

DeepLAB : Semantic Image Segmentation with Deep Convolutional Nets and Fully Connected CRFs

DeepLab V3+ : Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation (ECCV, 2018)

Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam

[DeepLab V3+ : Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation \(ECCV, 2018\)](#)

Abstract

1. Introduction

[DeepLab V3 + 저자들의 Contribution](#)

2. Related Work

[Atrous Convolution](#)

[Spatial pyramid pooling](#)

[Encoder-decoder](#)

3. Methods

3.1. Encoder-Decoder with Atrous Convolution

[DeepLab V3+ Architecture](#)

3.2. Modified Aligned Xception

4. Experimental Evaluation

4.1. Decoder Design Choices

[4.2. ResNet-101 as Network Backbone](#)

[4.3. Xception as Network Backbone](#)

[4.4. Improvement along Object Boundaries](#)

[4.5. Experimental Results on Cityscapes](#)

5. Conclusion

6. Reference

Abstract

DeepLab은 version 1부터 시작하여 지금까지 총 4번의 개정본(1, 2, 3, 3+)이 출판됨

- **DeepLab V1** : Semantic Image Segmentation with Deep Convolutional Nets and Fully Connected CRFs. (ICLR, 2015)
- **DeepLab V2** : Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs. (IEEE TPAMI, 2017)
- **DeepLab V3** : Rethinking Atrous Convolution for Semantic Image Segmentation. (arXiv, 2017)
- **DeepLab V3+** : Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation. (ECCV, 2018)

V1에서는 **DCNN**(Deep Convolutional Neural Network)에 **Fully Connected CRF** 적용

V2에서는 **Atrous Convolution** 개념 등장

V3에서는 **ASPP(Atrous Spatial Pyramid Pooling)** 기법을 제안

V3+에서는 separable convolution과 ASPP를 결합한 **ASSPP(Atrous Separable Spatial Pyramid Pooling)**을 제안

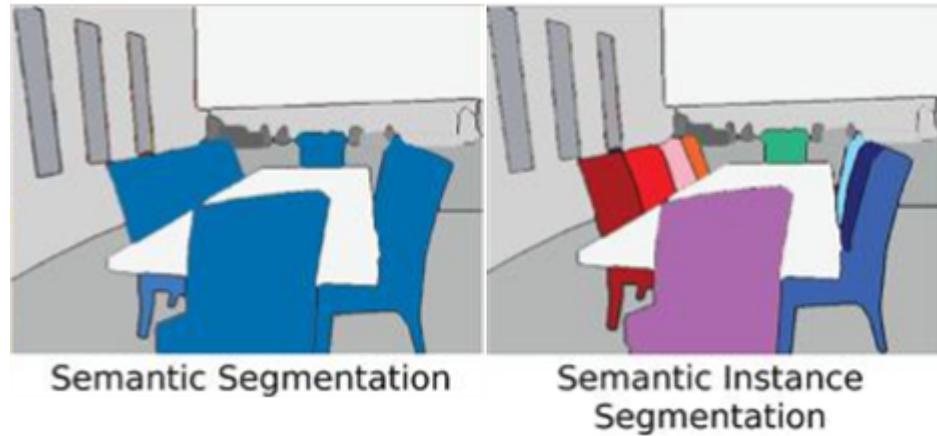
	mean	aero plane	bicycle	bird	boat	bottle	bus	car	cat	chair	cow	dining table	dog	horse	motor bike	person	potted plant	sheep	sofa	train	tv/ monitor
▶ DeepLabv3+_JFT [?]	89.0	97.5	77.9	96.2	80.4	90.8	98.3	95.5	97.6	58.8	96.1	79.2	95.0	97.3	94.1	93.8	78.5	95.5	74.4	93.8	81.6
▷ SRC-B-MachineLearningLab [?]	88.5	97.2	78.6	97.1	80.6	89.7	97.4	93.7	96.7	59.1	95.4	81.1	93.2	97.5	94.2	92.9	73.5	93.3	74.2	91.0	85.0
▷ DeepLabv3+_AASPP [?]	88.5	97.4	80.3	97.1	80.1	89.3	97.4	94.1	96.9	61.9	95.1	77.2	94.2	97.5	94.4	93.0	72.4	93.8	72.6	93.3	83.3
▷ ExFuse [?]	87.9	96.8	80.3	97.0	82.5	87.8	96.3	92.6	96.4	53.3	94.3	78.4	94.1	94.9	91.6	92.3	81.7	94.8	70.3	90.1	83.8
▷ DeepLabv3+ [?]	87.8	97.0	77.1	97.1	79.3	89.3	97.4	93.2	96.6	56.9	95.0	79.2	93.1	97.0	94.0	92.8	71.3	92.9	72.4	91.0	84.9
▷ DeepLabv3-JFT [?]	86.9	96.9	73.2	95.5	78.4	86.5	96.8	90.3	97.1	51.4	95.0	73.4	94.0	96.8	94.0	92.3	81.5	95.4	67.2	90.8	81.8
▷ DIS [?]	86.8	94.0	73.3	93.5	79.1	84.8	95.4	89.5	93.4	53.6	94.8	79.0	93.6	95.2	91.5	89.6	78.1	93.0	79.4	94.3	81.3
▷ CASIA_IVA_SDN [?]	86.6	96.9	78.6	96.0	79.6	84.1	97.1	91.9	96.6	48.5	94.3	78.9	93.6	95.5	92.1	91.1	75.0	93.8	64.8	89.0	84.6
▷ IDW-CNN [?]	86.3	94.8	67.3	93.4	74.8	84.6	95.3	89.6	93.6	54.1	94.9	79.0	93.3	95.5	91.7	89.2	77.5	93.7	79.2	94.0	80.8
▷ DFN [?]	86.2	96.4	78.6	95.5	79.1	86.4	97.1	91.4	95.0	47.7	92.9	77.2	91.0	96.7	92.2	91.7	76.5	93.1	64.4	88.3	81.2
▷ EncNet [?]	85.9	95.3	76.9	94.2	80.2	85.3	96.5	90.8	96.3	47.9	93.9	80.0	92.4	96.6	90.5	91.5	70.9	93.6	66.5	87.7	80.8
▷ HPN [?]	85.8	94.1	67.0	95.2	81.9	88.3	95.5	90.4	95.9	40.0	92.7	82.5	91.7	95.3	92.6	91.6	73.6	94.1	69.4	91.1	81.9
▷ DeepLabv3 [?]	85.7	96.4	76.6	92.7	77.8	87.6	96.7	90.2	95.4	47.5	93.4	76.3	91.4	97.2	91.0	92.1	71.3	90.9	68.9	90.8	79.3
▷ PSPNet [?]	85.4	95.8	72.7	95.0	78.9	84.4	94.7	92.0	95.7	43.1	91.0	80.3	91.3	96.3	92.3	90.1	71.5	94.4	66.9	88.8	82.0
▷ ** ResNet-38_COCO ** [?]	84.9	96.2	75.2	95.4	74.4	81.7	93.7	89.9	92.5	48.2	92.0	79.9	90.1	95.5	91.8	91.2	73.0	90.5	65.4	88.7	80.6

Pascal VOC 2012 leaderboard 결과

▼ Instance Segmentation vs. Semantic Segmentation

Image segmentation은 이미지의 영역을 분할해서 각 object에 맞게 합쳐주는 task

Image segmentation에는 **Semantic Segmentation**과 **Instance Segmentation**이 있음



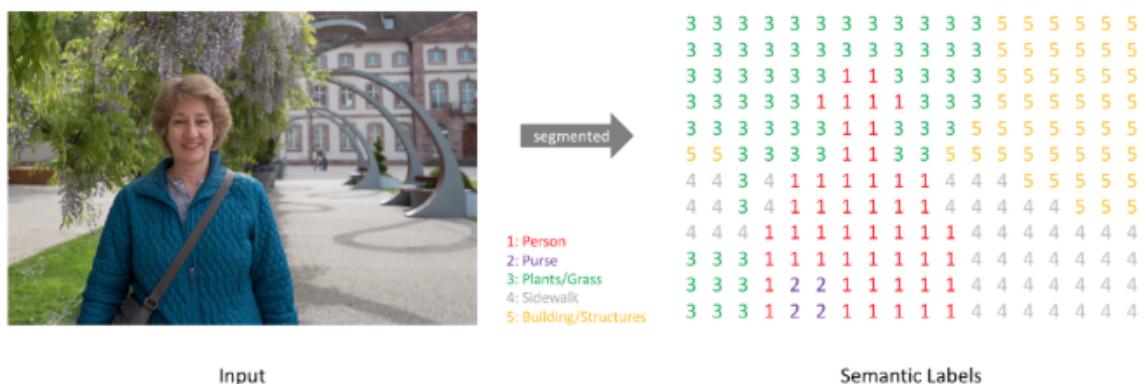
Semantic segmentation이란 Object segmentation을 하되,

