컨볼루션 신경망 소개

키론 오시어[1] 및 라이언 내쉬[2]

[1] 애버리스트위스 대학교 컴퓨터 과학과, 세레디지온, SY23 3DB
keo7@aber.ac.uk
[2] 랭커스터 대학교 컴퓨팅 및 커뮤니케이션 스쿨, 랭커스터 대학교, 랭커셔, LA1 4YW
nashrd@live.lancs.ac.uk

요약. 머신러닝 분야는 인공 신경망(ANN)의 등장과 함께 극적인 변화를 거듭해 왔습니다. 생물학적으로 영감을 받은 이 계산 모델은 일반적인 머신러닝 작업에서 이전 형태의 인공 지능의 성능을 훨씬 뛰어넘을 수 있습니다. 가장 인상적인 형태의 ANN 아키텍처 중 하나는 컨볼루션 신경망(CNN)입니다. CNN은 주로 어려운 이미지 기반 패턴 인식 작업을 해결하는 데 사용되며, 정확하면서도 간단한 아키텍처를 통해 ANN을 시작하는 간단한 방법을 제공합니다.

이 문서에서는 CNN에 대한 간략한 소개와 함께 최근 발표된 논문과 이 놀랍도록 환상적인 이미지 인식 모델을 개발하는 새로운 기법에 대해 설명합니다. 이 소개에서는 ANN과 머신 러닝의 기초에 대해 잘 알고 있다고 가정합니다.

키워드: 패턴 인식, 인공 신경망, 머신 러닝, 이미지 분석

1 소개

인공 신경망(ANN)은 인간의 뇌와 같은 생물학적 신경계가 작동하는 방식에서 많은 영감을 받은 계산 처리 시스템입니다. ANN은 주로 상호 연결된 수많은 계산 노드(뉴런이라고 함)로 구성되며, 이 노드들은 분산된 방식으로 서로 얽혀서 최종 출력을 최적화하기 위해 입력으로부터 집합적으로 학습합니다.

ANN의 기본 구조는 그림 1과 같이 모델링할 수 있습니다. 일반적으로 다차원 벡터의 형태로 입력 레이어에 입력을 로드하면 숨겨진 레이어에 이를 분배합니다. 그러면 숨겨진 레이어는 이전 레이어에서 결정을 내리고 그 자체의 확률적 변화가 최종 출력에 어떤 해를 끼치거나 개선하는지 평가하게 되는데, 이를 학습 과정이라고 합니다. 여러 개의 숨겨진 레이어가 서로 겹쳐져 있는 것을 일반적으로 딥러닝이라고 합니다.

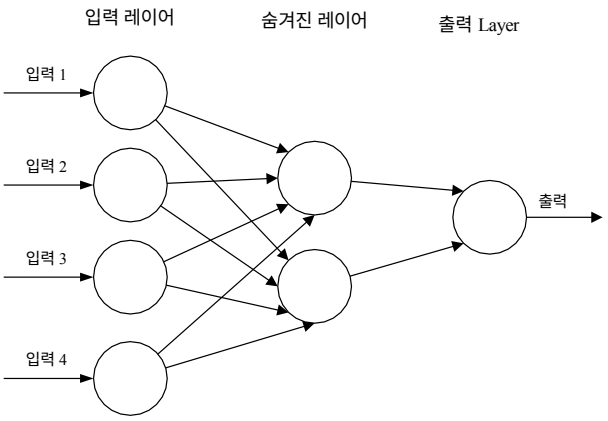

그림 1: 입력 계층, 숨겨진 계층, 출력 계층으로 구성된 간단한 3계층 피드포워드 신경망(FNN). 이 구조는 피드 포워드 신경망(FNN), 제한 볼츠만 머신(RBM), 순환 신경망(RNN) 등 여러 일반적인 ANN 아키텍처의 기초가 됩니다.

이미지 처리 작업의 두 가지 주요 학습 패러다임은 지도 학습과 비지도 학습입니다. **지도 학습은** 목표 역할을 하는 사전 레이블이 지정된 입력을 통해 학습하는 것입니다. 각 훈련 예제에는 입력값(벡터) 세트와 하나 이상의 관련 지정된 출력값이 있습니다. 이러한 형태의 학습의 목표는 학습을 통해 학습 예제의 출력값을 정확하게 계산하여 모델의 전체 분류 오류를 줄이는 것입니다.

