Siamese Network(샴 네트워크)

조사 날짜 : 9/25

작성자 : 심 영민

정의

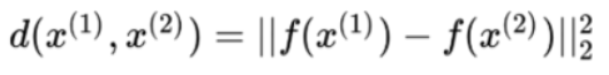
2개의 입력에 대해 독립적으로 2개의 합성곱 신경망을 실행한 뒤 비교하는 아이디어

구조

기존의 Convolution network를 통해서 Feature map을 뽑아낸다.

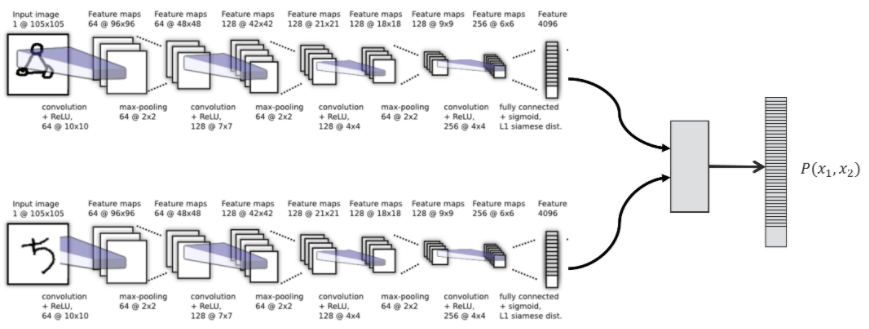
2개의 이미지에서 feature map을 뽑아낸 후 거리를 계산

거리가 작다면 두 사람은 비슷하고 거리가 크다면 두 사람이 다른 사람이라는 것



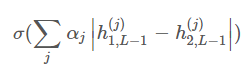
거리는 두 vector 사이의 노름으로 정의

* 두 네트워크에 두 사진을 입력으로 넣고 합성곱 신경망으로 인코딩
* 만약 두 사람이 비슷한 사람이라면 인코딩 사이의 거리 값이 작아야한다.
* 만약 두 사람이 다른 사람이라면 인코딩 사이의 거리 값이 커져야한다.
* 위 조건을 만족시키도록 학습을 시켜야한다.



각 네트워크를 거쳐 나온 4096개의 Feature를 가지고 L1 norm(absolute distance)을 구한다.

아래 식을 보면 αj는 트레이닝 과정에서 자동으로 학습되는 파라미터로 각 Feature에 대한 거리 값에 곱해준다. σ는 시그모이드 활성함수



이 값을 다시 fully connected layer를 거쳐 시그모이드 함수로 단일 값을 받아낸다.

1이면 같은 클래스, 0이면 다른 클래스.

여기 있는 fully connected layer가 파라미터 개수의 96%를 차지. 여기서 나온 예측 값과 타켓에 대하여 binary cross entropy를 이용한 로스를 구함.



학습에서 사용하는 손실함수(loss function)가 위와 같다면, 실제 one-shot task를 할 때는 다음과 같은 방식으로 진행. test image를 한 네트워크에 집어넣고, 다른 네트워크에는 support set에 있는 이미지를 하나씩 집어넣음. 여기서 가장 큰 확률 값을 가지는 이미지를 같은 클래스로 분류하는 것입니다. 식은 아래와 같음.



karse로 간단하게 모델 구조를 짜면

def W\_init(shape,name=None):

"""Initialize weights as in paper"""

values = rng.normal(loc=0,scale=1e-2,size=shape)

return K.variable(values,name=name)

def b\_init(shape,name=None):

"""Initialize bias as in paper"""

values=rng.normal(loc=0.5,scale=1e-2,size=shape)

return K.variable(values,name=name)

input\_shape = (105, 105, 1)

left\_input = Input(input\_shape)

right\_input = Input(input\_shape)

#build convnet to use in each siamese 'leg'

convnet = Sequential()

convnet.add(Conv2D(64,(10,10),activation='relu',input\_shape=input\_shape,

kernel\_initializer=W\_init,kernel\_regularizer=l2(2e-4)))

convnet.add(MaxPooling2D())

convnet.add(Conv2D(128,(7,7),activation='relu',

kernel\_regularizer=l2(2e-4),kernel\_initializer=W\_init,bias\_initializer=b\_init))

convnet.add(MaxPooling2D())

convnet.add(Conv2D(128,(4,4),activation='relu',kernel\_initializer=W\_init,kernel\_regularizer=l2(2e-4),bias\_initializer=b\_init))

convnet.add(MaxPooling2D())

convnet.add(Conv2D(256,(4,4),activation='relu',kernel\_initializer=W\_init,kernel\_regularizer=l2(2e-4),bias\_initializer=b\_init))

convnet.add(Flatten())

convnet.add(Dense(4096,activation="sigmoid",kernel\_regularizer=l2(1e-3),kernel\_initializer=W\_init,bias\_initializer=b\_init))