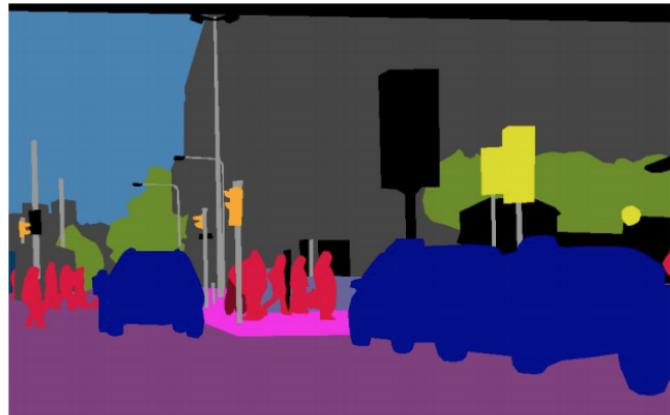
같은 class인 object들은 같은 영역/색으로 분할

반대로 Instance segmentation은 같은 class이여도 서로 다른 instance로 구분

→ object가 겹쳤을때 각각의 object를 구분해주지 못하는 문제를

Instance segmentation을 통해 해결할 수 있음





(b) semantic segmentation

Semantic segmentation의 목적은 각 픽셀별로 어떤 class에 속하는지 label을 구해주는 것

- pixel들이 각 class에 대해 binary하게 포함되는지 안되는지 여부만 따짐
- 같은 class의 object 들에 대해 서로 구분 지을 수 없다는 단점



(c) instance segmentation

Instance segmentation은 각 픽셀 별로 어떤 카테고리에 속하는지 계산하는 것이 아닌 각 픽셀 별로 object가 있는지 없는지 여부만 계산

일반적으로 Mask R-CNN과 같은 2-stage detector에서는 먼저 object들을 localization시킨 후 localize된 ROI(Region of Interest)마다 class의 개수만큼 binary mask(instance인지 아닌지) 마스크를 씌움

- class 개수만큼 output 채널이 존재하지 않고 ROI별로 output 채널이 존재
- 동일 class라도 서로 다른 instance, 즉 ROI가 focus하는 instance부분만 value를 갖음

1. Introduction

DeepLab V3 + 저자들의 Contribution

- DeepLab v3라는 강력한 encoder와 간단하지만 효과적인 **decoder**를 사용한 새로운 encoder-decoder 구조를 제안
- Atrous convolution을 통해 추출된 encoder features의 해상도를 임의로 조절할 수 있음
- **Xception model**을 채택하고, depthwise separable convolution을 atrous spatial pyramid pooling(ASSPP module)과 decoder에 적용
 - 이는 빠르고 강력한 encoder-decoder network를 만듦
- 제안된 모델은 PASCAL VOC 2012와 Cityscapes dataset들에서 sota의 성능을 보여 줌

2. Related Work

▼ FCN(Fully Convolutional Networks)란?

기존 CNN 모델의 Fully-Connected-Layer 층을 Convolutional Layer로 대체하는 것

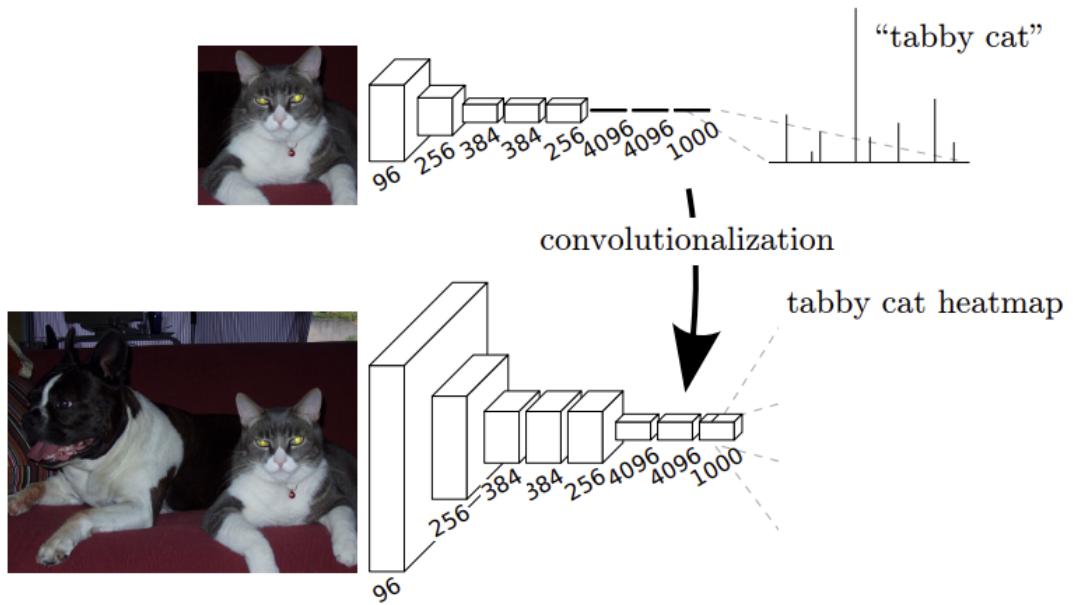
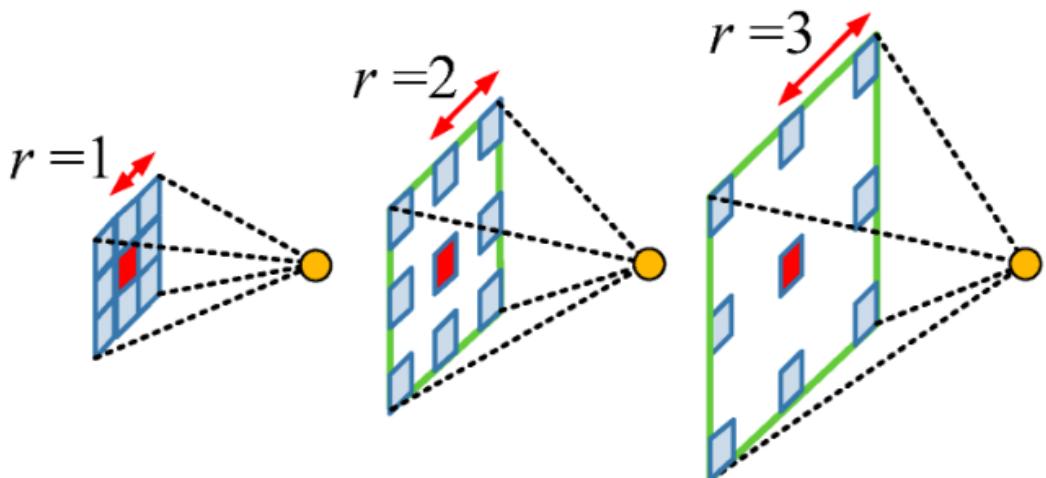


Figure 2. Transforming fully connected layers into convolution layers enables a classification net to output a heatmap. Adding layers and a spatial loss (as in Figure 1) produces an efficient machine for end-to-end dense learning.

