PR-002: Deformable Convolutional Networks (2017)

Data augmentation 할때 special transform을 Autometic하게

한계: 고정된 CNN필터(3x3, 5x5)가 너무 작위적이다.

이것을 학습시키자

정사각형이 아닌 조금 틀어진 형태로 배워보자.

1. 인트로

지금은 사람이 일일이 변형시켜준 후 Input하고 모델 capacity 러닝하게끔 만드는 것

두 가지 문제 존재

- Spacial한 transformation을 알고 있어야함
  항상 True는 아님, sensor signal이 있을 때 어떻게 augmentation 해야되는가
  arm을 detection할 때 size를 스케일링하면 안됨
  arm인지 아닌지 label을 바꿀 수 있는데
  어떤 domain specific한 사람의 knowledge를 집어넣어야 되는데 항상 가능한게 아니고 힘듦
  그러므로 사람이 주는게 아니라 알고리즘이 데이터를 학습해보자

아직도 D.A.의 transform invarinace를 많이 의존하고 있다.

우리가 fixed location에서 fixed filter를 쓴다

Fig.5.

Background를 배워야 할 때가 있고(왼쪽), 작은 걸 배워야 할 때가 있고(가운데), 큰걸 배워야 할 때가 있음(오른쪽)

큰거할땐 큰거쓰는게 맞겠죠 근데 그럼에도 고정하는걸 쓰고있으니 그러지말고 좀 유연하게 쓰자 하는게 논문의 주제임.

딥러닝을 한다는 것 = 사람의 개입 최소화

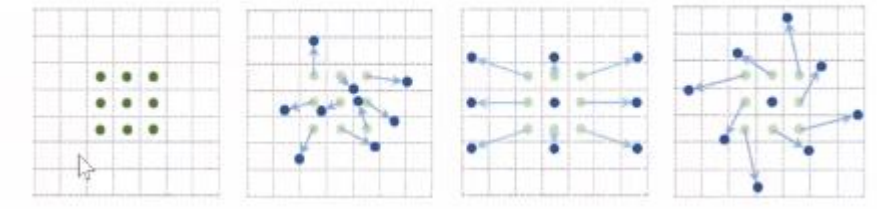
Handcraft를 줄이자, 사람이 개입하는 부분을 하나씩 없애자

그중에 없애야되는게 필터사이즈.

Q. VGG나 이런거 보면 작은 필터 레이어를 쌓으면 큰필터랑 같은 효과를 낼수잇다 라고 하는데 그런 개념으로 보면 큰필터는 필요없는거 아닌가요

A. linear, constatnt하게 growing을 하는데 쪼끄만건 성장속도도 달라야겠죠 그런걸 가지고있는거에요 깊게쌓으면 다 커버는 하긴 할텐데 그런것조차도 물체의 크기를 고려해야함

Fig.1.



Conv -> deformable Conv

만약 저 초록색 위치를 옮기면 어떨까

꼭 바로 주변이 아니라 움직일 수 있게끔 하면 좀더 자연스러운 넓은 영역의 것들을 배워서 가운데 학습에 쓸 수 있지 않을까

극단적인 케이스를 보면 만약에 정방으로 늘어나면 팽창하는것과 비슷하고 4번째는 rotation과 비슷한데 이런걸 다 커버한 일반화한게 deformable conv. 입니다

ROI -> deformable ROI 고정된 샘플의 위치를 변화시킬 수 있음

우리의 Method가 뭐와 관련되어 있냐면 spatial transform network와비슷하다는 것

어떤 그림이 주어졌을 때 가장 분류하기 좋은 형태(숫자를 가운데로 이동한다는 등)로 바꿔놓는다는 철학을 공유함.

2. Deformalbe ConvNet

2.1. Deformable Convolutioin

Offset에 대해서 summation하는 것.

여기선 delta가 꼭 1이 아니라 1.5 1.3 이 되어도 된다.

(1,1.5) 근데 이제 갖고있는 함수들은 grid마다 RGB값이 있음. 간단하게 interpolation해서 얻을 수 있다.

$$\mathcal{R} = \{(-1,-1),(-1,0),\ldots,(0,1),(1,1)\}$$

$$\mathbf{y}(\mathbf{p}_0) = \sum_{\mathbf{p}_n \in \mathcal{R}} \mathbf{w}(\mathbf{p}_n) \cdot \mathbf{x}(\mathbf{p}_0 + \mathbf{p}_n).$$

$$\mathbf{y}(\mathbf{p}_0) = \sum_{\mathbf{p}_n \in \mathcal{R}} \mathbf{w}(\mathbf{p}_n) \cdot \mathbf{x}(\mathbf{p}_0 + \mathbf{p}_n + \Delta\mathbf{p}_n).$$

