[8차시]

딥러닝 머신러닝 관련 주요 개념 정리

CNN 참조 사이트: https://brunch.co.kr/@kakao-it/158

1. CNN과 캡슐망(Capsnet)

- CNN은 데이터로부터 자동으로 특징(features)을 학습하는 대표적인 모델이다.
- 인간의 시각(vision) 정보 처리 방식을 흉내낸 것으로, 특히 이미지 인식과 분류에서 탁월한 성능을 낸다.
- CNN의 메커니즘은 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 구성)

• 컨볼루션층(convolution layer)

이미지로부터 특징을 추출한다.

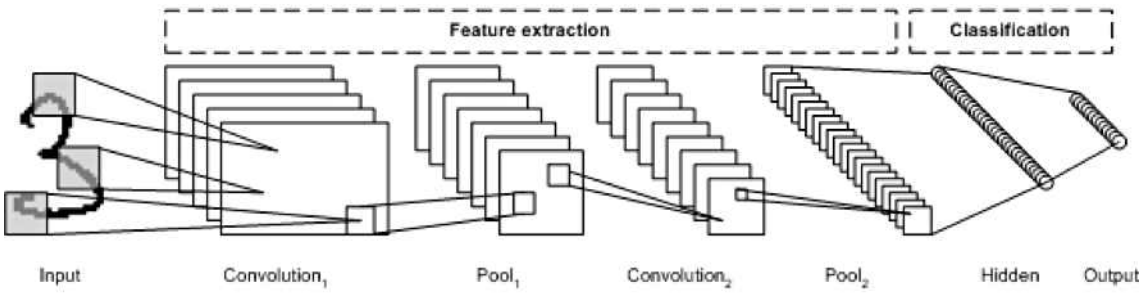
• 풀링층(pooling layer)

이미지에서 표본을 추출하는 방식으로 학습 속도를 높인다.

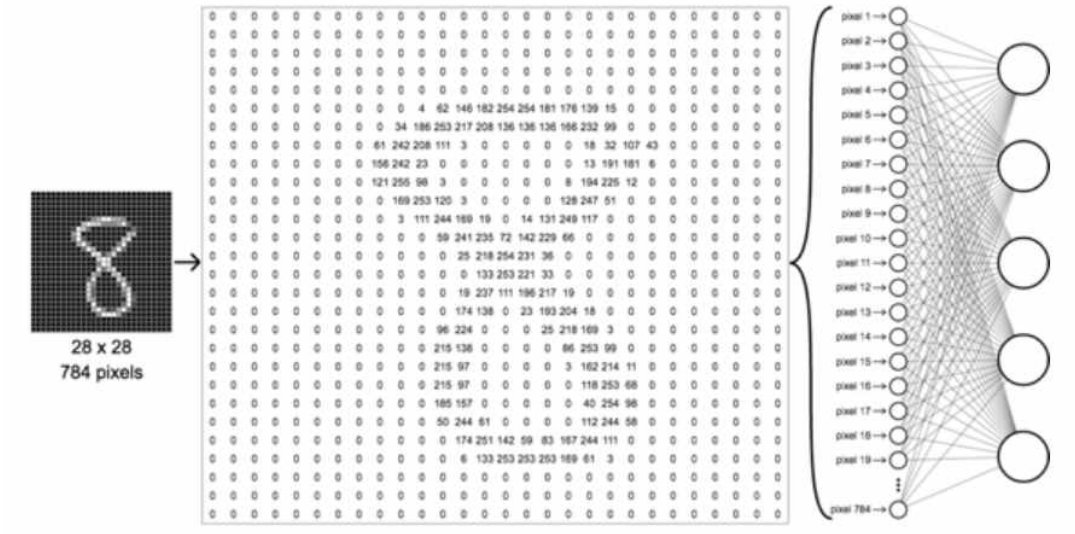
• FC 층(fully connected layer)

최종적인 분류 작업을 담당한다.