<https://medium.com/@msmapark2/fcn-논문-리뷰-fully-convolutional-networks-for-semantic-segmentation-81f016d76204>

Atrous Convolution



Atrous convolution은 기존 convolution과 다르게 **필터 내부에 빈 공간**을 둔 채로 작동

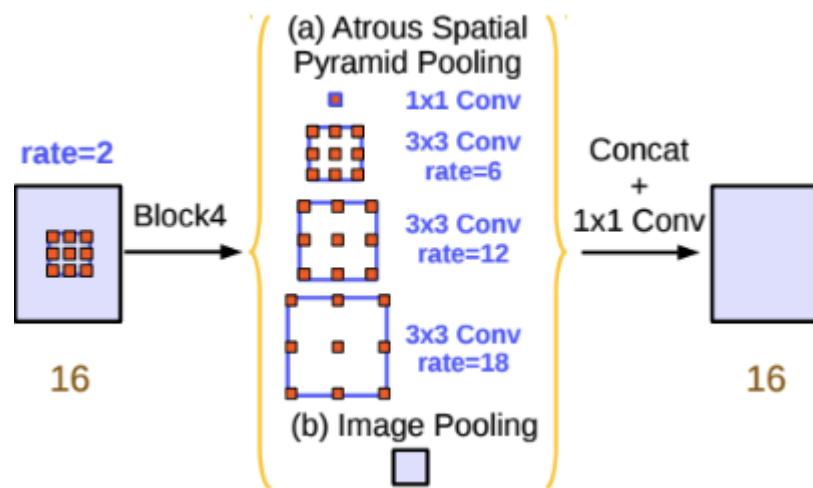
위 예시에서, 얼마나 빈 공간을 둘 지 결정하는 파라미터인 rate $r=1$ 일 경우 기존 convolution과 동일(r 이 커질 수록, 빈 공간이 넓어짐)

Atrous convolution을 활용함으로써 기존 convolution과 동일한 양의 **파라미터와 계산량을 유지**하면서도, **field of view(한 픽셀이 볼 수 있는 영역)**를 크게 가져갈 수 있게 됨

보통 semantic segmentation에서 높은 성능을 내기 위해서는 convolutional neural network의 마지막에 존재하는 한 픽셀이 입력 값에서 어느 크기의 영역을 커버할 수 있는지를 결정하는 **receptive field** 크기가 중요하게 작용

→ Atrous convolution을 활용하면 **파라미터 수를 늘리지 않으면서도 receptive field를 크게 키울 수 있기 때문에 DeepLab series에서는 이를 적극적으로 활용**

Spatial pyramid pooling



Semantic segmentation의 성능을 높이기 위한 방법 중 하나로 자주 활용되고 있는 기법

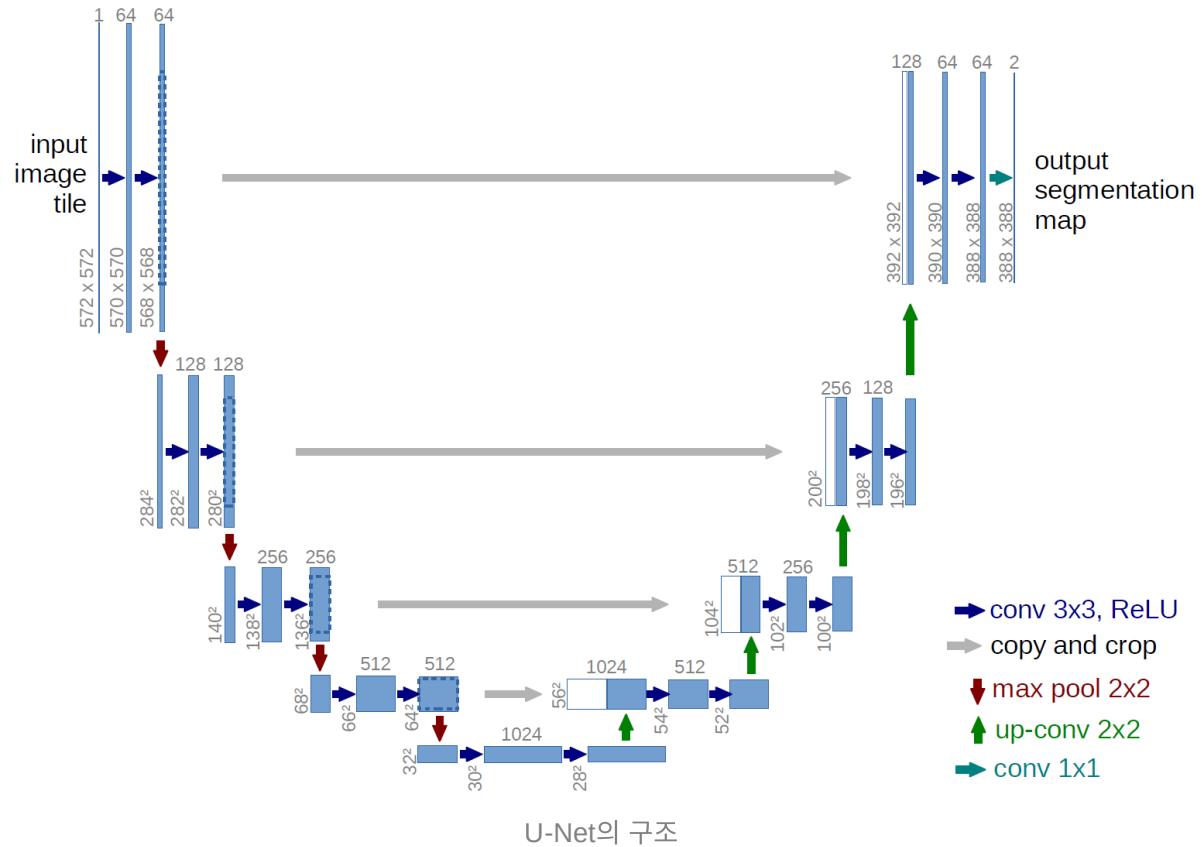
DeepLab V2에서는 feature map으로부터 여러 개의 **rate가 다른 atrous convolution을 병렬로 적용한 뒤, 이를 다시 합쳐주는 atrous spatial pyramid pooling(ASPP) 기법**을 활용

PSPNet(Pyramid Scene Parsing Network)에서도 atrous convolution을 활용하진 않지만 이와 유사한 pyramid pooling 기법 적극 활용

→ **multi-scale context**를 모델 구조로 구현하여 보다 정확한 semantic segmentation을 수행

DeepLab 시리즈는 **ASPP를 기본 모듈로 사용**하고 있음

Encoder-decoder



Encoder-decoder 구조 또한 semantic segmentation을 위한 CNN 구조로 자주 활용

특히 **U-Net**이라고 불리는 encoder-decoder 구조는 정교한 픽셀 단위의 **segmentation**이 요구되는 biomedical image segmentation task의 핵심 요소



Encoder-Decoder 구조

encoder 부분에서는 점진적으로 spatial dimension을 줄여가면서 **고차원의 semantic 정보**를 convolution filter가 추출

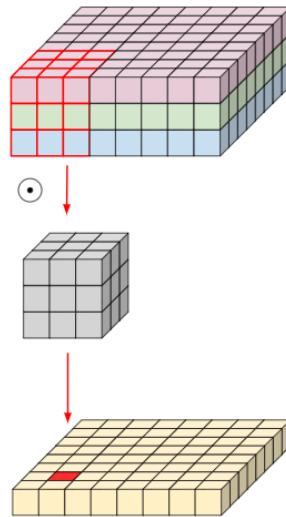
decoder 부분에서는 encoder에서 spatial dimension 축소로 인해 손실된 **spatial 정보**를 점진적으로 복원하여 보다 정교한 **boundary segmentation**을 완성

U-Net이 여타 encoder-decoder 구조와 다른 점은, 위 그림에서 가운데 놓인 회색 선

Spatial 정보를 복원하는 과정에서 이전 encoder feature map 중 **동일한 크기를 지닌 feature map을 가져와 prior로 활용함**으로써 더 정확한 boundary segmentation이 가능하게 만듦

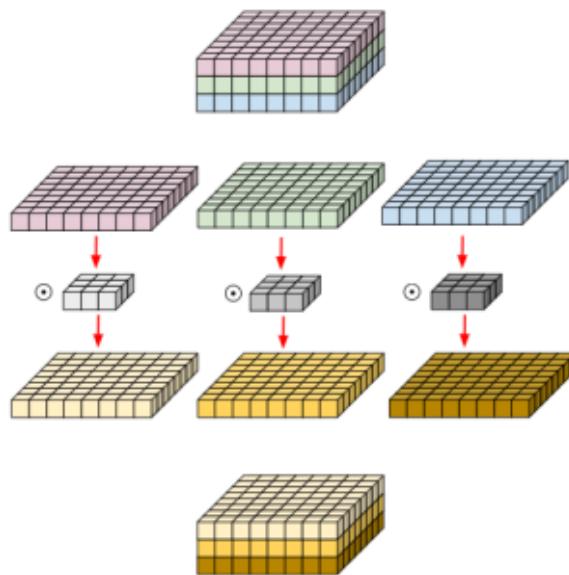
DeepLab V3+에서는 U-Net과 유사하게 intermediate connection을 가지는 encoder-decoder 구조를 적용하여 **보다 정교한 object boundary를 예측**

