제프리 힌튼의 캡슐망을 풀이하다

딥러닝(deep learning)의 대부인 제프리 힌튼(Geoffrey Hinton) 교수가 최근 '캡슐망(capsule networks)'이라는 새로운 신경망과 훈련 알고리듬인 '캡슐 간 동적 라우팅(dynamic routing between capsules)'[*1]을 논문을 통해 공개했다.

지난 1979년 힌튼 교수가 아이디어를 처음 고안한 지 무려 수십 년 만에 이를 실제로 구현해, 캡슐망이 CNN(convolutional neural networks)을 대체할 수도 있다는 점에서 이번 논문은 주목할 만하다. 물론 힌튼 교수가 오늘날 전 세계에서 널리 쓰이는 수많은 딥러닝 모델과 알고리듬을 개발해온 사람이라는 명성 또한 이번 논문에 대한 딥러닝 학계의 관심을 증폭시켰다'[2].

캡슐망이 고안된 배경과 그 구조를 이해하려면 우선 CNN을 살펴볼 필요가 있다. CNN이 가진 구조적 한계를 개선한 것이 바로 캡슐망이기 때문이다.

글 | 이수경 samantha.lee@kakaobrain.com

2016년 3월 알파고와 이세돌 9단이 펼치는 세기의 대결을 두 눈으로 목도한 이후 인공지능을 제대로 공부해 봐야겠다고 결심했습니다. 인공지능 본진이자 연구소인 카카오브레인으로 걸어 들어온 이유입니다. 인공지능 기술과 이로 인해 바뀔 미래 사회를 다루는 글을 통해 사람들과 소통하고 싶습니다.

글 | 강종호 wiles.inno@kakaobrain.com

어릴 때부터 수학을 좋아했고, 학부에서 수학을 전공했습니다. 대학원에서 금융을 전공하고, 취미로 프로그래밍과 수학, 통계를 공부하다 알파고를 만났습니다. 거대한 변화의 시작임을 느끼고 그 흐름을 타고 싶어 카카오브레인에 들어왔습니다. 인공지능으로 더 행복한 세상을 만들고 싶습니다.

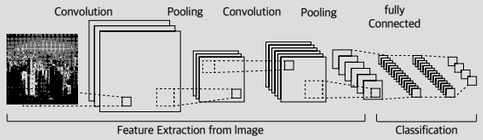
CNN이란

CNN은 데이터로부터 자동으로 특징(features)을 학습하는 대표적인 모델이다. 인간의 시각(vision) 정보 처리 방식을 흉내낸 것으로, 특히 이미지 인식과 분류에서 탁월한 성능을 낸다. 알파고의 승리도 CNN의 발전이 있었기에 가능했다. CNN의 메커니즘은 생각보다 간단하다. 입력과 가까운 층에서는 가장자리(edge), 곡선(curve)과 같은 저수준(low level) 특징을 학습한다. 점차 높은 층으로 올라 갈수록 질감(texture), 물체 일부분(object parts)과 같이 고수준(high level) 특징을 인식한다. 출력층에서는 물체의 종류를 인식하는 등 복잡한 추론을 수행한다.

CNN은 크게 세 가지 종류의 층으로 구성된다. 컨볼루션 층(convolution layer)은 이미지로부터 특징을 추출한다. 풀링 층(pooling layer)은 이미지에서 표본을 추출하는 방식으로 학습 속도를 높인다. FC 층(fully connected layer)은 최종적인 분류 작업을 담당한다.