비지도 학습은 학습 세트에 학습자가 포함되어 있지 않다는 점에서 다릅니다. 성공 여부는 일반적으로 네트워크가 관련 비용 함수를 줄이거나 늘릴 수 있는지 여부에 따라 결정됩니다. 그러나 대부분의 이미지 중심 패턴 인식 작업은 일반적으로 지도 학습을 사용한 분류에 의존한다는 점에 유의해야 합니다.

컨볼루션 신경망(CNN)은 학습을 통해 스스로 최적화하는 뉴런으로 구성된다는 점에서 기존 인공**신경망과** 유사합니다. 각 뉴런은 여전히 입력을 받아 연산(예: 스칼라 곱 뒤 비선형 함수)을 수행하며, 이는 수많은 ANN의 기초가 됩니다. 입력 원시 이미지 벡터부터 최종 출력인 클래스 점수까지, 네트워크 전체는 여전히 하나의 지각 점수 함수(가중치)를 표현합니다. 마지막 레이어에는 클래스와 관련된 손실 함수가 포함되며, 기존 ANN을 위해 개발된 모든 일반적인 팁과 요령이 여전히 적용됩니다.

CNN과 기존 ANN의 유일한 차이점은 CNN은 주로 이미지 내 패턴 인식 분야에서 사용된다는 점입니다. 이를 통해 이미지별 특징을 아키텍처에 인코딩하여 네트워크를 만들 수 있습니다.

이미지 중심 작업에 더 적합하며, 모델 설정에 필요한 매개 변수를 더욱 줄일 수 있습니다.

기존 형태의 ANN의 가장 큰 한계 중 하나는 이미지 데이터를 계산하는 데 필요한 계산 복잡성으로 인해 어려움을 겪는 경향이 있다는 것입니다. 손으로 쓴 숫자로 구성된 MNIST 데이터베이스와 같은 일반적인 머신 러닝 벤치마킹 데이터 세트는 이미지 크기가 28×28로 비교적 작기 때문에 대부분의 형태의 ANN에 적합합니다. 이 데이터 세트의 경우 첫 번째 숨겨진 계층의 단일 뉴런에는 784개의 가중치($28 \times 28 \times 1$, 여기서 1은 MNIST가 흑백 값으로만 정규화된 것을 염두에 두어야 함)가 포함되며, 이는 대부분의 형태의 ANN에서 관리할 수 있는 수준입니다.

64×64의 더 큰 컬러 이미지 입력을 고려하면 첫 번째 레이어의 단일 뉴런에 대한 가중치 수는 12,288로 크게 증가합니다. 또한 이러한 입력 규모를 처리하려면 네트워크도 색상 정규화된 MNIST 숫자를 분류하는 데 사용되는 네트워크보다 훨씬 커야 한다는 점을 고려하면 이러한 모델을 사용할 때의 단점을 이해할 수 있을 것입니다.

1.1 오버피팅

하지만 이것이 왜 중요할까요? 네트워크에 숨겨진 레이어의 수를 늘리고 그 안에 있는 뉴런의 수를 늘리면 될까요? 이 질문에 대한 간단한 대답은 '아니오'입니다. 그 이유는 두 가지가 있는데, 하나는 이 거대한 인공신경망을 훈련시킬 수 있는 연산 능력과 시간이 무한하지 않다는 단순한 문제 때문입니다.

두 번째 이유는 과적합의 효과를 막거나 줄이기 위해서입니다. **과적합이란** 기본적으로 네트워크가 여러 가지 이유로 인해 효과적으로 학습할 수 없는 경우를 말합니다. 이는 모든 머신 러닝 알고리즘은 아니더라도 대부분의 머신 러닝 알고리즘에서 중요한 개념이며, 그 영향을 줄이기 위해 모든 예방 조치를 취하는 것이 중요합니다. 모델이 과적합의 징후를 보이면 학습 데이터 세트뿐만 아니라 테스트 및 예측 세트에 대한 일반화된 특징을 정확히 찾아내는 능력이 저하될 수 있습니다.

이것이 바로 인공신경망의 복잡성을 줄이는 주된 이유입니다. 학습에 필요한 매개변수가 적을수록 네트워크가 과적합할 가능성이 줄어들고, 당연히 모델의 예측 성능도 향상됩니다.