Q. 그림이 이상해요

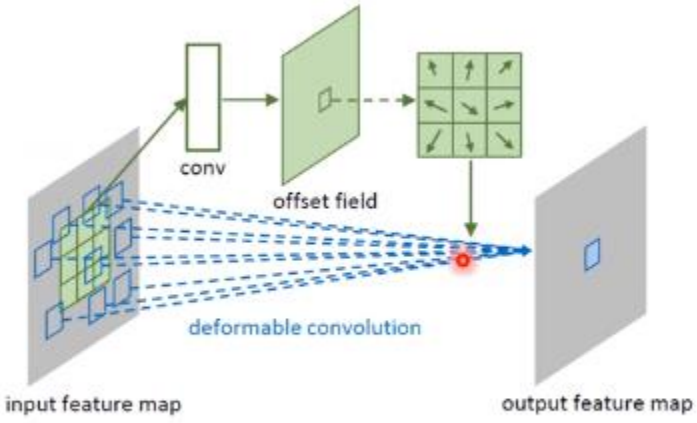


Figure 2: Illustration of 3×3 deformable convolution.

A. 실제 convolution이 이루어지는 부분은 저 빨간 부분임. 위에 있는 conv는 그냥 "떼어왔다" 라는 것. Offset Net이 있고 convolution Net이 있는데 그냥 하는게 아니라 offset parameter를 도입을 해서 conv가 일어난다.

2.2. Deformable RoI pooling

오른쪽에 봤듯이 max pooling 얘기인데 중심값이 있으면 중심값으로부터 주변값을 풀링하는거니까 offset을 도입하자.

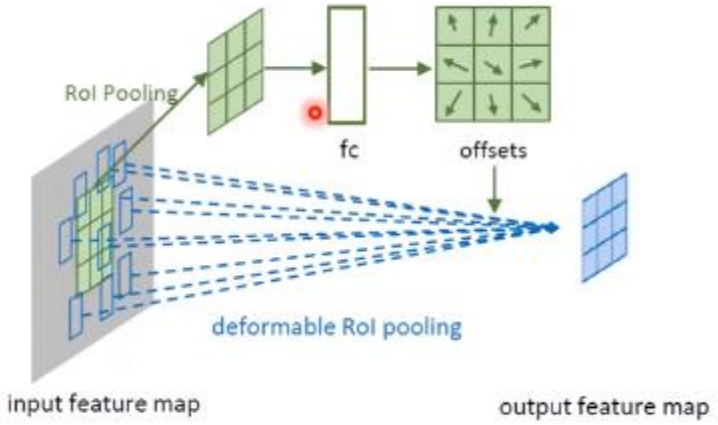


Figure 3: Illustration of 3×3 deformable RoI pooling.

$$\mathbf{y}(i,j) = \sum_{\mathbf{p} \in bin(i,j)} \mathbf{x}(\mathbf{p}_0 + \mathbf{p})/n_{ij},$$

$$\mathbf{y}(i,j) = \sum_{\mathbf{p} \in bin(i,j)} \mathbf{x}(\mathbf{p}_0 + \mathbf{p} + \Delta\mathbf{p}_{ij})/n_{ij}.$$

어떻게 학습할것이냐, 앞에선 CNN을 이용해서 하고 여기선 FC를 이용해서 할것임. 왜 달라지는지는 ...

2stage 3stage 나눠진게 아니라 label 이 있으면 back prop.해서 원래는 가중치만 학습했는데 back할 때 offset라인 네트워크도 같이 학습을 하자. 이는 STN이랑 똑같은 모양임 위쪽은 리니어. Input이 어떤 형태로 되냐면 결과적으로 classification이 잘 되는 형태로 input의 형태가 바뀔거고 그 형태가 아까 영상에서 보셨던(google) 가운데 딱 잘 박혀있는, 결국 이 파라미터들은 분류를 잘 하기 위한 것.

한계점

우리가 원하는 것은 사실은 다른 loss 따라서 배워지는 transformation들이 다르다는 것.

결과적으로 target task에 대해서는 아마도 더 적당한 input 형태가 만들어질 것.

Offset Net의 conv와 fc를 학습해서 델타p를 내놓는거고 아랫단에 w를 학습해서 거 두개가 합쳐진 상태에서 포워드..

2.3. Deformable ConvNets

장점

- 기존에 있던 Conv 레이어를 살짝만 바꿔도 작동을 함.
- 2~3 stage 그런 거 없다. 간단하다.