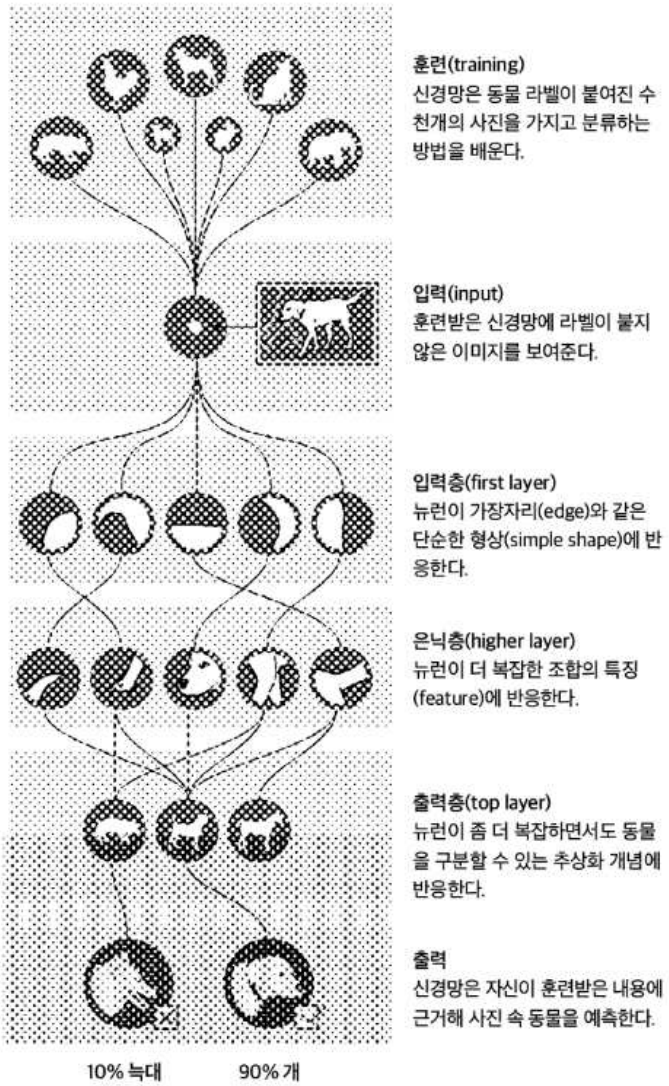
1) 컨볼루션층(convolution layer)



- 일반적인 신경망은 완전히 연결돼 있다.
- 모든 데이터가 하나의 신경층에서 다른 신경층으로 '전파(propagate)' 된다는 의미다.
- 28×28처럼 작은 크기의 이미지*3의 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 일**이다. 예를 들어, 10,000 개의 입력값과 100개의 특징을 학습한다면 1,000,000 개의 가중치(weight)를 학습해야 한다. 학습 과정에서의 계산 또한 28×28 이미지와 비교해 약 **100 배 이상의 시간이 걸린다.**



(이미지 28*28 크기의 손필기 숫자를 신경망에 넣는 방법)



(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.
- **패치(patch)**는 특징을 감지한다는 점에서 특징 추출기(feature detector) 또는 **커널(kernel), 필터(filter)**라고 부르기도 한다.

2) 풀링(pooling)층
- 풀링을 활용하면 학습 시간을 줄이면서도, 이미지 구성 요소의 위치 변화에 더 잘 대응할 수 있다.
- 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
- 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.
- CNN은 풀링을 통해 위치(translation)와 관계없이 객체를 동일하게 인식하지만, 방향(orientation)이나 비율(proportion)이 달라지면 서로 다른 객체로 인식한다.
- 또한 물체를 바라보는 **시점(viewpoint) 변화에 유독 취약**하다. 이미지의 각도나 크기가 변형될 경우 해당 이미지를 제대로 인식하지 못한다.
- 이를 개선하고자 다양한 방식으로 변형한 이미지를 학습 데이터로 활용하는 방법(**data augmentation**)을 사용하지만, 대신 **학습 시간이 증가하는 단점**이 있다.

(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.

◑ 신경망에 3D 세계를 구현하다.
- 힌튼 교수는 렌더링(rendering)에서 CNN의 한계를 극복할 아이디어를 얻었다.
- 렌더링은 매시(mesh)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.
※등가성질: 이미지 속 물체가 적절히 변환(위치, 방향 등)되면 추론 결과도 이에 상응해서 변해야 한다는 것

- 힌튼 교수는 CNN 층을 깊숙이 쌓는 대신, 캡슐을 계층적으로 쌓아 올린 **캡슐망(Capsnet)**을 고안했다.
- **캡슐은 여러 신경망 층으로 구성된 단위 요소**다. 그리고 중첩된 계층간 **동적 라우팅(dynamic routing)하는 방식**을 고안했다.