▼ Depthwise Separable Convolution



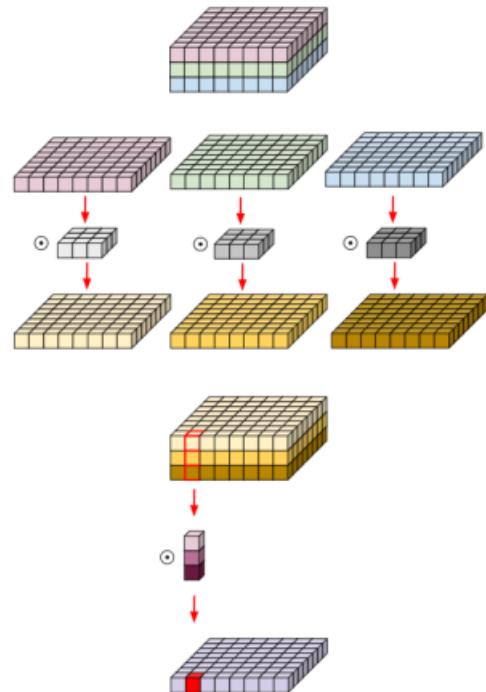
입력 이미지가 $8 \times 8 \times 3$ ($H \times W \times C$)이고, convolution filter 크기가 3×3 ($F \times F$)인 경우
filter 한 개가 가지는 파라미터 수는 $3 \times 3 \times 3$ ($F \times F \times C$) = 27

만약 filter가 4개 존재한다면, 해당 convolution의 총 파라미터 수는 $3 \times 3 \times 3 \times 4$ ($F \times F \times C \times N$)



Convolution 연산에서 **입력 영상의 channel 축을 모두 분리**시킴

channel을 항상 1로 가지는 여러 개의 convolution 필터로 대체시킨 연산을
depthwise convolution



depthwise convolution으로 나온 결과에 **1×1×C 크기의 convolution filter를 적용**한 것을 depthwise separable convolution

기존 convolution과 유사한 성능을 보이면서 사용되는 **파라미터 수와 연산량을 획기적으로 줄임**

ex. 입력값이 8×8×3이고 16개의 3×3 convolution 필터를 적용할 때 사용되는 파라미터의 수는

- Convolution: $3 \times 3 \times 3 \times 16 = 432$
- Depthwise separable convolution: $3 \times 3 \times 3 + 3 \times 16 = 27 + 48 = 75$

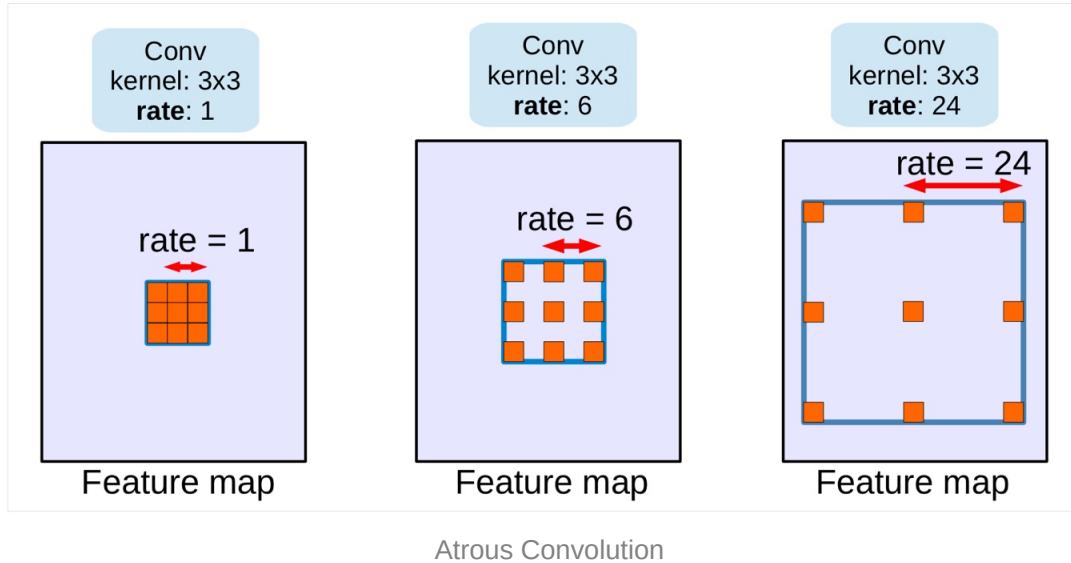
픽셀 각각에 대해서 label을 예측해야 하는 **semantic segmentation**은 CNN 구조가 깊어짐

→ receptive field를 넓히기 위해 **더 많은 파라미터들을 사용**

1. separable convolution을 활용할 경우 보다 **깊은 구조로 확장하여 성능 향상**
2. 기존 대비 **메모리 사용량 감소와 속도 향상**을 기대할 수 있음

3. Methods

3.1. Encoder-Decoder with Atrous Convolution



Atrous convolution(dilated convolution)은 DCNN에 계산되는 features의 해상도를 명시적으로 조절

다양한 크기의 정보를 포착하기 위해 convolution filter의 FOV(field-of-view)를 조절

※ Trous는 프랑스말로 구멍을 의미

$$y[i] = \sum_k x[i + r \cdot k]w[k]$$

i : each location in feature map

y : output feature map

x : input feature map

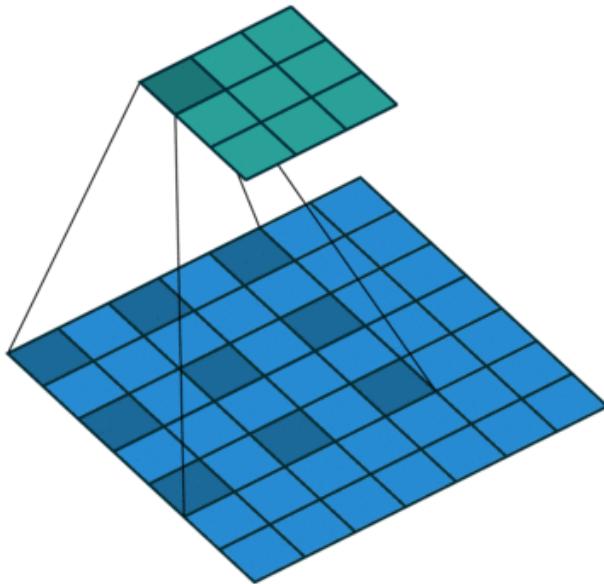
w : convolution filter

r : dilate rate

k : kernel size

input과 filter를 곱할 때, r 의 값에 따라 얼마나 input을 띄엄띄엄 선택하여 filter와 곱할지 결정됨

▼ Atrous Convolution 이란?



기존 컨볼루션 필터가 수용하는 픽셀 사이에 간격을 둔 형태

입력 픽셀 수는 동일하지만, 더 넓은 범위에 대한 입력을 수용할 수 있게 됨

dilation rate가 2인 3×3 커널은 9개의 파라미터를 사용하면서 5×5 커널과 동일한 view를 가짐

Dilated convolution 은 특히 real-time segmentation 분야에서 주로 사용됨

넓은 view 가 필요하고, 여러 컨볼루션이나 큰 커널을 사용할 여유가 없을 때 사용함

→ 즉, 적은 계산 비용으로 Receptive Field 를 늘리는 방법

이 Dilated convolution 은 필터 내부에 zero padding 을 추가해서 강제로 Receptive Field 를 늘리게 되는데, 위 그림에서 진한 파란색 부분만 weight 가 있고 나머지 부분은 0으로 채워지게 됨

이 Receptive Field는 필터가 한번 보는 영역으로 사진의 Feature를 파악하고, 추출하기 위해서는 넓은 Receptive Field 를 사용하는 것이 좋음

dimension 손실이 적고, 대부분의 weight 가 0 이기 때문에 연산의 효율이 좋음

(공간적 특징을 유지하는 Segmentation에서 주로 사용되는 이유)

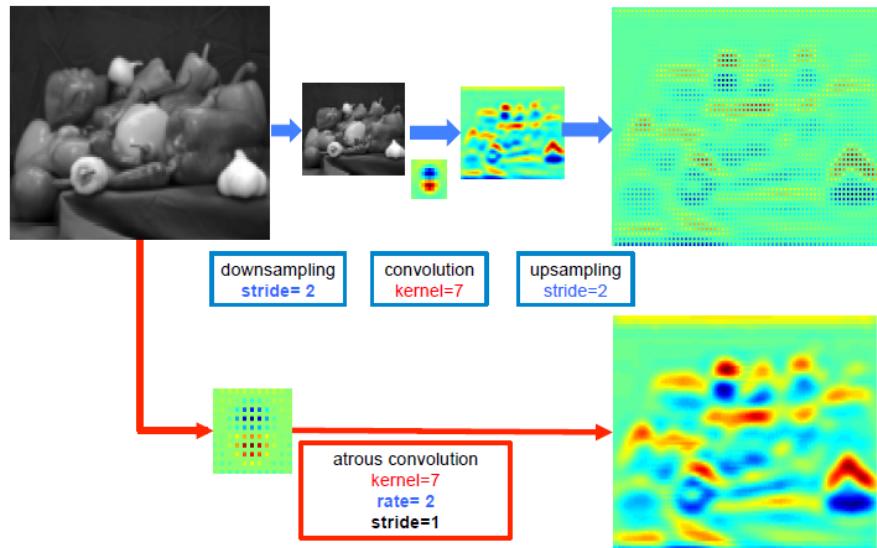


Fig. 3: Illustration of atrous convolution in 2-D. Top row: sparse feature extraction with standard convolution on a low resolution input feature map. Bottom row: Dense feature extraction with atrous convolution with rate $r = 2$, applied on a high resolution input feature map.

