Fully Convolutional Networks for Semantic Segmentation

Abstract & Introduction

- convolutional network는 recognition 분야에 발전이 있음. 최근에는 전체 이미지를 분류하는데 그치지 않고, 구조화된 output이 필요한 local 작업에도 진보가 있음 ex. Bounding Box Object Detection 객체 탐지. keypoint 예측. local correspondence 국소적 대응 등

→ coarse 거친 예측에서 fine한 예측으로 나아가기 위해선 **모든 픽셀에 대해 예측**해야 함

- convolutional network는 계층적인 feature를 추출할 수 있는 cv model로 본 논문은 **convolution network만을 사용해서 end-to-end 학습으로 픽셀 단위로 학습** 시키는 방식으로 semantic segmentation 분야에서 SOTA를 달성할 수 있음을 보여줌 + **supervised pre-training**로 시작한 연구는 이게 처음!
    - 기존 네트워크의 FCN 버전은 아무 크기의 입력에도 dense output을 예측할 수 있었음. 학습+추론을 한번에 처리하는 방식을 dense feedforward + backpropagation 을 통해 이루어짐. 이때 네트워크 내 upsampling layer는 pixel 단위 예측과 학습을 가능하도록 해줌 even if pooling으로 인해 해상도가 낮아졌다고 하더라도

      → 이론적으로, 실제로 매우 효율적, 다른 연구에서 사용되는 복잡한 다른 기법들을 사용할 필요가 없음. 기존에는 patchwise training (패치 단위로)가 흔했지만 이는 FCN만큼 효율적이지 않음

      → 이는 기존 R-CNN, YOLO 등에서 사용되었던 영역 제안, superpixel, random field나 local classifier와 같은 후처리나 복잡한 전처리를 사용하지 않음

- 여기서 핵심 아이디어는 input size에 상관없이 처리할 수 있고 그에 대응하는 크기의 output을 만들어내는 fully convolutional network를 구축해내는 것 → 효율적인 추론과 학습이 가능하도록 함
- FCV의 개념과 공간적으로 밀집된 예측 작업에 어떻게 활용될 수 있는 지 설명, 그리고 기존 모델들과의 연관성을 짚어보겠음. 또한 기존 이미지 분류 네트워크인 AlexNet,

VGGNet, GoogLeNet 을 FCN 구조로 바꾸고 그들이 학습한 표현을 **semantic segmentation** → **무엇인지 알기 위해선 전체적인 의미 semantic for what + 어디에 있는지 알기 위한 위치 정보 spatial for where**작업에 맞춰 fine tuning 을 진행

    - local → global 확장되는 피라미드 구조 속에서 encodnig + deep 의미 정보와 얕고 정밀한 정보를 결합하기 위한 skip architecture를 정의함

- 본 논문에서는 semantic information from deep, coarse layers + appearance information from shallow, fine layers → 정확하고 상세한 분할 결과를 생성하는 새로운 아키텍처를 정의함
- 또한 이전 연구들에서 사전학습 없이 작은 convnet 만 적용한 경우와 달리 본 논문에서는 학습된 표현으로 부터 fine tuning을 통해 dense prediction 문제로 transfer

Related Works

- 전이학습은 visual regconition에서 처음 도입되었고, 분류, 탐지, 분할로 전이되어 사용됨
- 최근에는 하이브리드 proposal 기반 classifier를 넘어, 직접적인 밀도 예측(dense prediction)을 위한 순수한 classification 네트워크로 확장됨
- 임의 크기의 입력에 대응할 수 있도록 합성곱 네트워크를 확장하는 아이디어는
  Matan 에서 처음 등장 : LeNet 을 확장해 우편번호 인식을 위한 1차원 문자열 입력을 처리. 이때
  입력이 1차원이었기에,
  Viterbi 디코딩으로 결과를 얻음
- 최근의 deep network에서 FCN은 다음과 같이 확장됨 → sliding window 방식, semantic segmentation, 이미지 복원 등