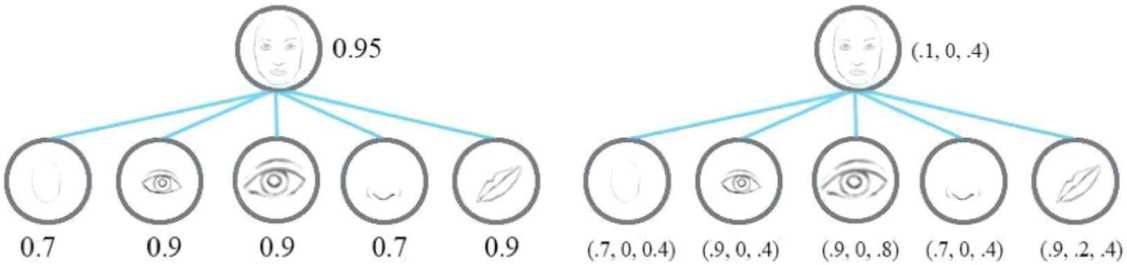
*scalar-out layer→vector-out layer
*max-pooling→dynamic routing

- 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.

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

(스쿼싱 함수)

- 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.
- 아래층 캡슐의 출력 벡터를 어떤 가중치로 사용할지는 맥스 풀링이 아닌 **동적 라우팅 알고리듬이 결정**한다.
- 캡슐망에서 캡슐의 모든 부모 캡슐은 캡슐의 출력을 예측한다.
- 예측의 가중합(weighted sum)은 스쿼싱 함수를 거쳐 캡슐의 출력이 된다.
- 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 인지**한다는 의미다.



- 위 그림에서 왼쪽처럼 CNN이 눈과 코, 입이 있으면 얼굴이라고 인식하지만, 오른쪽처럼 캡슐망은 두 눈이 인접해 있고 눈 사이 아래쪽에 코가 있으며, 눈 아래 입이 있으면 얼굴이라고 인식한다.

- 새롭게 고안된 캡슐망을 기반으로 MNIST데이터셋을 훈련시켜 본 결과, 최신 CNN 대비 에러율을 45%까지 줄였다.
- 아울러 화이트박스(white box)교란 공격(adversarial attack)에 대해서도 보다 효과적으로 저항했다.
- 하지만 캡슐망의 성능 검증은 아직 끝나지 않았다. 우선, MNIST 데이터셋(28×28, 6만개 흑백 이미지)에서는 잘 동작하더라도 이미지넷(256×256이상, 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 캡슐망보다는 CNN을 사용하는게 좋다.

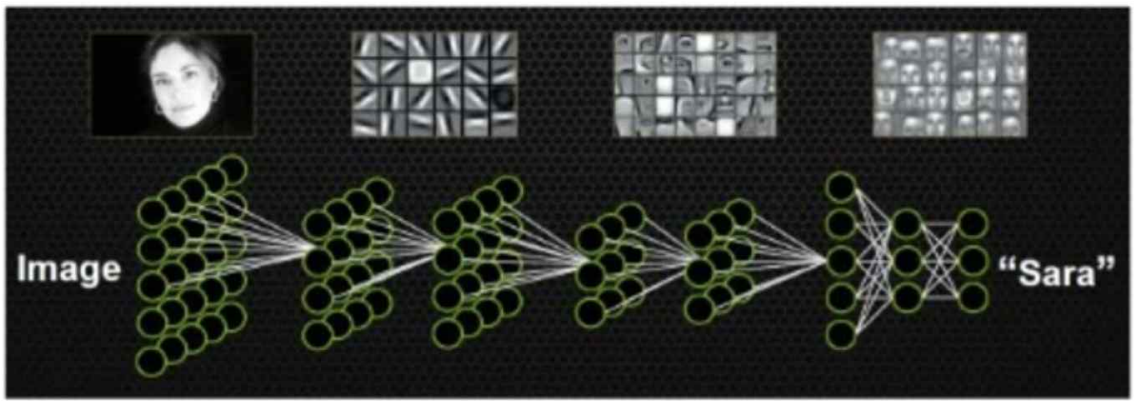
[캡슐 네트워크(캡스넷 - Capsnet)]