2 CNN 아키텍처

앞서 언급했듯이 CNN은 주로 입력이 이미지로 구성된다는 점에 중점을 둡니다. 따라서 특정 유형의 데이터를 처리하는 데 가장 적합한 방식으로 아키텍처를 설정하는 데 중점을 둡니다.

주요 차이점 중 하나는 CNN 내의 레이어가 입력의 공간적 차원(높이와 **너비**)과 **깊이라는** 세 가지 차원으로 구성된 뉴런으로 구성된다는 것입니다. 깊이는 ANN 내의 총 레이어 수를 의미하는 것이 아니라 활성화 볼륨의 3차원을 의미합니다. 표준 ANN과 달리 특정 레이어 내의 뉴런은 그 앞의 레이어의 작은 영역에만 연결됩니다.

실제로 이는 앞서 제시한 예제에서 입력 '볼륨'의 차원이 64 × 64 × 3(높이, 너비, 깊이)이 된다는 것을 의미하며, 전체 입력 차원을 깊이 차원에 걸친 클래스 점수의 작은 볼륨으로 압축했으므로 최종 출력 레이어는 1 × 1 × n(여기서 n은 가능한 클래스 수를 나타냄)의 차원으로 구성된다는 것을 뜻합니다.

2.1 전체 아키텍처

CNN은 세 가지 유형의 레이어로 구성됩니다. 컨볼루션 레이어, 풀링 레이어, **완전 연결 레이어입니다**. 이러한 레이어가 쌓이면 CNN 아키텍처가 형성됩니다. MNIST 분류를 위한 단순화된 CNN 아키텍처는 그림 2에 나와 있습니다.

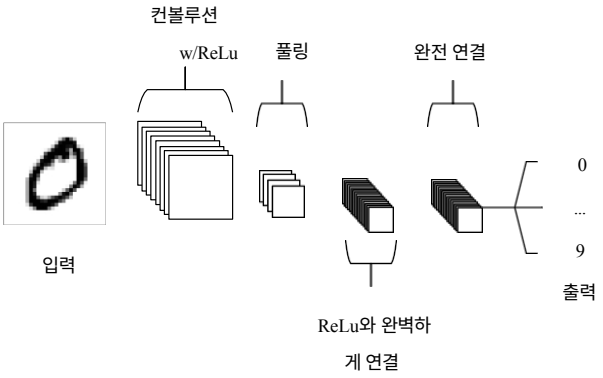

그림 2: 단 5개의 레이어로 구성된 간단한 CNN 아키텍처

위의 예제 CNN의 기본 기능은 네 가지 주요 영역으로 나눌 수 있습니다.

1. 다른 형태의 ANN에서 볼 수 있듯이 **입력 레이어는** 이미지의 픽셀 값을 보유합니다.
2. **컨볼루션 레이어는** 뉴런의 가중치와 입력 볼륨에 연결된 영역 사이의 스칼라 곱을 계산하여 입력의 로컬 영역에 연결된 뉴런의 출력을 결정합니다. **정류된 선형 단위**(일반적으로 ReLu로 줄여서 부름)는 다음을 적용하는 것을 목표로 합니다.

의 출력에 시그모이드와 같은 '요소별' 활성화 함수를 추가합니다.
이전 레이어에서 생성된 활성화.

3. 그러면 **풀링 레이어는** 주어진 입력의 공간적 차원을 따라 간단히 다운샘플링을 수행하여 해당 활성화 내의 파라미터 수를 더욱 줄입니다.

4. 그런 다음 **완전히 연결된 레이어는** 표준 ANN에서 볼 수 있는 것과 동일한 작업을 수행하고 활성화에서 클래스 점수를 생성하여 분류에 사용하려고 시도합니다. 또한 성능 향상을 위해 이러한 레이어 사이에 ReLu를 사용할 수도 있습니다.

이 간단한 변환 방법을 통해 CNN은 컨볼루션 및 다운샘플링 기술을 사용하여 원래 입력 레이어를 레이어별로 변환하여 분류 및 회귀 목적의 클래스 점수를 생성할 수 있습니다.