Method

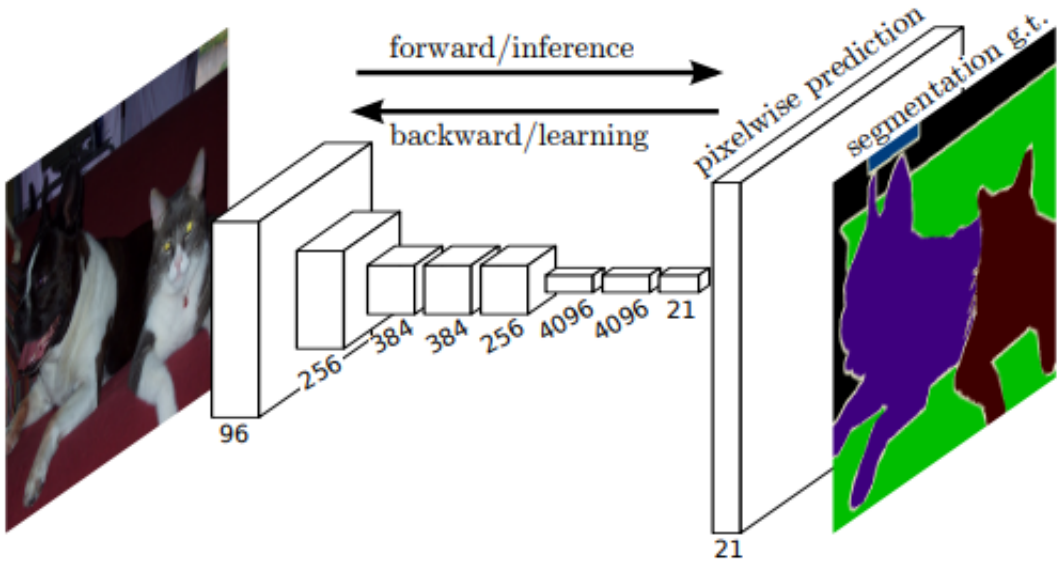

Figure 1. Fully convolutional networks can efficiently learn to make dense predictions for per-pixel tasks like semantic segmentation.

- Convnet의 각 데이터 계층은 h×w×d 크기의 3차원 배열 → 이미지이며, 픽셀 크기 hxw, d개의 색상 채널. 번째 계층은 이미지 자체이며, 상위 계층의 각 위치는 이미지 내 특정 위치와 연결된 receptive field를 가짐. Convnet은 변환 불변성을 기반으로 하며, 합성곱, 풀링, 활성화 함수 같은 연산은 지역 입력에 대해 작동함
- FCN(완전 합성곱 네트워크)은 이러한 구조를 활용하여 임의 크기의 입력을 처리하고 공간적으로 대응하는 출력을 생성할 수 있음

$$\mathbf{y}_{ij} = f_{ks}\left(\{\mathbf{x}_{si+\delta i, sj+\delta j}\}_{0 \le \delta i, \delta j \le k}\right)$$

fks로 계층 유형을 결정함 → 행렬 곱 혹은 비선형 transformation 등

$$f_{ks} \circ g_{k's'} = (f \circ g)_{k'+(k-1)s', ss'}.$$

FCN과 loss function을 결합하면 특정 작업이 정의됨

  - FCN과 손실 함수를 결합하면 특정 작업이 정의되며, 최종 계층의 손실을 전체 이미지 단위로 최적화가 가능함 → **따라서 임의 크기의 입력을 자연스럽게 처리하고 대응하는 공간 차원의 출력 생성이 가능함**
  - receptive field가 겹칠 경우, 전체 이미지를 한 번에 처리하는 것이 패치별로 독립 처리(feed forward + backpropagation)하는 것보다 계산적으로 더 효율적임