- 제프리 힌튼(geoffrey hinton)은 다이나믹 루팅(Dynamic routing)을 사용하여 CNN의 문제점을 극복하는 방법을 제시
- 참조 사이트: https://jayhey.github.io/deep%20learning/2017/11/28/Caps

1. Problems of CNN

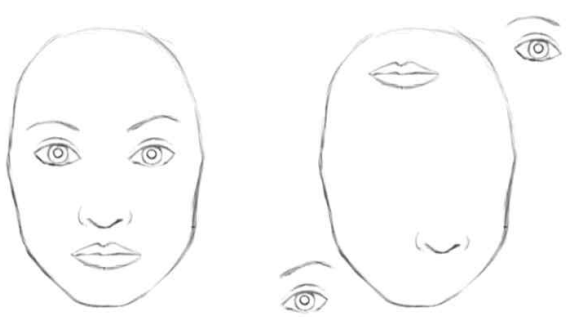
1) Convolution filter의 한계:

- CNN은 콘볼루션 필터들은 이미지 픽셀에서 중요한 부분이 무엇인지 디텍트를 한다.
- 처음에는 간단한 특징(simple feature - edge, color, …)들을 포착하지만 점점 higher level로 갈수록 더욱 복잡한 특징(complex feature)들을 잡게 된다.
- 그리고 가장 최상위에 위치한 top layer에서 분류를 하게 된다.
- 그러나 higher layer은 단순히 lower layer들의 가중합(weighted sum)이라는게 가장 큰 문제점, 가중합은 simple feature와 complex feature간의 **위치 관계를 전혀 고려하지 않다는 것**



2) Max-pooling

- CNN은 이런 문제점을 풀기 위해 **max pooling방법**을 도입
- Max-pooling은 피쳐맵들의 spatial(공간) size를 줄여주어 higher layer들의 시야각(field of view)을 넓혀주는 역할을 한다.
- FPS 게임의 경우 예: 보통 FOV(field of view)옵션이 있고, 이 옵션을 높여주면 시야각이 넓어져서 한 눈에 볼 수 있는 공간이 커진다. 즉, FOV값을 가장 낮게 설정하면 노란색 직사각형이 시야가 되는 것이고, FOV 값이 크다면 파란색 직사각형 이상의 시야로 볼 수 있다.



- 풀링을 활용하면 한 눈에 보는 시야가 넓어지고 어느 정도 spatial한 정보들을 가져갈 수 있다.
- 하지만 가장 활성화되는 feature detector의 위치 정보들은 놓치고 있다.
- 위 오른쪽 그림은 같은 얼굴이지만 눈 코 입 위치가 다르다. **CNN같은 경우는 둘 다 얼굴이라고 인식하는게 문제**

■ 풀링이 좋지 않은 4가지 이유

가) 인지 심리학 관점에서 CNN은 맞지 않다

- 위 그림을 봤을 때 사람들은 "자전거 선수가 매우 빠른 속도로 자전거를 타고 있다"라고 인식을 한다. 우리가 보는건 간단히 얘기해도 사람, 사이클 복장, 자전거, 도로 등인데 어떻게 이를 조합해서 "자전거 선수가 매우 빠른 속도로 자전거를 타고있다"라고 인식을 하는지는 여전히 인지심리학에서도 풀리지 않은 문제. 확실한건 사람, 사이클 복장, 도로, 자전거 등을 보고 저런 방식으로 인식을 한다는 점인데 **Pooling은 이를 전혀 설명하지 못하고 있다.**

나) 잘못된 문제를 풀고 있음

invariance(불변량,변하지 않는)가 아닌 equivariance(등변 량 맵, invariance를 일반화 한것)가 필요하다는 것

다) 내재되어있는 선형적인 구조를 사용하지 않는다(Fails to use underlying linear structure)

같은 물체를 보더라도 여러 방면에서 봤을 때 이 물체는 pixel intensity 공간에서 매우 비선형적이다. 이를 globally linear한 곳으로 변환시킨다면 대량의 extrapolation(보외법: 원래의 관찰 범위를 넘어서서 다른 변수와의 관계에 기초하여 변수의 값을 추정하는 과정, 관찰된 값들 사이의 추정치를 만들어내는 보간법과 비슷하지만 보외법은 더 큰 불확실성과 무의미한 결과 생성에 대한 더 높은 위험에 종속된다.)이 가능