Segmentation 문제에서 Dilated Convolution을 통해 이득을 보는 경우를 볼 수 있음

단순히 Pooling - Convolution 후 Upsampling 하는 것과 Dilated Convolution(Atrous convolution)을 하는 것의 차이를 볼 수 있음

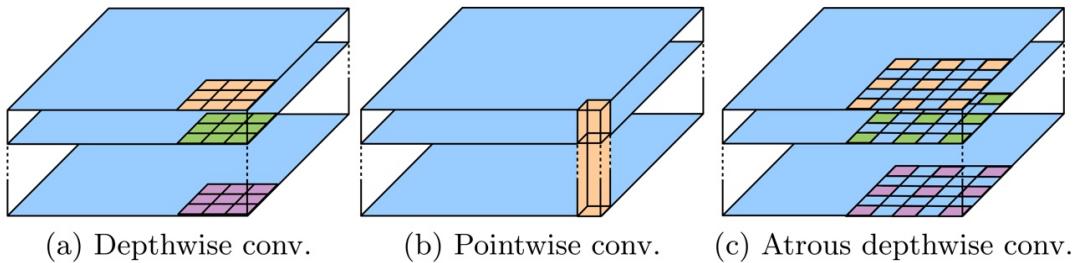
전자의 경우에는 공간적 정보의 손실이 있는 것을 upsampling 하면 해상도가 떨어지게 됨

하지만 dilated convolution 의 그림을 보면, Receptive field 를 크게 가져가면서 convolution을 하면 정보의 손실을 최대화하면서 해상도는 큰 output 을 얻을 수 있음

Atrous convolution은 pooling을 수행하지 않고도 receptive field를 크게 가져갈 수 있음

→ spatial 정보의 손실이 적음

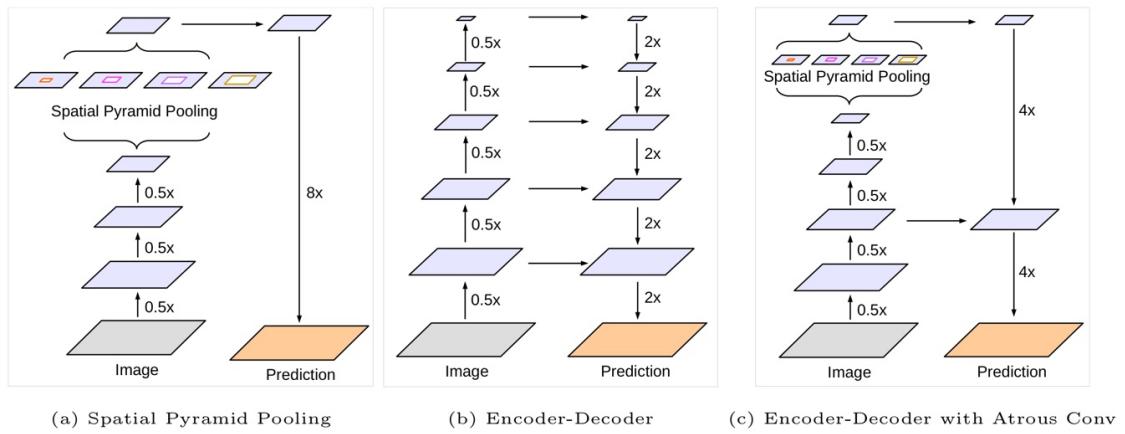
연산에 사용되지 않은 부분(hole)은 weight가 0이기 때문에 연산의 효율이 좋음



atrous depthwise convolution(c)과 pointwise convolution(b)를 결합시킨 **atrous separable convolution**을 사용

→ 제안된 모델의 성능을 유지하면서(혹은 좋게 하면서) 연산 복잡도를 감소시킴

DeepLab V3+ Architecture



(a): DeepLab V3, (b): U-Net, (c): DeepLab v3+

DeepLab V3 → DeepLab V3+

Encoder : ResNet with atrous convolution → Xception (Inception with separable convolution)

ASPP : ASPP → ASSPP (Atrous Separable Spatial Pyramid Pooling)

Decoder : Bilinear upsampling → Simplified U-Net style decoder

1. ResNet을 사용하던 encoder가 **atrous separable convolution**을 적극 활용한 구조인 **Xception**으로 대체됨
2. Multi-scale context를 얻기 위해 활용되던 ASPP에는 **separable convolution**과 **atrous convolution**을 결합한 atrous separable convolution이 적용된 ASSPP로 대체됨
3. 기존에 단순하게 bilinear upsampling으로 해결했던 decoder 부분이 **U-Net과 유사한 형태의 decoder**로 대체됨
 - Encoder와 ASPP, 그리고 decoder 모두 separable convolution을 적극 활용함으로써 파라미터 사용량 대비 성능 효율을 극대화

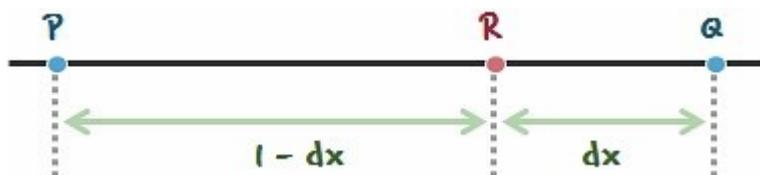
▼ Bilinear Upsampling이란?

DownSampling(Encoder) : stride를 2 이상 갖는 convolution이나 pooling을 사용하여 dimension을 줄여 적은 메모리로 convolution 수행

Upsampling(Decoder) : Downsampling을 통해서 받은 결과의 dimension을 늘려서 입력과 같은 dimension을 만듦

→ 즉, Encoder를 통해서 입력 받은 이미지의 정보를 압축된 벡터에 표현하고 Decoder를 통해서 원하는 결과물의 크기로 만들어냄

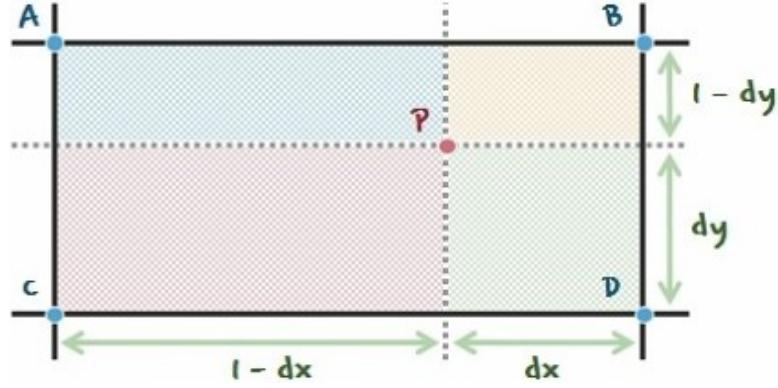
Linear interpolation(선형 보간법)



