. 칼레니첸코, W. 왕, T. Weyand, M. Andreetto, 및 H. Adam, "MobileNets: 모바일 비전 애플리케이션을 위한 효율적인 컨볼루션 신경망", *arXiv:1704.04861*, 2017.

[65] X. Zhang, X. Zhou, M. Lin, and J. Sun, "ShuffleNet: 모바일 기기를 위한 매우 효율적인 컨볼루션 신경망", *CVPR*, 2018.

[66] K. He, X. Zhang, S. Ren, J. Sun, "정류기에 대해 깊이 파헤치기: 이미지넷 분류에서 인간을 능가하는 성능," in *ICCV*, 2015.

[67] S. 자가루이코와 N. 코모다키스, "넓은 잔여 네트워크", 에서 *BMVC*, 2016.

[68] X. 가스탈디, "흔들기-흔들기 정규화", *arXiv preprint arXiv:1705.07485*, 2017.

[69] T. DeVries와 G. W. Taylor, "향상된 컨볼루션 신경망의 정규화," *arXiv preprint arXiv:1708.04552*, 2017.

[70] A. Krizhevsky와 G. Hinton, "작은 이미지에서 여러 계층의 기능 학습", Citeseer, Tech. Rep., 2009.

[71] G. Huang, Y. Sun, Z. Liu, D. Sedra, K. Q. Weinberger, "확률적 깊이를 가진 딥 네트워크," in *ECCV*, 2016.

[72] L. Shen, Z. Lin, G. Sun, J. Hu, "Places401 and places365 models", https://github.com/lishen-shirley/Places2-CNNs, 2016.

[73] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, 및 A. Torralba, "Places: 장면 인식을 위한 1,000만 개의 이미지 데이터베이스," *IEEE 트랜잭션 패턴 분석 및 기계 지능*, 2017.

[74] L. Shen, Z. Lin, 및 Q. Huang, "심층 컨볼루션 신경망의 효과적인 학습을 위한 릴레이 역전파," in *ECCV*, 2016.

[75] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dolla´r 및 C. L. Zitnick, "Microsoft COCO: 컨텍스트의 공통 개체," in *ECCV*, 2014.

[76] R. Girshick, I. Radosavovic, G. Gkioxari, P. Dolla´r 및 K. He, "Detectron", https://github.com/facebookresearch/detectron, 2018.

[77] D. Han, J. Kim, "심층 피라미드형 잔여 네트워크", in *CVPR*, 2017.

[78] X. Zhang, Z. Li, C. C. Loy, D. Lin, "Polynet: 매우 심층적인 네트워크에서 구조적 다양성을 추구하다", *CVPR*, 2017.

[79] E. D. Cubuk, B. Zoph, D. Mane, V. Vasudevan, 및 Q. V. Le, "Autoaugment: 데이터로부터 증강 정책 학습하기," *arXiv preprint arXiv:1805.09501*, 2018.

[80] D. Mahajan, R. Girshick, V. Ramanathan, K. He, M. Paluri, Y. Li, A. Bharambe, 및 L. van der Maaten, "약하게 감독된 사전 교육( ) 의 한계 탐구", *ECCV,* 2018.

[81] H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng, "Convolutional deep belief networks for scalable unsupervised learning of hierar- chical representations," in *ICML*, 2009.

[82] J. Yosinski, J. Clune, Y. Bengio, H. Lipson, "심층 신경망의 특징은 얼마나 이전 가능한가?" in *Conference on Neural 정보 처리 시스템*, 2014.

[83] A. S. Morcos, D. G. Barrett, N. C. Rabinowitz 및 M. Botvinick, "일반화를 위한 단일 방향의 중요성", *ICLR*, 2018.

[84] J. Hu, L. Shen, S. Albanie, G. Sun, A. Vedaldi, "Gather-excite: 컨볼루션 신경망에서 특징 컨텍스트 활용하기," *신경 정보 처리 시스템 컨퍼런스*, 2018.