즉, 물체의 다양한 모습을 추정이 가능한데, **pooling에는 이러한 extrapolation과정이 없다.**

라) Dynamic routing과는 어울리지 않음

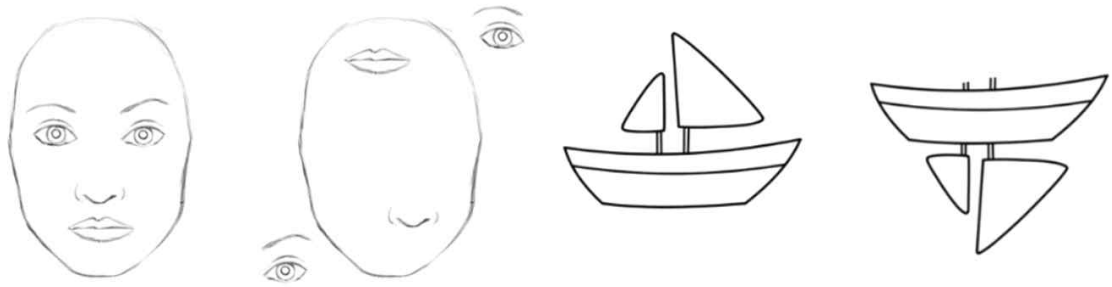
풀링에는 캡스넷의 가장 큰 특징 중 하나인 dynamic routing을 사용할 수 없다.

■ Capsules

- 우리가 세상을 이해할 때 “entity(실재)”와 “entity의 property(개체)”로 이해한다고 할 수 있다. 위에서 설명드린 자전거 그림을 다시 생각해 보면 entity는 “자전거를 타고 있는 사람”이고 property는 “자전거, 도로위에서 넘어질 듯 달리는 모습, 속도, 사이클복장을 입은 사람, 텍스쳐 등등”이다.
- CNN은 이러한 entity들을 표현하기에는 너무나도 구조의 수가 부족하다. CNN의 neuron과 layer들로는 entity를 표현을 할 수 없다.
- 다이나믹 루팅을 거친다면 최종적은 캡슐은 다음과 같은 것들을 나타 낼 수 있다.

1) Advantage of capsules(캡슐의 장점)

- 캡슐은 “고차원 공간에서의 우연의 일치는 매우 적다” 라는 가정
- 아래 그림에서 왼쪽 얼굴 그림을 보시면, 눈, 코, 입이 제자리에 붙어있는 얼굴이랑 따로 놀고 있는 얼굴이 있다. CNN의 경우 설사 눈, 코, 입이 있다는 사실을 포착하더라도 이들의 부위를 따로 따로 인식을 하지 다같이 공간적인 - 계층적인 관계를 고려하지 않는다. 제대로 된 얼굴 데이터만 가지고 학습하더라도 이목구비가 따로 노는 얼굴도 얼굴로 인식하는 문제점이 생기게 된다. 오른쪽 그림에서도 거꾸로 된 돛단배를 같은 돛단배로 인식하는 문제가 생기게 된다는 것



- 캡스넷(CapsNet)의 경우 이러한 entity를 고차원 공간에서 바라보게 되는데, 이렇게 바라보면 눈, 코, 입의 **공간적인 관계**까지 고려하게 된다. 고차원 공간에서조차 제대로 된 얼굴이 나왔다면 이건 얼굴이 맞다고 판단할 수 있다는 뜻이 된다.

- 맥스 풀링은 학습 가능한 파라미터가 존재하지 않으며 단순히 가장 큰 특징적인 값만 뽑아내서 FOV를 넓히는 것 밖에 못한다는 것, 하지만 캡슐은 벡터로 이루어져 있기에 다이나믹 루팅(dynamic routing)을 통해 오브젝트 파트(눈,코,입 등)들의 상대적 위치까지 조합할 수 있다.

2) 캡슐의 특징정리

- 하위 캡슐은 다이나믹 루팅 과정을 통해 이를 가장 잘 처리할 수 있는 상위 캡슐로 연결
- 하위 캡슐에서는 local information이 “place-coded”(유지 된다) 된다.
- 상위 캡슐에서는 훨씬 더 많은 positional information이 “rate-coded”(조합 된다)된다.

다이나믹 루팅 과정을 통해서 캡슐은 더 복잡한 entity를 훨씬 자유롭게 표현