값을 아는 두 점 P, Q 사이의 모르는 값 R을 유추할 때, 세 점 P, Q, R의 값들이 선형적인 관계를 갖는다고 가정

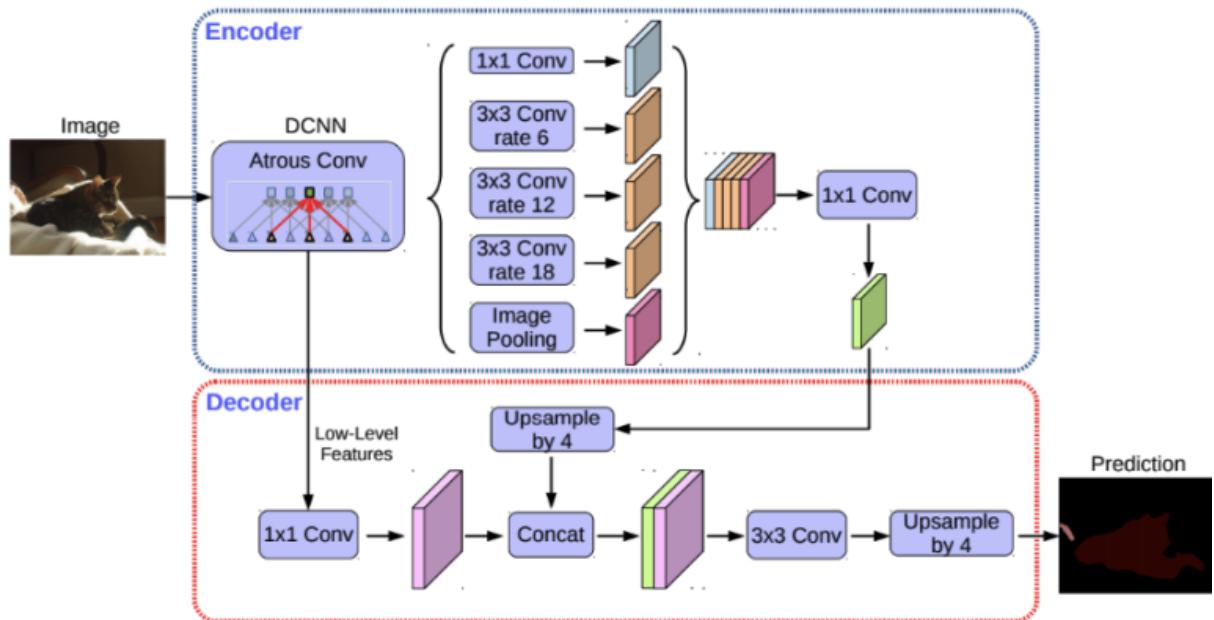
두 점 P, Q의 값과 거리 비를 이용하여 R의 값을 구함(P와 Q 사이의 거리를 모두 1로 가정)

Bilinear interpolation(양선형 보간법)



네 꼭지점에서의 값이 주어져 있을 때 내부의 임의의 점에서의 값 추정
 linear interpolation을 x축과 y축으로 두 번 적용하여 값을 유추하는 방법
 네 개의 인접한 점들의 값과 그에 따른 면적을 가중치로 하여 값을 구함

→ input 크기만큼 resolution을 복구하는 과정에서 Bilinear Interpolation 사용



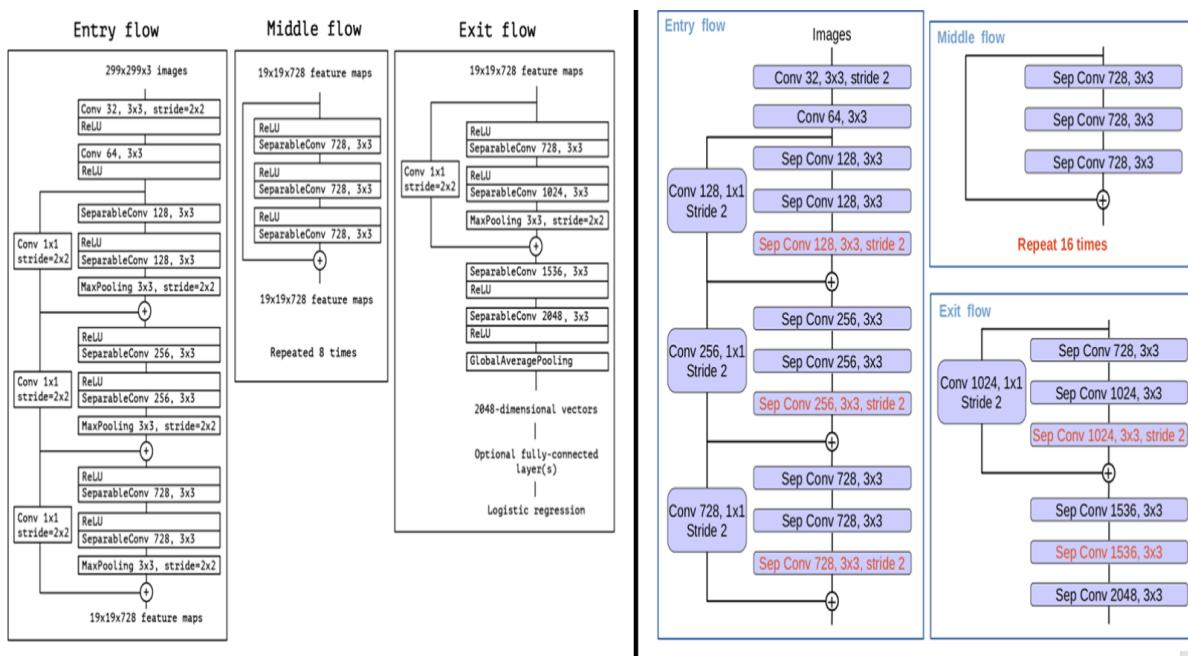
DeepLab v3에서는 encoder feature들을 단순히 8배 bilinear upsample를 수행
 이러한 naive decoder는 **object segmentation details**를 복원하지 못 함



DeepLab V3+ 순서도

1. DeepLab v3+에서는 우선 Encoder에서 나온 encoder feature를 **4배 bilinear upsampling**
 2. 그 다음 backbone network 중간에서 나온 feature maps(Low-Level Features)을 1×1 convolution을 적용하여 **channel을 줄인 뒤 이 둘을 concatenate**
 3. 그 다음 3×3 convolution layers를 거친 후 output을 원래 input size로 **4배 bilinear upsampling**

3.2. Modified Aligned Xception



왼쪽: Xception, 오른쪽: modified aligned Xception

Xception 모델은 classification에서 빠른 연산과 함께 좋은 성능을 보임

다음과 같이 수정을 거친 Xception 모델을 backbone network(modified aligned Xception)로 사용

- Entry flow에서는 빠른 연산과 메모리 효율을 위해 수정을 하지 않음

- 모든 max pooling 연산은 **depthwise separable convolution** with striding으로 대체
- 각각의 3x3 depthwise convolution 뒤에 **batch normalization**과 **ReLU** 활성화 함수를 추가

4. Experimental Evaluation

ImageNet-1k로 pre-train된 **ResNet-101** 또는 **modified aligned Xception**을 **backbone**으로 사용

모델 평가 데이터로 PASCAL VOC 2012 semantic segmentation benchmark를 사용

Dataset은 1464개의 train data, 1449개의 validation data, 1456개의 test data로 구성되어 있고,

train data를 data augmentation을 통해 10582개의 train data를 사용

평가지표로는 pixel intersection-over-union averaged across the 21 classes(**mIOU**)를 사용

학습방법은 DeepLab v3와 같은 방법으로 진행

- Learning rate policy: 초기 learning rate=0.007이고, poly learning rate policy를 사용
- Crop size: Atrous convolution이 큰 dilate rate에서 효과적이기 위해 큰 crop size를 사용해야 하므로 513x513 crop image를 사용
- Batch normalization: batch size = 16