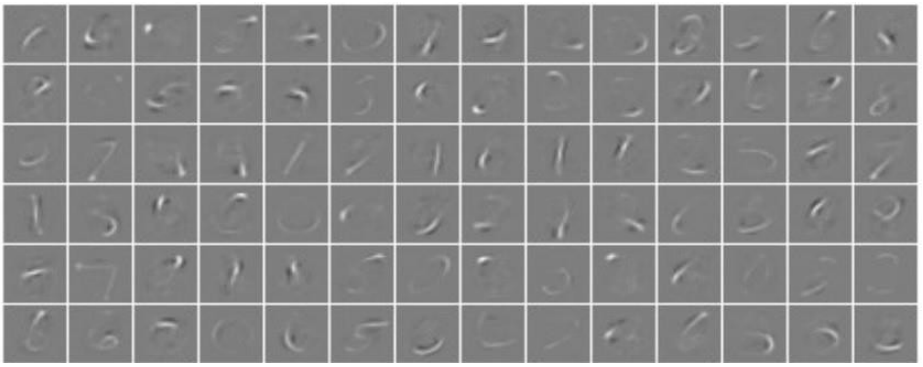

그림 3: 손으로 쓴 숫자로 구성된 MNIST 데이터베이스를 학습한 후 단순 심층 CNN의 첫 번째 컨볼루션 계층에서 가져온 활성화. 자세히 살펴보면 네트워크가 특정 숫자 숫자에 고유한 문자를 성공적으로 포착한 것을 확인할 수 있습니다.

그러나 단순히 CNN 아키텍처의 전반적인 구조를 이해하는 것만으로는 충분하지 않다는 점에 유의해야 합니다. 이러한 모델을 생성하고 최적화하는 데는 상당한 시간이 소요될 수 있으며 상당히 혼란스러울 수 있습니다. 이제 각 레이어를 자세히 살펴보면서 하이퍼파라미터와 연결성을 자세히 살펴보겠습니다.

2.2 컨볼루션 레이어

이름에서 알 수 있듯이 컨볼루션 레이어는 CNN의 작동 방식에서 중요한 역할을 합니다. 레이어 매개변수는 학습 가능한 **커널의** 사용에 중점을 둡니다.

이러한 커널은 일반적으로 공간적 차원은 작지만 입력의 전체 깊이에 걸쳐 퍼져 있습니다. 데이터가 컨볼루션 레이어에 도달하면 이 레이어는 입력의 공간 차원에 걸쳐 각 필터를 컨볼루션하여 2D 활성화 맵을 생성합니다. 이러한 활성화 맵은 그림 3에서 볼 수 있듯이 시각화할 수 있습니다.

입력을 미끄러지면서 해당 커널의 각 값에 대해 스칼라 곱이 계산됩니다. (그림 4) 이를 통해 네트워크는 입력의 특정 공간 위치에서 특정 특징을 발견할 때 '발동'하는 커널을 학습하게 됩니다. 이를 일반적으로 **활성화라고** 합니다.

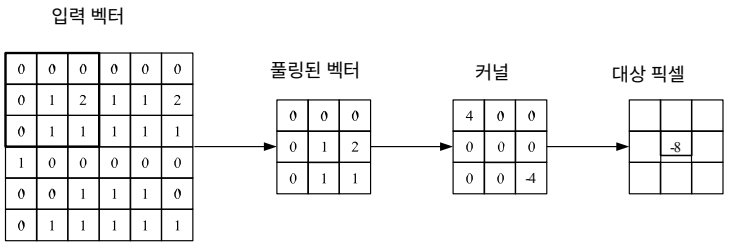

그림 4: 컨볼루션 레이어의 시각적 표현. 커널의 중심 요소는 입력 벡터 위에 배치되며, 이 요소는 자체와 주변 픽셀의 가중치 합으로 계산되어 대체됩니다.

모든 커널에는 해당 활성화 맵이 있으며, 이 맵은 깊이 차원을 따라 쌓여 컨볼루션 레이어에서 전체 출력 볼륨을 형성합니다.

앞서 언급했듯이 이미지와 같은 입력에 대해 ANN을 훈련하면 모델이 너무 커서 효과적으로 훈련할 수 없습니다. 이는 표준 ANN 뉴런의 완전 연결 방식에서 비롯되므로 이를 완화하기 위해 컨볼루션 레이어의 모든 뉴런은 입력 볼륨의 작은 영역에만 연결됩니다. 이 영역의 크기를 일반적으로 뉴런의 **수용 필드 크기라고** 합니다. 깊이를 통한 연결의 크기는 거의 항상 입력의 깊이와 같습니다.