Adapting classifiers for dense prediction

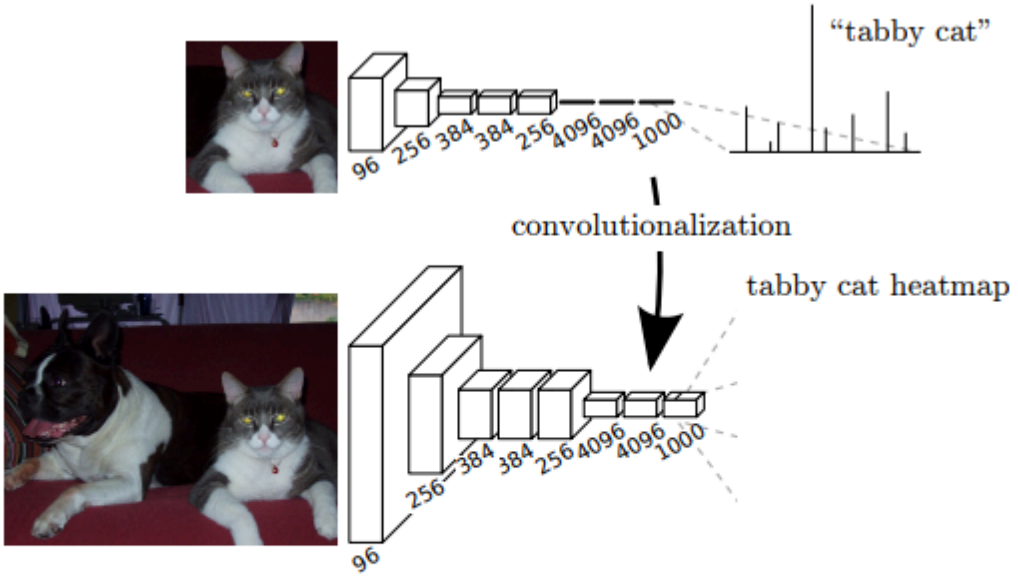

Figure 2. Transforming fully connected layers into convolution layers enables a classification net to output a heatmap. Adding layers and a spatial loss (as in Figure 1) produces an efficient machine for end-to-end dense learning.

- LeNet, AlexNet 그리고 더 깊은 후속 모델의 전형적인 recognition neural network는 일반적으로 고정 크기의 입력을 받아 공간적 정보가 없는 출력을 생성함 → 이러한 네트워크의 fully connected layer는 고정된 차원을 가지며 공간 좌표를 없앰 → **이때 이러한 완전 연결 계층을 입력 영역 전체를 덮는 커널을 가진 합성곱으로 해석이 가능** : FCN으로 변환 **→ 임의 크기의 입력을 받아 classifier map을 출력하는 구조가 됨**
- 변환된 모델이 생성하는 출력 맵은 원래 네트워크가 특정 입력 패치에서 수행한 평가와 동일하지만, 중첩된 패치들의 계산이 크게 최적화되어 수행됨. FCN 버전은 단순한 패치별 접근 방식보다 약 5배 빠름
- 이러한 합성곱 기반 모델의 공간적 출력 맵은 semantic segmentation 과 같은 dense prediction 문제에 적합한 b.c. 모든 출력 cell에 대해 ground truth가 제공되므로 forward pass + backpropagation 과정이 단순하고, 성곱의 본질적인 계산 효율성과 최적화의 이점을 그대로 활용 가능
- 본 논문에서는 분류 네트워크를 완전 합성곱 네트워크로 재해석하여 임의 크기의 입력에 대한 출력 맵을 생성할 수 있도록 했지만, 출력 차원은 일반적으로 subsampling으로 인해 감소함 → 분류 네트워크는 필터 크기를 작게 유지하고 계산량을 합리적인 수준으로 유지하기 위해 하위 샘플링을 수행 → 이때 출력이 원본 입력보다 tough 해짐.

Shift-and-stitch is filter rarefaction

- input shifting and output interlacing는 OverFeat에서 도입한 개념으로 interpolation 없이 tough한 출력을 dense prediction으로 변환하는 방법임
- 출력이 f 만큼 다운샘플링되었다면, 입력을 왼쪽과 위쪽 패딩을 사용하여 x 픽셀 오른쪽으로, y 픽셀 아래쪽으로 이동 → 모든 경우에 대해 각각 변환함

→ 이렇게 얻은 f^2 입력을 ConvNet에 통과시키고, 출력을 엇갈려 배치하여 예측 값이 receptive field 의 중심 픽셀에 대응하도록함

$$f'_{ij} = \begin{cases} f_{i/s,j/s} & \text{if } s \text{ divides both } i \text{ and } j; \\ 0 & \text{otherwise,} \end{cases}$$

- 같은 효과를 내기 위해 convnet의 stride와 filter 크기를 조정하는 방법도 있음. 하지만 필터가 희소해지는 문제가 있음.