4.1. Decoder Design Choices

Decoder의 구조 결정을 위해 ResNet-101을 backbone으로 사용하는 Encoder를 사용해 서 실험

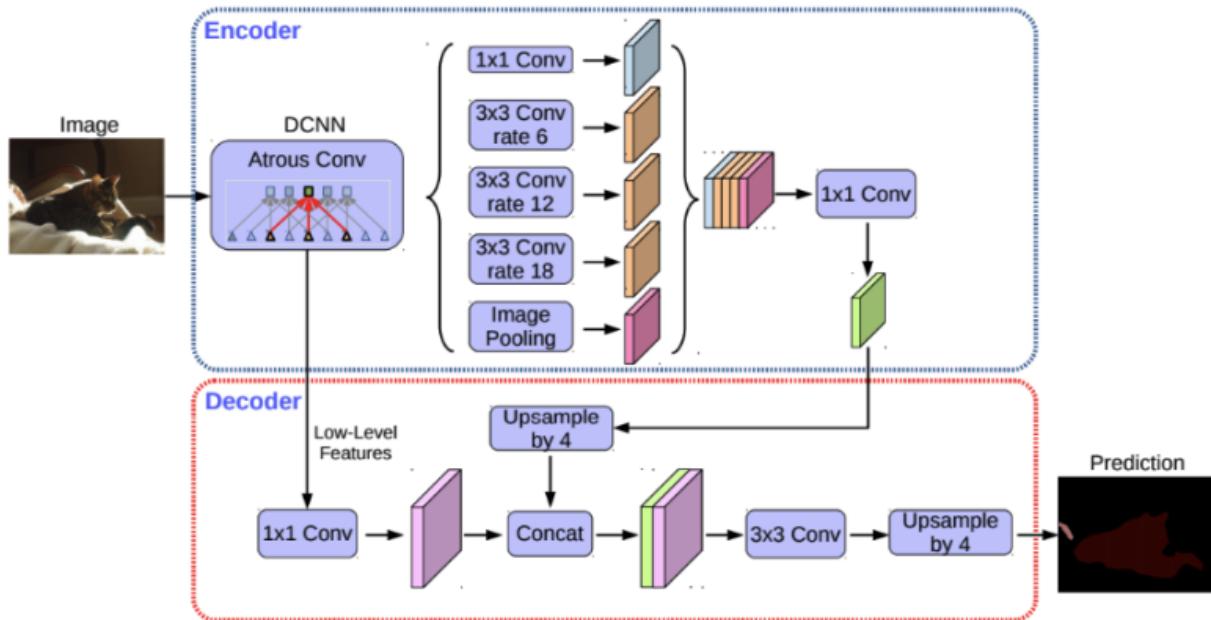
Channels	8	16	32	48	64
mIOU	77.61%	77.92%	78.16%	78.21%	77.94%

Table 1. PASCAL VOC 2012 *val* set. Effect of decoder 1×1 convolution used to reduce the channels of low-level feature map from the encoder module. We fix the other components in the decoder structure as using $[3 \times 3, 256]$ and Conv2.

Features Conv2 Conv3	3×3 Conv Structure	mIOU
✓	$[3 \times 3, 256]$	78.21%
✓	$[3 \times 3, 256] \times 2$	78.85%
✓	$[3 \times 3, 256] \times 3$	78.02%
✓	$[3 \times 3, 128]$	77.25%
✓	$[1 \times 1, 256]$	78.07%
✓ ✓	$[3 \times 3, 256]$	78.61%

Table 2. Effect of decoder structure when fixing $[1 \times 1, 48]$ to reduce the encoder feature channels. We found that it is most effective to use the Conv2 (before striding) feature map and two extra $[3 \times 3, 256]$ operations. Performance on VOC 2012 *val* set.

- low-level feature map에 적용하는 1×1 convolution을 통해 channel을 얼마나 줄여야 하는지
→ **48 channel**
- Decoder 마지막에 사용되는 3×3 convolution을 구조를 어떻게 해야 할지
→ **$[3 \times 3, 256]$ convolution을 2번 적용**
- Encoder module의 어느 부분에서 나오는 low-level feature map을 사용해야 할지
→ backbone network의 conv3 feature map을 같이 사용했을 때 성능 향상이 별로 없으므로 backbone network의 **conv2 feature map**만 low-level feature로 사용



4.2. ResNet-101 as Network Backbone

Encoder train OS	Decoder eval OS	MS	Flip	mIOU	Multiply-Adds
16	16			77.21%	81.02B
16	8			78.51%	276.18B
16	8		✓	79.45%	2435.37B
16	8		✓ ✓	79.77%	4870.59B
16	16	✓		78.85%	101.28B
16	16	✓	✓	80.09%	898.69B
16	16	✓	✓ ✓	80.22%	1797.23B
16	8	✓		79.35%	297.92B
16	8	✓	✓	80.43%	2623.61B
16	8	✓	✓ ✓	80.57%	5247.07B
32	32			75.43%	52.43B
32	32	✓		77.37%	74.20B
32	16	✓		77.80%	101.28B
32	8	✓		77.92%	297.92B

Table 3. Inference strategy on the PASCAL VOC 2012 *val* set using *ResNet-101*. **train OS:** The *output stride* used during training. **eval OS:** The *output stride* used during evaluation. **Decoder:** Employing the proposed decoder structure. **MS:** Multi-scale inputs during evaluation. **Flip:** Adding left-right flipped inputs.

- 평가 시 output stride=8 즉 **denser feature map**을 사용하는 것과 **multi-scale inputs**는 성능을 향상시킴

- **Flip**은 연산 복잡도를 2배 늘리지만 아주 약간의 성능 향상
- **Decoder**를 사용할 때 적은 연산량의 증가로 높은 성능 향상
- Training시 output stride가 32인 경우, 즉 atrous convolution을 사용하지 않을 경우 computation은 줄어들지만 성능 또한 함께 떨어짐

4.3. Xception as Network Backbone

Encoder train OS	Decoder eval OS	MS	Flip	SC	COCO	JFT	mIOU	Multiply-Adds
16	16						79.17%	68.00B
16	16		✓				80.57%	601.74B
16	16		✓	✓			80.79%	1203.34B
16	8						79.64%	240.85B
16	8		✓				81.15%	2149.91B
16	8		✓	✓			81.34%	4299.68B
16	16	✓					79.93%	89.76B
16	16	✓	✓				81.38%	790.12B
16	16	✓	✓	✓			81.44%	1580.10B
16	8	✓					80.22%	262.59B
16	8	✓	✓				81.60%	2338.15B
16	8	✓	✓	✓			81.63%	4676.16B
16	16	✓			✓		79.79%	54.17B
16	16	✓	✓	✓	✓		81.21%	928.81B
16	8	✓			✓		80.02%	177.10B
16	8	✓	✓	✓	✓		81.39%	3055.35B
16	16	✓			✓	✓	82.20%	54.17B
16	16	✓	✓	✓	✓	✓	83.34%	928.81B
16	8	✓			✓	✓	82.45%	177.10B
16	8	✓	✓	✓	✓	✓	83.58%	3055.35B
16	16	✓			✓	✓	83.03%	54.17B
16	16	✓	✓	✓	✓	✓	84.22%	928.81B
16	8	✓			✓	✓	83.39%	177.10B
16	8	✓	✓	✓	✓	✓	84.56%	3055.35B

Table 5. Inference strategy on the PASCAL VOC 2012 *val* set when using modified *Xception*. **train OS**: The *output stride* used during training. **eval OS**: The *output stride* used during evaluation. **Decoder**: Employing the proposed decoder structure. **MS**: Multi-scale inputs during evaluation. **Flip**: Adding left-right flipped inputs. **SC**: Adopting depthwise separable convolution for both ASPP and decoder modules. **COCO**: Models pretrained on MS-COCO. **JFT**: Models pretrained on JFT.

- Depthwise convolution을 **ASPP**과 **Decoder module**에 적용하면 성능은 비슷하게 유지하면서 연산을 줄여줌
- 추가 **pre-train한 모델을 사용**할 때 성능이 향상

ex. pre-train된 모델을 사용할 때, 위 사진에서 MS는 MS-COCO dataset을 이용해 DeepLab v3+ 모델 전체를 사전학습한 것이고, JFT는 ImageNet-1k와 JFT-300M dataset

모두를 이용해 backbone network를 사전학습한 것

Method	mIOU
Deep Layer Cascade (LC) [82]	82.7
TuSimple [77]	83.1
Large_Kernel_Matters [60]	83.6
Multipath-RefineNet [58]	84.2
ResNet-38_MS_COCO [83]	84.9
PSPNet [24]	85.4
IDW-CNN [84]	86.3
CASIA_IVA_SDN [63]	86.6
DIS [85]	86.8
DeepLabv3 [23]	85.7
DeepLabv3-JFT [23]	86.9
DeepLabv3+ (Xception)	87.8
DeepLabv3+ (Xception-JFT)	89.0

Table 6. PASCAL VOC 2012 test set results with top-performing models.