예를 들어, 네트워크에 입력되는 이미지가 64 × 64 × 3 크기의 이미지(64 × 64 크기의 RGB 컬러 이미지)이고 수용 필드 크기를 6 × 6으로 설정하면 컨볼루션 계층 내의 각 뉴런에 총 108개의 가중치가 있습니다. (6 × 6 × 3에서 3은 볼륨의 깊이에 걸친 연결의 크기입니다.) 이를 다른 형태의 ANN에서 볼 수 있는 표준 뉴런에 비유하자면 각각 12,288개의 가중치를 포함하게 됩니다.

또한 컨볼루션 레이어는 출력의 최적화를 통해 모델의 복잡성을 크게 줄일 수 있습니다. 이는 **깊이**, **보폭**, **제로 패딩** 설정이라는 세 가지 하이퍼파라미터를 통해 최적화됩니다.

컨볼루션 레이어가 생성하는 출력 볼륨의 **깊이는** 레이어 내의 뉴런 수를 통해 입력의 동일한 영역에 수동으로 설정할 수 있습니다. 이는 숨겨진 레이어의 모든 뉴 런 이미리 모든 단일 뉴런에 직접 연결되어 있는 다른 형태의 인공 신경망에서도 볼 수 있습니다. 이 하이퍼파라미터를 줄이면 네트워크의 전체 뉴런 수를 크게 최소화할 수 있지만, 모델의 패턴 인식 기능도 크게 저하될 수 있습니다.

또한 수용 필드를 배치하기 위해 입력의 공간 차원을 중심으로 깊이를 설정하는 **보폭을** 정의할 수 있습니다. 예를 들어 보폭을 1로 설정하면 수용 필드가 심하게 겹쳐져 매우 큰 활성화가 발생합니다. 또는 보폭을 더 큰 숫자로 설정하면 겹치는 양이 줄어들고 공간 차원이 낮은 출력을 생성합니다.

제로 패딩은 입력의 테두리를 채우는 간단한 프로세스로, 출력 볼륨의 치수를 더욱 효과적으로 제어할 수 있는 방법입니다.

이러한 기법을 사용하면 컨볼루션 레이어 출력의 공간 차원이 변경된다는 점을 이해하는 것이 중요합니다. 이를 계산하려면 다음 공식을 사용하면 됩니다:

$$\frac{(V\text{-}R) + 2Z\ S}{+\ 1}$$

여기서 V*는* 입력 볼륨 크기(*높이×너비×깊이*)를 나타내고, R*은* 수용 필드 크기, Z*는* 제로 패딩 설정량, S*는* 보폭을 *나타냅니다*. 이 방정식의 계산 결과가 정수와 같지 않으면 뉴런이 주어진 입력에 깔끔하게 맞출 수 없으므로 보폭이 잘못 설정된 것입니다.

지금까지의 최선의 노력에도 불구하고 *실제* 치수의 이미지를 입력하면 여전히 모델의 성능이 저하된다는 사실을 알게 될 것입니다. 그러나 컨볼루션 레이어 내의 전체 파라미터 수를 크게 줄일 수 있는 방법이 개발되었습니다.

매개변수 공유는 특정 공간 영역에서 하나의 영역 특징이 계산에 유용하다면 다른 영역에서도 유용할 가능성이 높다는 가정 하에 작동합니다. 출력 볼륨 내의 각 개별 활성화 맵을 동일한 가중치와 편향으로 제한하면 컨볼루션 레이어에서 생성되는 파라미터 수가 크게 줄어듭니다.

역전파 단계가 발생하면 출력의 각 뉴런은 전체 기울기를 나타내며, 이 기울기는 깊이 전체에 걸쳐 합산될 수 있습니다.
-를 사용하여 모든 가중치를 업데이트하는 것이 아니라 단일 가중치 세트만 업데이트합니다.

2.3 풀링 레이어

풀링 레이어는 표현의 차원을 점진적으로 줄여 모델의 파라미터 수와 계산 복잡성을 더욱 줄이는 것을 목표로 합니다.