→ Shift-and-stitch는 receptive field 크기를 유지하면서 output을 밀집화하는 대신, finer scale 정보를 포착하는 능력이 감소하는 trade-off가 있음. 따라서 실험 진행하였지만 최종 모델에는 사용하지 않음. 대신 upsampling을 활용한 end-to-end learning이 더 효율적이라고 판단함.

Upsampling is backwards strided convolution

- Dense prediction을 위해 interpolation(예: bilinear interpolation)도 가능하지만, deconvolution (a.k.a transposed convolution)이 더 효과적임.
- Upsampling by factor fff은 사실상 stride 1/f1/f1/f로 하는 convolution과 동일한 개념임.
- Deconvolution을 사용하면 네트워크 내에서 upsampling을 수행할 수 있으며, pixelwise loss를 이용한 backpropagation이 가능함.
- 단순한 bilinear upsampling filter를 고정하는 대신, 학습 가능한 deconv layers를 쌓으면 **nonlinear upsampling**도 가능함.
- 실험 결과, in-network upsampling이 dense prediction을 학습하는 데 더 빠르고 효과적이었으며, 최적의 segmentation 모델에서 이를 활용함.

Patchwise training is loss sampling

- Stochastic optimization에서 gradient 계산은 training distribution에 의해 결정됨.
- Patchwise training과 fully convolutional training 둘 다 특정 분포를 만들 수 있지만, 효율성은 overlap과 minibatch 크기에 따라 다름.

- Fully convolutional training = 각 이미지의 모든 receptive field를 포함하는 patchwise training임.
- Random patch selection은 gradient 계산에서 특정 패치를 제외하는 방식으로 쉽게 복구 가능함.
- 패치 간 overlap이 크면, fully convolutional 방식이 여전히 훈련 속도를 높일 수 있음.
- Class imbalance는 patchwise training에서는 sampling으로, fully convolutional training에서는 loss weighting으로 해결 가능함.
- Sampling을 활용한 학습을 실험했으나, dense prediction에서는 속도나 성능 향상이 없었음. Whole image training이 더 효과적이고 효율적임.

Segmentation Architecture

- ILSVRC classifier들을 Fully Convolutional Networks으로 변환하고, upsampling + pixelwise loss를 추가하여 dense prediction을 수행함.
- Segmentation 학습을 위해 fine-tuning 진행함.
- Coarse한 semantic 정보와 local appearance 정보를 결합하는 **skip architecture**를 제안하여 예측 성능 개선함.
- PASCAL VOC 2011 데이터셋으로 학습 및 검증함.
- Per-pixel multinomial logistic loss 사용 + mean IU(metric)로 성능 평가.

From classifier to dense FCN

- 기존 classification 모델(AlexNet, VGG, GoogLeNet)을 **convolutionalized**하여 FCN으로 변환함.
- Fully connected layer → convolution layer로 변환 후, **1×1 conv layer** 추가하여 PASCAL 클래스별 점수를 예측함.
- **Deconvolution layer**로 coarse output을 bilinear upsampling하여 pixel-dense output 생성함.

Table 1. We adapt and extend three classification convnets to segmentation. We compare performance by mean intersection over union on the validation set of PASCAL VOC 2011 and by inference time (averaged over 20 trials for a 500×500 input on an NVIDIA Tesla K40c). We detail the architecture of the adapted nets as regards dense prediction: number of parameter layers, receptive field size of output units, and the coarsest stride within the net. (These numbers give the best performance obtained at a fixed learning rate, not best performance possible.)