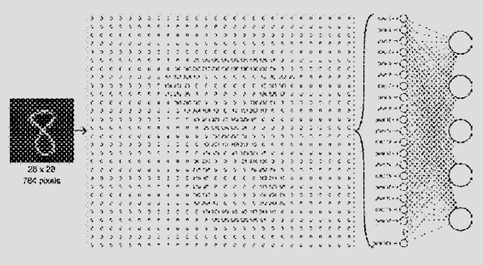
1) 컨볼루션 층

[그림 1] CNN의 구조



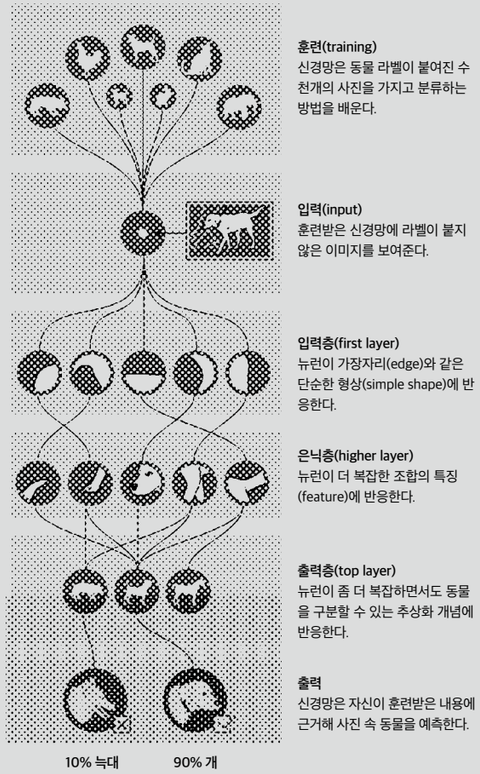
일반적인 신경망은 완전히 연결돼 있다. 모든 데이터가 하나의 신경층에서 다른 신경층으로 '전파(propagate)'된다는 의미다. 28×28처럼 작은 크기의 이미지[*3]의 경우 784(=28×28)개의 입력값만 처리하면 된다. 전체 이미지에서 특징을 학습하는 데 큰 무리가 없는 수준이다. 문제는 1024×768처럼 실제로 우리 실생활에서 사용되는 큰 이미지를 학습시키는 일이다. 예를 들어, 10^4개의 입력값과 100개의 특징을 학습한다면 10^6개의 가중치(weight)를 학습해야 한다. 학습 과정에서의 계산 또한 28×28 이미지와 비교해 약 100배 이상의 시간이 걸린다.

[그림 2] 이미지 | 28×28 크기의 손필기 숫자를 신경망에 넣는(feed) 방법[*4]



다행스러운 건 이미지가 정적(stationary)이라는 점이다. 이미지의 한 부분의 통계치가 다른 부분과 비슷하다는 의미로, 한 영역에서 학습한 특징은 다른 영역에서도 유사한 패턴을 찾는데 활용할 수 있다. 완전히 연결된 신경망 대신 CNN을 사용할 수 있는 이유다.

[그림 3] 신경망이 사진에서 개를 인식하는 방법



컨볼루션 층에서는 작은 이미지 영역인 패치(patch)를 큰 이미지 위에 빙빙 돌려가면서(convolve) 각각 다른 특징 활성값(activation value)을 얻는다. 이 패치는 특징을 감지한다는 점에서 특징 추출기(feature detector) 또는 커널(kernel), 필터(filter)라고 부르기도 한다.

2) 풀링 층

캡슐망과 관련지어 볼 구성요소는 풀링 층[*5]이다. 풀링을 활용하면 학습 시간을 줄이면서도, 이미지 구성 요소의 위치 변화에 더 잘 대응할 수 있다. 일반적으로 사용되는 맥스 풀링(max pooling)은 주변 영역의 추론 결과값 중 최댓값만을 상위층으로 보낸다. 특징 탐색 영역(feature map)은 1/4로 축소되고, 위층에서의 특징 추출 및 추론에 대한 부담은 크게 줄어든다. 더불어 분류 작업에 유리한

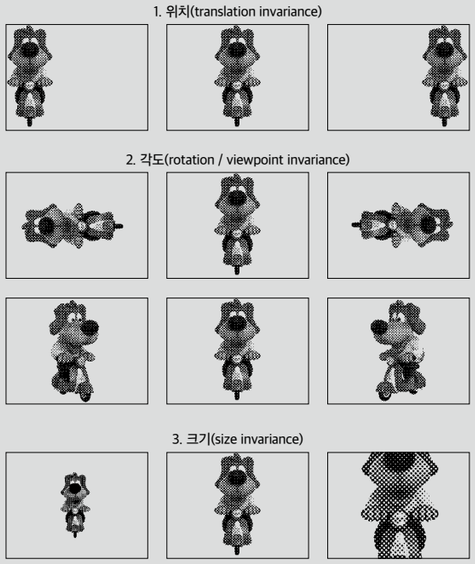
불변성질(invariance)을 얻을 수 있는 장점도 있다. 예를 들어, 얼굴의 방향이나 각도에 따라 특징이 서로 다르게 추출되면 상위층에서 이를 제대로 인식할 수 없는데, 맥스 풀링은 위치에 상관없이 눈, 코, 입을 인식할 수 있도록 해준다.