풀링 레이어는 입력의 각 활성화 맵에 대해 작동하며 "MAX" 함수를 사용하여 차원을 확장합니다. 대부분의 CNN에서 이러한 **풀링 레이어는** 입력의 공간 차원을 따라 2×2의 커널이 2의 보폭으로 적용되는 **최대 풀링 레이어의** 형태로 제공됩니다. 이렇게 하면 깊이 볼륨을 표준 크기로 유지하면서 활성화 맵의 크기를 원래 크기의 25%로 축소할 수 있습니다.

풀링 레이어의 파괴적인 특성으로 인해 일반적으로 관찰되는 최대 풀링 방법은 두 가지뿐입니다. 일반적으로 풀링 레이어의 보폭과 필터는 모두 2×2로 설정되어 레이어가 입력의 공간 차원 전체에 걸쳐 확장될 수 있습니다. 또한 보폭을 2로 설정하고 커널 크기를 다음과 같이 설정하는 **중복 풀링을** 사용할 수도 있습니다.
3. 풀링의 파괴적인 특성으로 인해 커널 크기가 3을 초과하면 일반적으로 모델의 성능이 크게 저하됩니다.

최대 풀링 외에도 CNN 아키텍처에는 일반 풀링이 포함될 수 있다는 점을 이해하는 것도 중요합니다. **일반 풀링** 레이어는 L1/L2 정규화, 평균 풀링 등 다양한 공통 연산을 수행할 수 있는 풀링 뉴런으로 구성됩니다. 하지만 이 튜토리얼에서는 주로 최대 풀링의 사용에 초점을 맞출 것입니다.

2.4 완전히 연결된 레이어

완전히 연결된 레이어에는 인접한 두 레이어의 뉴런이 그 안의 어떤 레이어와도 연결되지 않고 직접 연결된 뉴런이 포함되어 있습니다. 이는 전통적인 형태의 인공 신경망에서 뉴런이 배열되는 방식과 유사합니다. (그림 1)

3 레시피

CNN을 구성하는 데 필요한 레이어의 수는 비교적 적지만, CNN 아키텍처를 구성하는 정해진 방법은 없습니다. 즉, 단순히 몇 개의 레이어를 조합하여 작동하기를 기대하는 것은 어리석은 일입니다. 관련 문헌을 읽어보면 다른 형태의 ANN과 마찬가지로 CNN도 일반적인 아키텍처를 따르는 경향이 있다는 것을 알 수 있습니다. 이 일반적인 아키텍처는 그림 2에서 볼 수 있듯이 컨볼루션 레이어를 쌓은 다음 레이어를 반복적으로 풀링한 다음 완전히 연결된 레이어로 피드 포워드하는 방식입니다.

또 다른 일반적인 CNN 아키텍처는 그림 5와 같이 각 풀링 레이어 앞에 두 개의 컨볼루션 레이어를 쌓는 것입니다. 여러 컨볼루션 레이어를 쌓으면 입력 벡터의 더 복잡한 특징을 선택할 수 있으므로 이 방법을 강력히 권장합니다.

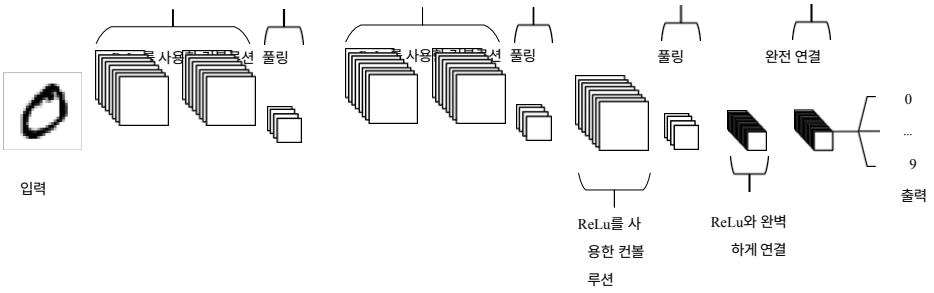

그림 5: 풀링 레이어를 통과하기 전에 하나 또는 여러 개의 완전히 연결된 ReLus 사이에 컨볼루션 레이어가 연속적으로 쌓인 일반적인 형태의 CNN 아키텍처입니다.