	FCN-AlexNet	FCN-VGG16	FCN-GoogLeNet[4]
mean IU	39.8	**56.0**	42.5
forward time	50 ms	210 ms	59 ms
conv. layers	8	16	22
parameters	57M	134M	6M
rf size	355	404	907
max stride	32	32	32

- FCN-VGG16이 56.0 mean IU로 기존 state-of-the-art 성능(52.6 mean IU)을 뛰어넘음.
- 추가 데이터 학습 시 59.4 mean IU까지 향상됨.
- GoogLeNet의 classification 성능은 좋았지만, segmentation 성능은 기대에 미치지 못함.

Combining what and where

- Segmentation을 위한 새로운 **Fully Convolutional Network (FCN)** 설계함.
- 기존 FCN (FCN-32s)은 classification 모델을 변환하여 segmentation에 적용할 수 있지만, 최종 prediction layer의 stride가 **32 픽셀**이라 upsampling한 output이 너무 coarse함.
- 이를 해결하기 위해 **finer stride를 가진 lower layers와 최종 prediction layer를 연결**하는 skip 구조 추가함.
- 네트워크 구조가 선형(line topology)에서 **DAG (Directed Acyclic Graph)** 형태로 변경됨.
- **Fine한 layer와 coarse한 layer를 결합**하면 **local prediction이 global 구조를 고려하면서 이루어짐**.

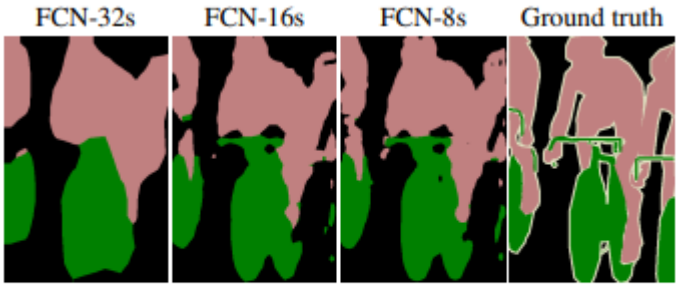

Figure 4. Refining fully convolutional nets by fusing information from layers with different strides improves segmentation detail. The first three images show the output from our 32, 16, and 8 pixel stride nets (see Figure 3).

[FCN-16s: Pool4를 활용한 Skip Architecture]

- 최종 stride를 **16 픽셀**로 줄이기 위해 **pool4 layer에서 예측을 수행**함.
- Pool4 위에 **1×1 convolution layer** 추가하여 class score 예측.
- **Conv7(fc7)에서 얻은 stride 32의 prediction을 2× upsampling**하여 pool4의 prediction과 합침.
- Upsampling은 bilinear interpolation으로 초기화하되, 학습 가능하도록 설정함.
- 이후, stride 16 prediction을 다시 원본 이미지 크기로 upsampling하여 최종 output 생성 → **FCN-16s**.
- FCN-32s의 parameter를 초기값으로 사용하고, pool4 관련 파라미터는 0으로 초기화하여 기존 예측을 유지하도록 학습 시작.
- 학습률을 **100배 감소**시켜 안정적인 학습 진행.
- 결과적으로 **mean IU 62.4**로 향상되었으며, finer structure 개선됨.

[FCN-8s: Pool3 추가로 더욱 정교한 예측]

- **Pool3의 prediction을 pool4 + conv7의 fused prediction과 2× upsampling하여 결합**.
- 최종적으로 **FCN-8s** 모델을 만들었으며, 성능이 **62.7 mean IU**로 약간 향상됨.
- Output이 더 부드럽고 세밀해졌지만, 추가적인 fusion의 효과는 점점 감소하는 **diminishing returns** 단계에 도달함.
- **IU metric과 시각적 품질 향상을 고려했을 때, 더 낮은 layer와의 fusion은 진행하지 않음**.