#encode each of the two inputs into a vector with the convnet

encoded\_l = convnet(left\_input)

encoded\_r = convnet(right\_input)

#merge two encoded inputs with the l1 distance between them

L1\_distance = lambda x: K.abs(x[0]-x[1])

both = merge([encoded\_l,encoded\_r], mode = L1\_distance, output\_shape=lambda x: x[0])

prediction = Dense(1,activation='sigmoid',bias\_initializer=b\_init)(both)

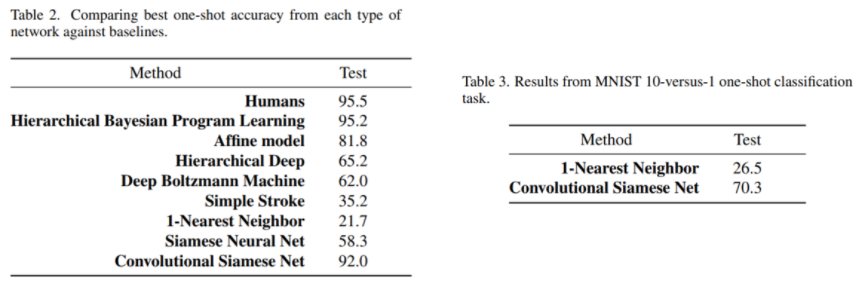
siamese\_net = Model(input=[left\_input,right\_input],output=prediction)

optimizer = Adam(0.00006)

siamese\_net.compile(loss="binary\_crossentropy",optimizer=optimizer)

Results

학습률(Learning rate)이라던가 epoch수 등등은 그냥 넘어가고



테스트 정확도를 봐도 굉장히 좋은 결과를 얻은 것을 확인.

왼쪽은 Omniglot 데이터셋을 대상으로 one-shot performace를 여러 모델로 측정한 결과입니다. 사람과 비교해도 큰 차이가 나지 않는 92%의 정확도를 보여줍니다. 오른쪽은 MNIST 데이터셋을 대상으로 1-NN과 비교실험한 결과인데, 70.3%라는 준수한 성능을 보여주고 있습니다.

참고 : https://sorenbouma.github.io/blog/oneshot/

참고 : https://jayhey.github.io/deep%20learning/2018/02/06/saimese\_network/

iPhone FaceID

TrueDepth 카메라는 30,000개 이상의 보이지 않는 점을 투영하고 분석하여 정확한 얼굴 데이터를 수집

이를 통해 얼굴의 심도 맵을 만들고 얼굴의 적외선 이미지도 촬영

Secure Enclave 내에서 보호되는 A11, A12 Bionic, A12X Bionic 및 A13 Bionic 칩의 일부 신경 엔진이 심도 맵과 적외선 이미지를 수식으로 변환한 후 해당 수식을 등록된 얼굴 데이터와 비교

화장을 하거나 수염을 기르는 등 외모의 변화에 자동으로 맞추어 조정됩니다. 턱수염을 완전히 면도하는 경우와 같이 외모가 크게 바뀌면 Face ID는 얼굴 데이터를 업데이트하기 전에 암호를 사용하여 사용자의 신원을 확인합니다. Face ID는 모자, 스카프, 안경, 콘택트렌즈, 다양한 선글라스를 착용해도 작동하도록 설계되었으며 실내외는 물론 암흑 속에서도 작동하도록 설계

Face ID를 사용하려면 기기를 바닥에 놓거나 손으로 자연스럽게 잡고 있는 상태에서 TrueDepth 카메라의 초점이 얼굴을 향하도록 해야 합니다. TrueDepth 카메라의 시야 범위는 전면 카메라로 사진을 찍거나 FaceTime으로 통화할 때와 비슷합니다. Face ID는 팔을 펴서 기기를 잡고 볼 때 가장 잘 작동합니다(얼굴에서 25~50cm 떨어진 거리)

TrueDepth 적외선 카메라

이스라엘 기업인 프라임센스(PrimeSense)를 3억4천500만 달러에 인수한 결과로 프라임센스가 트루뎁스 카메라를 개발하는데 지대한 공헌을 했다.

프라임센스는 3D 센싱(Sensing) 기업으로 3D 환경, 즉 3차원 증강현실과 가상현실(3D AR/VR)을 구축하는 원천기술을 확보했다. 3D 동작을 감지하는 칩(Chip)이 주특기이다. 마이크로소프트(MS)의 동작인식 기술인 키넥트(Kinect)와 유사하다.