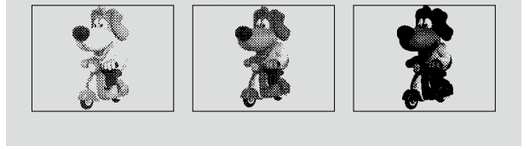
CNN이 가진 구조적 한계

탐색 속도를 높이고자 고안한 맥스 풀링은 아이러니하게도 CNN의 취약점으로 작용한다. 맥스 풀링을 거치면 이미지 구성 요소의 공간 관계에 관한 정보를 잃는다. 예로 들면, 사람 얼굴을 인식하는 CNN은 피카소의 작품 속(기형적인) 대상(object)을 사람으로 인식한다. 맥스 풀링을 통해 눈, 코, 입, 귀의 상대적인 위치, 방향과 상관없이 특징을 추출하기 때문이다.

종합하자면, CNN은 풀링을 통해 위치(translation)와 관계없이 객체를 동일하게 인식하지만, 방향(orientation)이나 비율(proportion)이 달라지면 서로 다른 객체로 인식한다. 아울러 물체를 바라보는 시점(viewpoint) 변화에 유독 취약하다. 이미지의 각도나 크기가 변형될 경우 해당 이미지를 제대로 인식하지 못한다는 의미다. 이런 이유로 힌튼 교수도 오래 전부터 맥스 풀링에 관해 회의적인 모습을 보이며 이를 재앙(disaster)이라 표현하기도 했다. 이를 개선하고자 다양한 방식으로 변형한 이미지를 학습 데이터로 활용하는 방법(data augmentation)을 사용하지만, 대신 학습 시간이 증가하는 단점이 있다.

[그림 4] 다양한 방식으로 변형하여 학습 데이터로 활용한 이미지





교란 샘플(adversarial example)은 CNN의 약점을 보여주는 또 다른 예다. 이 샘플은 사람의 판단에는 영향을 주진 않으나 머신러닝에서 오분류(misclassification)를 낳는다. '1픽셀로 딥 뉴럴 네트워크 속이기(One pixel attack for fooling deep neural networks)' 논문에서는 단 하나의 픽셀만 달라져도 대상을 제대로 인지하지 못한다는 결과를 보여주기도 했다.

신경망에 3D 세계를 구현하다

힌튼 교수는 렌더링(rendering)에서 CNN의 한계를 극복할 아이디어를 얻었다. 렌더링은 매시(mesh)[*6] 객체를 포즈(pose, 위치와 방향) 정보를 활용해 시각적 이미지를 구성하는 것을 의미한다. 힌튼 교수는 인간의 뇌가 바로 이 렌더링의 역과정(inverse graphics)을 통해 영상을 인식한다고 봤다. 즉, 눈으로 획득한 시각 정보를 계층적인 표현으로 해체(deconstruct)한 뒤, 사전에 습득한 지식과 매칭해 물체의 종류와 포즈 정보를 역추론한다는 설명이다. 이를 위해서는 신경망이 불변성질(invariance) 대신, 등가성질(equivariance)을 가져야 한다고 봤다. 이미지 속 물체가 적절히 변환(위치, 방향 등)되면 추론 결과도 이에 상응해서 변해야 한다는 것이다.

그래서 힌튼 교수는 CNN 층을 깊숙이 쌓는 대신, 캡슐을 계층적으로 쌓아 올린 캡슐망을 고안했다. 캡슐은 여러 신경망 층으로 구성된 단위 요소다. 그리고 중첩된 계층간 동적 라우팅(dynamic routing)하는 방식을 고안했다.