Table 2. Comparison of skip FCNs on a subset of PASCAL VOC2011 validation[7]. Learning is end-to-end, except for FCN-32s-fixed, where only the last layer is fine-tuned. Note that FCN-32s is FCN-VGG16, renamed to highlight stride.

	pixel acc.	mean acc.	mean IU	f.w. IU
FCN-32s-fixed	83.0	59.7	45.4	72.0
FCN-32s	89.1	73.3	59.4	81.4
FCN-16s	90.0	75.7	62.4	83.0
FCN-8s	**90.3**	**75.9**	**62.7**	**83.2**

- **Pooling layer의 stride를 줄이는 방식이 가장 간단한 refinement 방법**이지만, VGG16 기반 네트워크에서는 문제가 발생함.
  - Pool5의 stride를 1로 설정하면 fc6의 커널 크기를 **14×14**로 유지해야 하며, 이는 연산 비용이 크고 학습이 어려움.
  - Pool5 이후의 layer를 작은 필터로 재설계하려 했지만, **성능이 기존 모델보다 떨어짐**.
  - 원인은 **ImageNet으로 pretrain된 upper layer의 가중치 초기화가 중요한 역할을 하기 때문**일 가능성이 있음.
- **Shift-and-Stitch 기법**을 사용해 finer prediction을 시도했지만, **layer fusion 대비 효율성이 낮음**.

Experiment & Result

Experimental framework

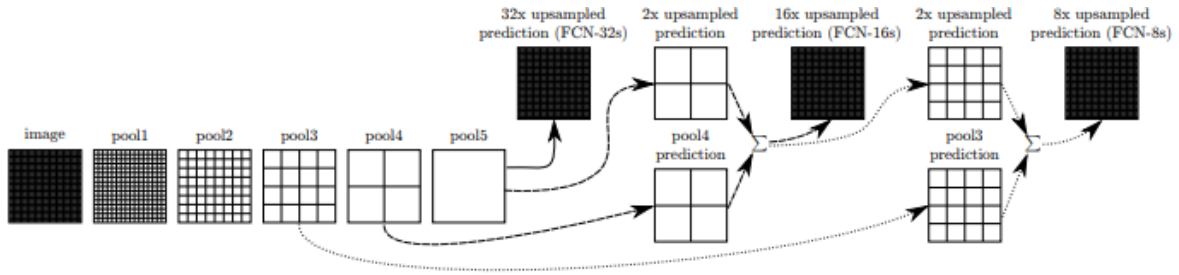

Figure 3. Our DAG nets learn to combine coarse, high layer information with fine, low layer information. Layers are shown as grids that reveal relative spatial coarseness. Only pooling and prediction layers are shown; intermediate convolution layers (including our converted fully connected layers) are omitted. Solid line (FCN-32s): Our single-stream net, described in Section 4.1, upsamples stride 32 predictions back to pixels in a single step. Dashed line (FCN-16s): Combining predictions from both the final layer and the `pool4` layer, at stride 16, lets our net predict finer details, while retaining high-level semantic information. Dotted line (FCN-8s): Additional predictions from `pool3`, at stride 8, provide further precision.

[**Optimization**]

- SGD with Momentum 사용, Minibatch size: 20, LR은 line search로 최적화 , Momentum: 0.9, Weight Decay: 5×10^{-4} 또는 2×10^{-4}

- Bias 학습률은 2배로 증가시켰으나, 전반적인 학습은 LR에 가장 민감함.
- Class scoring convolution layer는 zero-initialization 사용 → Random initialization 대비 성능/수렴 속도 개선되지 않음.
- Dropout은 원래 classifier에서 사용된 경우에만 유지함.

[**Fine-tuning**]

- 전체 네트워크를 end-to-end로 fine-tuning함.
- 최종 classifier layer만 fine-tuning할 경우 성능이 70%에 그침 (Table 2 참조).
- 처음부터(scratch) 학습하는 것은 비효율적임 (기존 classification network 학습에 시간이 과도하게 소요됨).
- Fine-tuning 소요 시간:
  - FCN-32s: 3일 (Single GPU 기준)
  - FCN-16s, FCN-8s 업그레이드: 각각 1일 추가 소요.