참고 : https://patents.google.com/patent/US20190370529A1/en?oq=20190370529

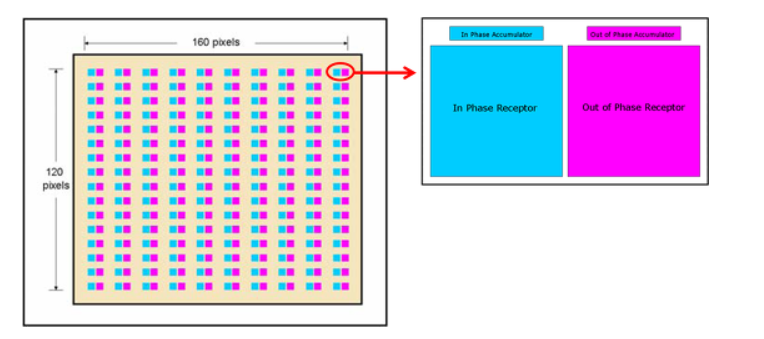
Intel RealSense

인간을 비롯한 동물들은 두 눈을 이용하여 사물을 바라보고 깊이감을 인지할 수 있다. 이를 증명하는 나름 유명한 실험이 한 가지 있다. 한쪽 눈을 감고 두 손가락을 맞대려 하면 쉽게 맞닿지 않는다는 것이 바로 그것이다. 이러한 일이 생기는 이유는 한쪽 눈만으로는 공간감을 인지할 수 있고 우리가 생각하기에는 공간을 인지하고 있다고 생각하지만 실제로는 한쪽 눈으로는 2차원적인 이미지만을 인지할 수밖에 없기 때문이다. 인텔 리얼센스는 이러한 원리를 이용하여 2개 혹은 3개의 카메라를 이용해 공간을 인지하고 사물의 입체감과 깊이감, 거리감, 크기 감을 인지할 수 있다.

리얼센스 스냅샷은 2개 이상의 카메라를 이용하여 사물을 인지하는데 비해 리얼센스 3D 카메라는 적외선을 이용하여 사물을 인지한다. 리얼센스 카메라는 한 개의 메인 카메라와 한 개의 적외선 프로젝터, 한 개의 적외선 카메라로 구성되어 있다. 기본적인 원리는 적외선을 이용하여 사물의 굴곡과 거리를 감지하여 3D 데이터를 추출해내고 그 위에 메인카메라로 촬영한 2D 이미지를 덧씌워 사물의 입체적인 이미지를 촬영해내는 것이다. 보통 인텔 리얼센스라 하면 바로 적외선을 이용하는 리얼센스 카메라 쪽을 말하며 사물을 정확하게 인지할 수 있는 인지컴퓨팅 기술 중 하나이다.

TOF





각각의 셀이 2개씩의 receptor 쌍으로 구성

카메라로부터 강한 빛을 전방에 쏜 후에 반사되어 돌아오는 빛을 감지하여 거리를 측정하는 방식

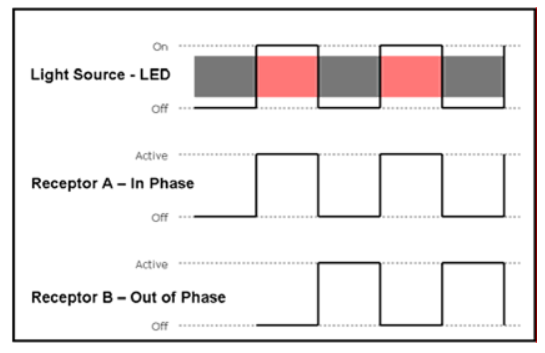
각 Receptor들은 그림과 같이 in phase receptor (하늘색)와 out phase receptor (붉은색) 쌍으로 구성되어 있는데, in phase receptor는 in phase에만 활성화되어 빛을 감지하고, out phase receptor는 out phase에만 활성화되어 빛을 감지한다.

빛을 쏘는 부분은 렌즈 주변에 LED들에서 빛이 나감.

거리를 측정하기 위해 인위적으로 빛을 쏴야 하는데 이러한 방식을 active light source 방식

핵심 원리

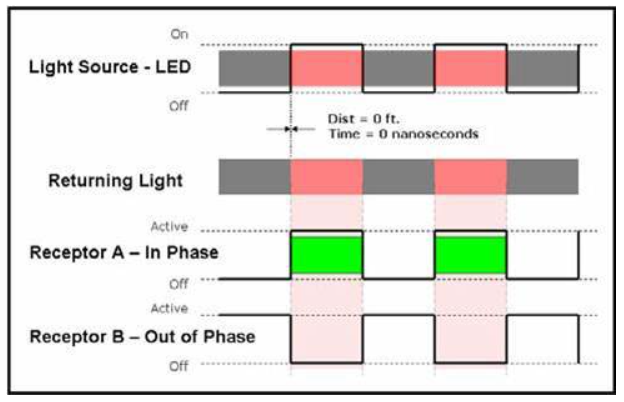
LED 발신부에서는 빛을 쏠 때 굉장히 빠른 간격으로 점멸을 시키면서 즉, 모듈레이션(modulation)을 시키면서 쏘고 수신부에서는 이 모듈레이션 간격과 동기화하여 receptor들을 활성화 시키는 것.