또한 큰 컨볼루션 레이어를 여러 개의 작은 크기의 컨볼루션 레이어로 분할하는 것이 좋습니다. 이는 주어진 컨볼루션 레이어 내에서 계산 복잡성을 줄이기 위한 것입니다. 예를 들어 수용 필드가 *3×3인* 컨볼루션 레이어 세 개를 서로 겹쳐서 쌓는다고 가정해 보겠습니다. 첫 번째 컨볼루션 레이어의 각 뉴런은 입력 벡터의 3×3 뷰를 갖게 됩니다. 그러면 두 번째 컨볼루션 레이어의 뉴런은 입력 벡터의 5×5 뷰를 갖게 됩니다. 그러면 세 번째 컨볼루션 레이어의 뉴런은 입력 벡터의 7×7 뷰를 갖게 됩니다. 이러한 스택은 비선형성을 특징으로 하므로 더 적은 수의 파라미터로 입력의 더 강력한 특징을 표현할 수 있습니다. 그러나 여기에는 특히 역전파 알고리즘을 사용할 때 뚜렷한 메모리 할당 문제가 따른다는 점을 이해하는 것이 중요합니다.

입력 레이어는 재귀적으로 2로 나눌 수 있어야 합니다. 일반적인 숫자 32 × 32, 64 × 64, 96 × 96, 128 × 128, 224 × 224입니다.

작은 필터를 사용할 때는 보폭을 1로 설정하고 컨볼루션 레이어가 입력의 차원성을 재구성하지 않도록 제로 패딩을 활용하세요. 사용할 제로 패딩의 양은 수용 필드 크기에서 1을 빼고 2.activation으로 나누어 계산해야 합니다.

CNN은 매우 강력한 머신 러닝 알고리즘이지만, 리소스를 엄청나게 많이 사용할 수 있습니다. 이 문제의 예를 들어 큰 이미지를 필터링할 때(128×128 이상의 이미지는 큰 것으로 간주될 수 있음) 입력이 227×227이고(ImageNet에서 보는 것처럼) 패딩이 0인 64개의 커널로 필터링하는 경우 결과는 227×227×64 크기의 활성화 벡터 3개(약 천만 개의 활성화로 계산됨) 또는 이미지당 70MB의 엄청난 메모리가 필요합니다. 이 경우 두 가지 옵션이 있습니다. 첫째, 다음과 같은 방법으로 입력 이미지의 공간 차원을 줄일 수 있습니다.

원시 이미지의 크기를 조금 덜 무겁도록 조정합니다. 또는 이 문서의 앞부분에서 설명한 모든 내용을 무시하고 필터 크기를 더 큰 간격(1이 아닌 2)으로 선택할 수도 있습니다.

위에서 설명한 몇 가지 경험 법칙 외에도 일반화된 ANN 훈련 기법에 대한 몇 가지 '요령'을 숙지하는 것도 중요합니다. 이를 위해 제프리 힌튼의 "제한된 볼츠만 머신 트레이닝을 위한 실용적인 가이드"를 읽어보시기 바랍니다.

4 결론

합성곱 신경망은 문제 영역 전체에 초점을 맞추는 대신 특정 입력 유형에 대한 지식을 활용한다는 점에서 다른 형태의 인공 신경망과 다릅니다. 따라서 훨씬 더 간단한 네트워크 아키텍처를 설정할 수 있습니다.

이 백서에서는 합성곱 신경망의 기본 개념을 개괄적으로 설명하고, 합성곱 신경망을 구축하는 데 필요한 계층을 설명하며, 대부분의 이미지 분석 작업에서 네트워크를 가장 잘 구성하는 방법을 자세히 설명합니다.

신경망을 이용한 이미지 분석 분야의 연구는 최근 들 어 다소 둔화되고 있습니다. 이는 부분적으로는 매우 강력한 머신러닝 알고리즘을 모델링하는 데 필요한 복잡성과 지식의 수준에 대한 잘못된 믿음 때문이기도 합니다. 저자들은 이 논문이 어떤 식으로든 이러한 혼란을 줄이고 초보자도 이 분야에 더 쉽게 접근할 수 있게 되기를 바랍니다.

감사

저자들은 유용한 토론과 제안을 해주신 Chuan Lu 박사와 Nicholas Dimonaco에게 감사의 말씀을 전합니다.