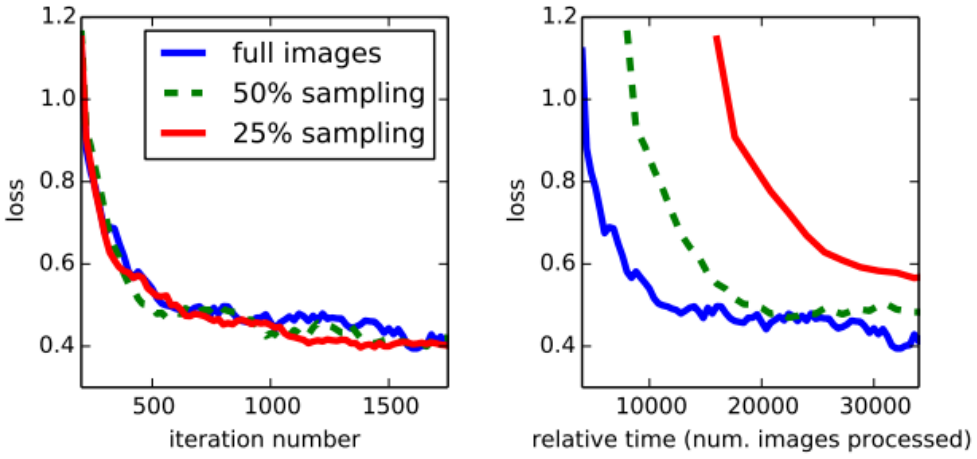

Figure 5. Training on whole images is just as effective as sampling patches, but results in faster (wall time) convergence by making more efficient use of data. Left shows the effect of sampling on convergence rate for a fixed expected batch size, while right plots the same by relative wall time.

[Patch Sampling]

- FCN은 이미지 전체를 큰 겹치는 patch grid로 학습하는 반면, 기존 연구들은 patch를 무작위로 샘플링하는 방식을 사용함.

- 무작위 샘플링은 batch variance를 높여 수렴 속도를 향상시킬 수 있다는 장점이 있음 → 실험 결과: 샘플링 방식은 **수렴 속도에는 큰 차이가 없었으나, batch 크기 증가로 인해 전체 학습 시간이 늘어남.**
    - 결론적으로 전체 이미지 학습 방식이 더 효율적하여 이후 실험에서도 이 방식을 채택.

[Class Balancing]

- FCN은 클래스 가중치 조정 또는 샘플링으로 class balance 가능.능→ PASCAL 데이터셋이 약간 불균형적 (배경이 75%)이지만, 추가적인 class balancing이 필요하지 않음.

[Dense Prediction]

- Deconvolution layers를 사용해 점수를 원본 크기로 upsample최종 deconvolution 필터 → Bilinear interpolation 고정.
- 중간 upsampling layer → Bilinear interpolation으로 초기화 후 학습 → 이때 Shift-and-stitch 및 filter rarefaction은 사용하지 않음.

[Data Augmentation]

- 랜덤 mirroring + 최대 32픽셀 translation jittering 적용 → 성능 향상 없음 → Augmentation 효과 미미.

[More Training Data]

- PASCAL VOC 2011 학습셋: 1,112개 이미지 + Hariharan et al. (2014) 데이터셋 추가 → 8,498개 이미지 학습 가능.
    - 추가 학습 데이터 사용 시 FCN-VGG16 성능 3.4점 향상 (Mean IU: 59.4)

Results

Table 3. Our fully convolutional net gives a 20% relative improvement over the state-of-the-art on the PASCAL VOC 2011 and 2012 test sets, and reduces inference time.

	mean IU VOC2011 test	mean IU VOC2012 test	inference time
R-CNN [12]	47.9	-	-
SDS [16]	52.6	51.6	~ 50 s
FCN-8s	**62.7**	**62.2**	~ **175 ms**

Table 4. Results on NYUDv2. *RGBD* is early-fusion of the RGB and depth channels at the input. *HHA* is the depth embedding of [14] as horizontal disparity, height above ground, and the angle of the local surface normal with the inferred gravity direction. *RGB-HHA* is the jointly trained late fusion model that sums RGB and HHA predictions.