즉, 위 그림과 같이 LED를 켜는 동안에는 in phase receptor들만을 활성화시키고, LED를 끌 동안에는 out phase receptor들만을 황성화시킵니다.

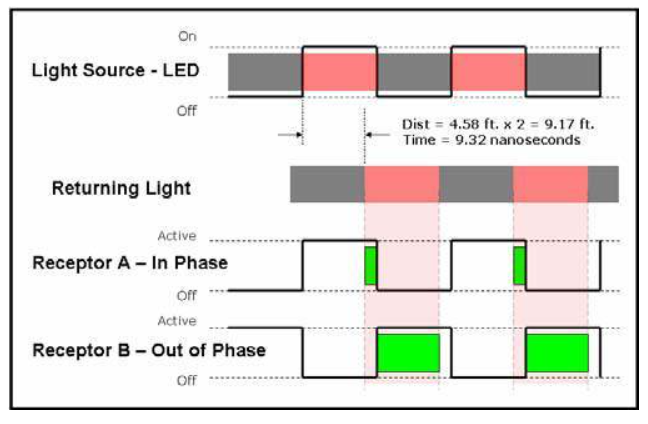
이와 같이 in phase receptor들과 out phase receptor들을 시간차를 두고 서로 다르게 활성화시키면 사물과의 거리에 따라서 수신되는(누적되는) 빛의 양에 차이가 발생하게 됩니다. 이 차이 즉, in phase receptor에 수신된 빛의 양과 out phase receptor에 수신된 빛의 양의 차이를 비교하여 사물과의 거리를 측정하는게 TOF 카메라의 기본 원리입니다.

먼저, 사물이 카메라 바로 앞에 있는 경우를 생각해 보겠습니다 (즉, 거리 = 0인 경우).



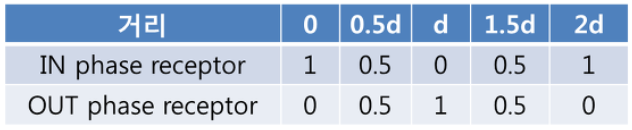
이 경우, 위 그림과 같이 빛이 갔다가 반사되어 오는 시간이 없기 때문에 LED의 점멸 주기가 그대로 빛의 수신(returning light) 주기가 됩니다. 따라서, 이 경우에는 in phase receptor 들만 빛을 수신하게 되고 out phase receptor들은 전혀 빛을 수신하지 못하게 됩니다.

다음으로 사물이 카메라로부터 어느 정도 떨어져 있어서 빛의 발신부와 수신부 사이에 시간차가 발생하는 경우를 살펴 보겠습니다.



이 경우, 위 그림과 같이 빛이 갔다가 되돌아오는 시간이 있기 때문에 수신부에 도달하는 빛은 LED 점별 주기와 시간차가 발생하게 됩니다. 그런데, in phase와 out phase를 활성화시키는 주기는 LED 점멸 주기와 일치하기 때문에 위 그림처럼 in phase receptor와 out phase receptor에 누적되는 빛의 양(녹색 부분)에 차이가 발생하게 됩니다.

즉, TOF 카메라의 기본 감지 거리(one light pulse distance의 2분의 1)를 d라 하고 receptor들에 수신되는 빛의 총합을 1이라 했을 때, 카메라와 사물과의 거리에 따라 receptor에 수신되는 빛의 양은 다음과 같이 변할 것입니다.



여기서, d는 한 펄스 동안 빛이 이동하는 거리의 1/2로서 d = 빛의 속도 x 모듈레이션 간격 / 2로 계산할 수 있습니다(빛은 사물까지 갔다가 되돌아오기 때문에 실제 사물까지의 거리는 빛의 이동거리/2임).

이상으로 TOF 카메라의 기본적인 거리 측정 원리에 대해 설명을 드렸습니다.

물론 실제 제품화될 때에는 이러한 기본 원리를 바탕으로 여러 기법들이 들어갑니다. 예를 들어, 거리 d를 전후로 수신 특성이 대칭되는 모호성(ambiguity) 문제를 해결하기 위해 receptor 활성화 주기를 LED 점멸 주기와 약간 어긋나게(shift) 하거나 수신 특성이 2d를 주기로 반복되는 문제를 해결하기 위해 한 영상 프레임을 획득하는 동안 서로 다른 두 종류의 점멸주기를 사용하는 등의 기법들이 사용된다고 합니다.