*scalar-out layer→vector-out layer
*max-pooling→dynamic routing

기존 신경망에서는 각각의 뉴런이 독립적으로 동작한다. 반면, 캡슐망에서는 뉴런들의 그룹인 캡슐이 단위 요소다. 뉴런의 출력값이 스칼라(scalar)이지만, 여러 뉴런으로 이루어진 캡슐의 출력값은 벡터가 된다. 그리고 이 벡터의 크기는 어떤 개체가 존재할 확률을, 벡터의 방향은 그 개체의 성질을 표현한다. 여기서 새롭게 고안한 비선형 함수인 '스쿼싱 함수(squashing function)'가 벡터 전체에 적용된다. 스쿼싱 함수를 통해 벡터의 크기는 1을 넘지 않게 된다. 개체가 존재할 확률을 효과적으로 나타내는 장치인 셈이다.

[수식 1] 스쿼싱 함수

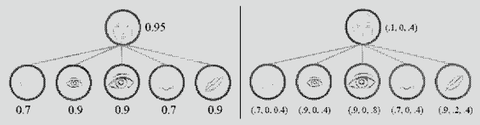
$$V_j = \frac{\|s_j\|^2}{1+\|s_j\|^2}\frac{s_j}{\|s_j\|}$$

일반적인 CNN에서는 층을 깊게 쌓아야만 물체의 특징을 인식할 수 있다. 반면 캡슐망은 층을 깊게 쌓지 않아도 된다. 캡슐이 출력하는 벡터 자체에 많은 정보가 담겨 있어서다. 아울러, 아래층 캡슐의 출력 벡터를 어떤 가중치로 사용할지는 맥스 풀링이 아닌 동적 라우팅 알고리듬이 결정한다.

캡슐망에서 캡슐의 모든 부모 캡슐은 캡슐의 출력을 예측한다. 예측의 가중합(weighted sum)은 스쿼싱 함수를 거쳐 캡슐의 출력이 된다. 그리고 이 예측 벡터와 자식 캡슐의 출력 간 '내적(scalar product)'이 큰 (즉 연관성이 큰) 부모 캡슐과 결합이 강화된다(가중치가 커진다). 학습이 잘 됐을 때 정렬된 이미지가 들어온다면, 최종 출력 벡터의 방향은 이미지의 회전이나 크기 같은 정보를 포함한다(여기서 최종 출력 벡터는 출력층의 캡슐들이 출력하는 벡터 중에서 크기가 가장 큰 벡터를 의미한다). 그리고 이와 강하게 결합된 부모 캡슐의 예측 벡터들도 비슷한 방향을 가진다. 만약 눈, 코, 입의 크기와 각도가 제멋대로인 얼굴이 입력으로 들어온다면, 이러한 요소를 감지한 벡터들의 방향은 서로 어긋나게 된다. 방향이 어긋나면 자연스럽게 그 합의 크기는 감소한다. 이는 얼굴이라고 판단하는 확률이 낮아지는 셈이다. 단순히 객체를 구성하는 각 요소의 존재만으로 객체를 인지하는데 그치지 않고, 각 요소 간 상관관계까지 고려해 객체를 인지한다는 의미다.

쉽게 이야기하면, [그림 5]의 왼쪽처럼 CNN이 눈과 코, 입이 있으면 얼굴이라고 인식하지만, [그림 5]의 오른쪽처럼 캡슐망은 두 눈이 인접해 있고 눈 사이 아래쪽에 코가 있으며, 눈 아래 입이 있으면 얼굴이라고 인식한다.

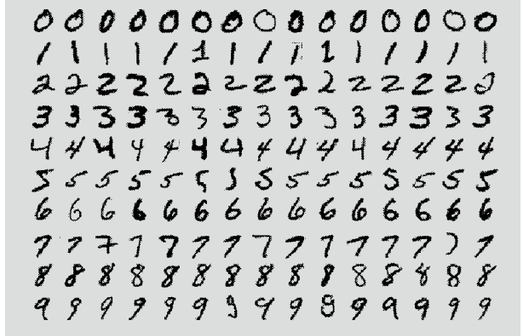
[그림 5] 좌측 그래프에서는 개별 부위의 존재 여부만 가지고 사람의 얼굴이라고 인식한다. 반면, 우측 그래프는 개별 부위의 존재 여부뿐만 아니라 상대적인 위치값까지 고려해 사람의 얼굴을 인식한다[*7].