	pixel acc.	mean acc.	mean IU	f.w. IU
Gupta *et al.* [14]	60.3	-	28.6	47.0
FCN-32s RGB	60.0	42.2	29.2	43.9
FCN-32s RGBD	61.5	42.4	30.5	45.5
FCN-32s HHA	57.1	35.2	24.2	40.4
FCN-32s RGB-HHA	64.3	44.9	32.8	48.0
FCN-16s RGB-HHA	**65.4**	**46.1**	**34.0**	**49.5**

Table 5. Results on SIFT Flow[10] with class segmentation (center) and geometric segmentation (right). Tighe [33] is a non-parametric transfer method. Tighe 1 is an exemplar SVM while 2 is SVM + MRF. Farabet is a multi-scale convnet trained on class-balanced samples (1) or natural frequency samples (2). Pinheiro is a multi-scale, recurrent convnet, denoted RCNN$_3$ ($\circ^3$). The metric for geometry is pixel accuracy.

	pixel acc.	mean acc.	mean IU	f.w. IU	geom. acc.
Liu *et al.* [23]	76.7	-	-	-	-
Tighe *et al.* [33]	-	-	-	-	90.8
Tighe *et al.* [34] 1	75.6	41.1	-	-	-
Tighe *et al.* [34] 2	78.6	39.2	-	-	-
Farabet *et al.* [8] 1	72.3	50.8	-	-	-
Farabet *et al.* [8] 2	78.5	29.6	-	-	-
Pinheiro *et al.* [28]	77.7	29.8	-	-	-
FCN-16s	**85.2**	**51.7**	39.5	76.1	**94.3**

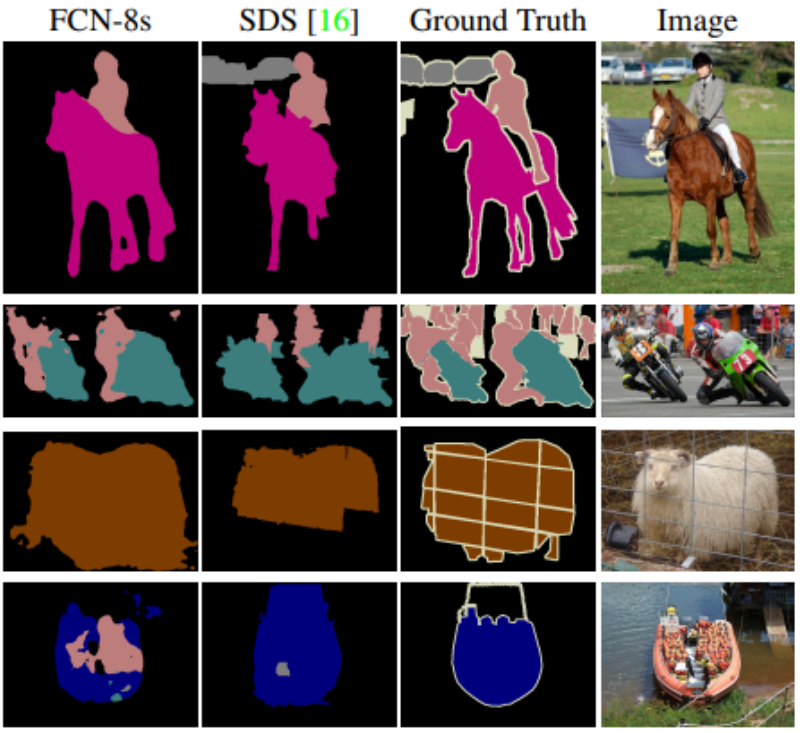

Figure 6. Fully convolutional segmentation nets produce state-of-the-art performance on PASCAL. The left column shows the output of our highest performing net, FCN-8s. The second shows the segmentations produced by the previous state-of-the-art system by Hariharan *et al.* [16]. Notice the fine structures recovered (first row), ability to separate closely interacting objects (second row), and robustness to occluders (third row). The fourth row shows a failure case: the net sees lifejackets in a boat as people.