DeepLab v3+가 sota 성능을 보여줌

4.4. Improvement along Object Boundaries

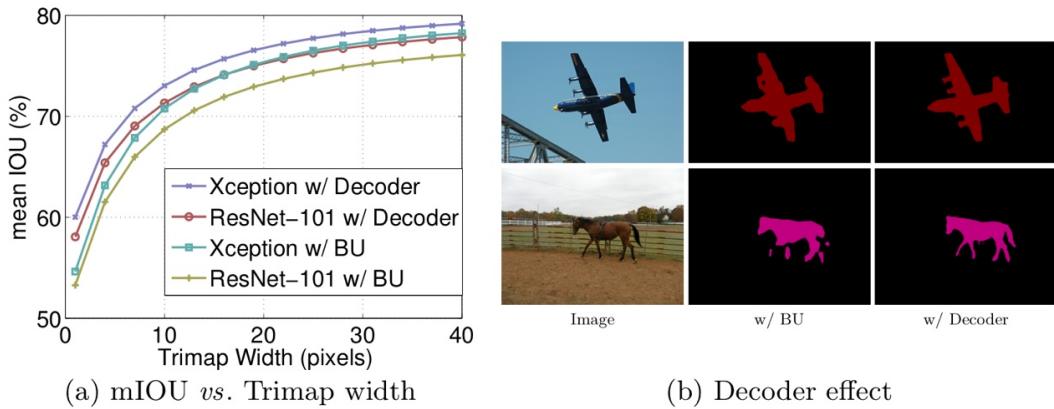


Fig. 5. (a) mIOU as a function of trimap band width around the object boundaries when employing $\text{train output stride} = \text{eval output stride} = 16$. BU: Bilinear upsampling. (b) Qualitative effect of employing the proposed decoder module compared with the naive bilinear upsampling (denoted as BU). In the examples, we adopt Xception as feature extractor and $\text{train output stride} = \text{eval output stride} = 16$.

Decoder를 사용한 모델이 Bilinear upsample 모델보다 성능이 더 좋고 객체의 경계선을 더 잘 포착

4.5. Experimental Results on Cityscapes

Backbone	Decoder	ASPP	Image-Level	mIOU
X-65		✓	✓	77.33
X-65	✓	✓	✓	78.79
X-65	✓	✓		79.14
X-71	✓	✓		79.55

(a) *val* set results

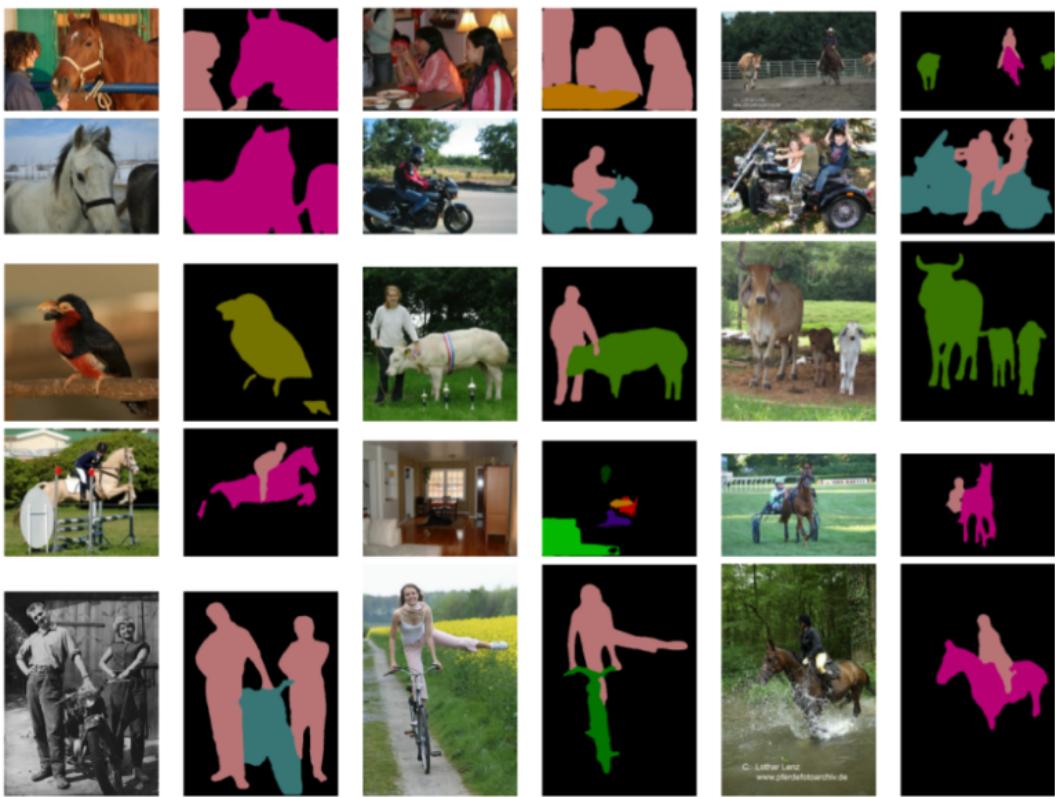
Method	Coarse	mIOU
ResNet-38 [83]	✓	80.6
PSPNet [24]	✓	81.2
Mapillary [86]	✓	82.0
DeepLabv3	✓	81.3
DeepLabv3+	✓	82.1

(b) *test* set results

Table 7. (a) DeepLabv3+ on the Cityscapes *val* set when trained with *train_fine* set.
(b) DeepLabv3+ on Cityscapes *test* set. **Coarse:** Use *train_extra* set (coarse annotations) as well. Only a few top models are listed in this table.

- Backbone network를 깊은 모델을 사용할 때 성능이 향상
- Cityscapes dataset은 PASCAL VOC 2012 dataset과 다르게 low-level image feature를 사용하지 않을 때 성능이 더 향상됨
- DeepLab v3+가 다른 모델들 보다 좋은 성능을 보임

5. Conclusion



위 Visualization 결과를 보면 상당히 안정적이고 정확하게 각각의 픽셀에 대해 클래스를 예측

1. Xception 기반의 encoder

→ 양질의 high level semantic 정보를 가지는 feature를 추출

2. ASPP 모듈

→ 각 픽셀이 여러 스케일의 context 정보를 취해 보다 정확한 추론

3. U-Net 구조의 decoder

→ 각 물체에 해당하는 정교한 boundary를 그려낼 수 있음

6. Reference

논문1. DeepLab V1 : <https://arxiv.org/pdf/1412.7062.pdf>

논문2. DeepLab V2 : <https://arxiv.org/pdf/1606.00915.pdf>

논문3. DeepLab V3 : <https://arxiv.org/pdf/1706.05587.pdf>

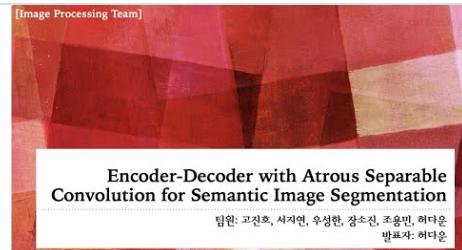
논문4. DeepLab V3+ : <https://arxiv.org/pdf/1802.02611.pdf>

- Youtube

DeepLabV3+ - 허다운

딥러닝논문스터디 - 34번째 이미지처리팀 허다운님의 'DeepLab: Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation' 입니다. 모임 참여 및 문의는

▶ <https://www.youtube.com/watch?v=TjHR9Z9iNLA>



- Blog

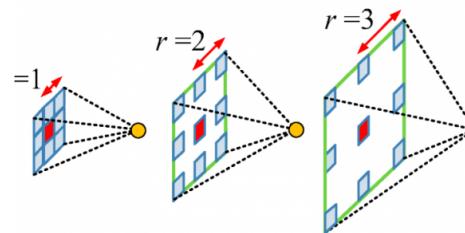
[Semantic Segmentation] DeepLab v3+ 원리

논문1. DeepLab V1 : <https://arxiv.org/pdf/1412.7062.pdf> 논문2.

DeepLab V2 : <https://arxiv.org/pdf/1606.00915.pdf> 논문

3. DeepLab V3 : <https://arxiv.org/pdf/1706.05587.pdf> 논문4.

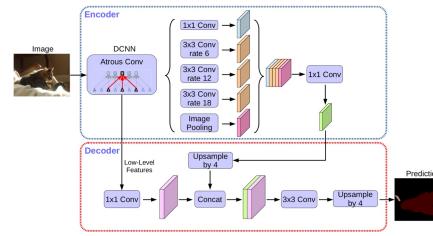
▶ <https://kuklife.tistory.com/121>



DeepLab v3+ 논문 리뷰

논문 제목: Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation Spatial pyramid pooling(SPP)
와 encoder-decoder 구조는 semantic segmentation에서 사용된다.

▶ <https://velog.io/@skhim520/DeepLab-v3>



[Deep Learning] 딥러닝에서 사용되는 다양한 Convolution 기법들

또한, 영상 내의 객체에 대한 정확한 판단을 위해서는 Contextual Information이 중요하다. 가령, 객체 주변의 배경은 어떠한 환경인지, 객체 주변의 다른 객체들은 어떤 종류인지 등. Object Detection이나 Object

▶ <https://eehoeskrap.tistory.com/431>