새롭게 고안된 캡슐망을 기반으로 MNIST[*8] 데이터셋을 훈련시켜 본 결과, 최신 CNN 대비 에러율을 45%까지 줄였다. 아울러 화이트박스(white box)[*9] 교란 공격(adversarial attack)에 대해서도 보다 효과적으로 저항했다.

물론 캡슐망을 현업에 적용하는 데는 다소 시간이 걸릴 것으로 보인다. 이번 논문이 최신의 딥러닝 신경망을 소개하고 있으나 그 성능 검증은 아직 끝나지 않았다. 우선, MNIST 데이터셋(28×28, 6만개 흑백 이미지)에서는 잘 동작하더라도 이미지넷(256×256 이상, 100만개 컬러 이미지)과 같은 매우 방대한 크기의 데이터셋에서도 비슷한 성능을 낼지는 미지수다. 아울러 학습에 걸린 시간 또한 기존 CNN보다 더 긴 시간이 걸렸다. 정확도에 큰 차이가 없다면 컴퓨팅 자원을 더 소모하는 캡슐망을 쓸 이유가 없다.

[그림 6] MNIST 데이터셋



캡슐망은 이제서야 그 존재를 알렸다. 힌튼 교수가 오랜 시간 가슴 속에 품어온 아이디어를 실제로 구현해내는 단계에 이르렀다. 물론 아직 더 많은 검증은 필요하나 희망적인 부분도 있다. 캡슐망은 기존 CNN보다 사람이 사물을 인식하는 방식에 더 가깝다는 이유에서다. 이를 통해 합리적인 결과를 내지 못하는 아이디어에 그치는 것이 아니라 딥러닝의 새 시대를 여는 시발점이 되기를 기대해본다.

***1** 참고 | 이 논문의 제1저자는 사라 사보(Sara Sabour), 제2저자는 니콜라스 프로스트(Nicholars Frost)다. 논문 저자의 이름 순서와 내용, 인용을 분석해보자면 해당 연구는 구글브레인팀에서 진행하고 힌튼 교수는 검수를 맡은 것으로 추측된다. 그로부터 8일 후 공개된 'EM 라우팅을 활용한 행렬 캡슐(Matrix Capsules with EM Routing) 논문에는 익명의 저자들(anonymous authors)로 기재돼 있으나, 학계에서는 힌튼 교수도 저자로 참여했을 것으로 추정하고 있다. ***2** 참고 | http://v.sports.media.daum.net/v/20160308060204892 ***3** 참고 | 모든 이미지는 픽셀값의 행렬로 표시할 수 있다. 행렬 속 픽셀값의 범위는 0~255사이다. 그리고 R(Red), G(Green), B(Blue) 3종의 채널로 구성된다. ***4** 참고 | http://ml4a.github.io/ml4a/neural_networks/ ***5** 참고 | 이미지의 특정 영역을 요약한다는 의미에서 표본추출(sub-sampling) 또는 리사이징(resizing)이라 표현하기도 한다. 풀링에는 평균값을 계산하는 '에버리지 풀링 (average pooling)' 등 다양한 방법이 존재한다. 최근에는 실제로 더 나은 결과를 보여주는 '맥스 풀링(max pooling)'이 주로 쓰인다. ***6** 참고 | 3차원 그래픽에서 다면체를 구성하는 다각형(pologon)과 정점(vertex)들의 집합 ***7** 참고 | https://jhui.github.io/2017/11/03/Dynamic-Routing-Between-Capsules/ ***8** 참고 | 필기체 숫자를 분류하는 머신러닝 모델을 평가하기 위한 표준화된 데이터. 18*18 픽셀 크기의 6만개 훈련 데이터와 1만개 테스트 데이터로 구성돼 있다. ***9** 참고 | 응용 프로그램의 내부 구조와 동작을 검사하는 소프트웨어 테스트 방식