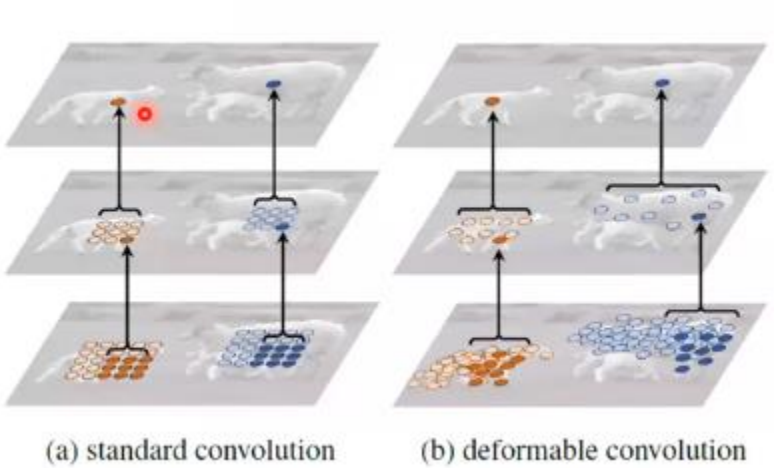
네트워크를 두단계로 나누면

- Conv feature 학습
- Task specific

얘도 여러가지 커널을 씌움.

나눈 이유는 O.D.할 때 pre-trained 된거 사용하니까 나눠서 첫번째는 모델 쓰시고 두번째는 specific하게 바꾸시고. 그 상태에서 back해서 파라미터 학습하시면 됩니다.

3. Understanding Deformable ConvNets



(a) standard convolution (b) deformable convolution

사이즈가 딱 저만할 필요는 없음. 그래서 b에서는 나비효과처럼 살짝 늘려도 레이어가 쌓이고 쌓이다 보면 엄청나게 커지며 훨씬 넓은 영역을 커버함.

Background를 배우는데는 전체를 다 볼 필요는 없다. Respective field가 똑 같은 패턴이 반복이 되기 때문에 large object보다 적은 부분만 보더라도 물체인식엔 문제가 없다.

멍멍이를 보면 다리 부분 박스가 늘어나 있는 것을 볼 수 있다.

3.1. Realted Works

STN

Linear transform을 배움. 로테이션하고 translation하는 두개를 배우니까 그게 한계점.

또다른거 하나는 bounding box를 잡으면 얘를 다시 정면에 딱 갖다 박아야되는데 그러려면 얘를 이제 interpolaiton해서 바꿔야된다는 것. 정면 사진으로. 그런데 이게 간단한 계산이긴 하지만 pixel이다보니 expensive함.

발상의 전환은 여태까진 filter weight을 신경썼는데 얘는 필터에(의) 어떤 샘플을 이용하느냐. 이 사람은 이제 w가 아니라 x에 신경을 쓴 것.

Receptive Field

이론상으로는 3x3이면 9x9로 커지고 이렇게 커져야되는데 그렇게 linear하게 커지지 않고 square root라는 것. 왜그런지 생각해보면 3x3이 가운데 있는게 stride를 꼭 그렇게 하지 않으니까 아무래도 겹쳐서 반영이 되는게 많으니 자연스레 가우시안처럼 중요하게 이용되고 반영되는게 많이

있음. 이 필드가 커지는 사이즈도 동등하게 9개가 중요하지 않기 때문에 growing size가 느리다는 것. 반면 D-ConvNet은 빠르다.

Atrous conv.

팽창 conv도 하는데 얘는 정방인데 D는 아님.

Dynamic Filter는 상황상황에 맞게끔 필터들을 여러 개 생성을 해놓음. 근데 우리는 필터가 아니라 x를 바꾸겠다는 것.

4. Experiment

짱.

Q&A

Weekly superivsed에서는 fixed 보다 D-를 쓰는게 더 좋을 것이다.

Q. 가운데 영향력 큰거는 D에서도 해결 안되지않음?

A. 네. 그래도 가랭이가 더 잘찢어지니까 더 잘 되는듯

Q. 연산량

A. STN 보단 적다. 왜냐면 바운딩박스를 찾았다고 해도 원하는 사이즈로 돌려놓긴 위해선 옮기는데 그게 다 미분해야 백이 되니까. 근데 D는 단순히 3x3 파라미터 서브넷 하나만더 배워준거기 때문.

마지막 last 3 layer만 했을때가 가장 효과적이라고 함. 만약에 이거를 data augmentation 하려고 360도 돌리는거 생각하면 이게 더 효과적일 듯.

Q. RGB 채널별로 컬러를 바꿔주는 것도 생각

A. 어차피 last layer에서 하니까 다 섞이니까 괜찮을 듯.