SSD 논문

## & 기존 개체 방식과 SSD 비교 &

### 기존 방식

1. Bounding Box 후보를 생성 (Hypothesis Generation) -> 바운딩 박스 제안
2. 각 박스 내 픽셀 또는 Feature를 리샘플링(Resampling)하여 특징 추출
3. 고성능 분류기(Classifier)를 적용하여 객체 탐지 수행
-> 이 방식은 Faster R-CNN과 같은 모델이 대표적, PASCAL VOC, COCO, ILSVRC 등에서 높은 정확도를 기록

기존 방식의 문제점
- 연산량이 많아 계산 비용이 높음
- 임베디드 시스템에서 사용하기 어려움(속도가 상대적으로 느려서)

## ssd

- 기존 Faster R-CNN과 달리 "바운딩 박스 제안(Region Proposal) + 리샘플링" 과정 제거
  -> 단일 신경망에서 바로 탐지를 수행
- 기본 박스(Default Boxes)를 미리 설정하고, 네트워크가 한 번(One-Shot)에 모든 객체를 탐지
- 이 방식 덕분에 SSD는 낮은 해상도의 입력 이미지에서도 속도가 크게 향상, 기존 방법들과 유사한 정확도를 유지
- Faster R-CNN(7 FPS, 73.2% mAP)보다 훨씬 빠르고 (59 FPS, 74.3% mAP), YOLO(45 FPS, 63.4% mAP)보다 정확도가 훨씬 높음

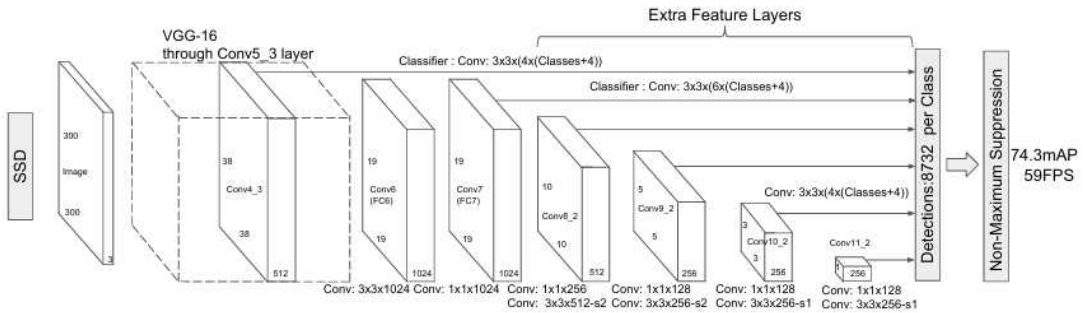
## # 실험 결과

- 300×300 입력 이미지 → PASCAL VOC2007에서 74.3% mAP, 59 FPS (Nvidia Titan X 기준)
- 512×512 입력 이미지 → 76.9% mAP, Faster R-CNN보다 높은 성능
- YOLO보다 높은 정확도를 제공하며, Faster R-CNN보다 속도가 빠름
결론: SSD는 빠르고 정확한 단일 단계 객체 탐지 모델로, 실시간 탐지 시스템에 적합한 모델임
mAP: 탐지 정확도  FPS: 탐지 속도

# 속도 향상의 핵심 요소
작은 컨볼루션 필터를 사용하여 객체 분류 및 바운딩 박스 보정 수행, 다양한 종횡비(Aspect

Ratio)를 고려하여 예측을 수행하는 개별 필터 사용
여러 Feature Map에서 다중 스케일(Multi-Scale) 탐지를 수행하여 작은 객체부터 큰 객체까지 탐지 가능 -> 단일 신경망에서 구현 가능 하게함
# 다중 스케일(feature maps)의 활용
서로 다른 해상도의 여러 feature map에서 예측을 결합함으로써, SSD는 작은 객체부터 큰 객체까지 다양한 크기의 객체를 효과적으로 탐지할 수 있습니다.

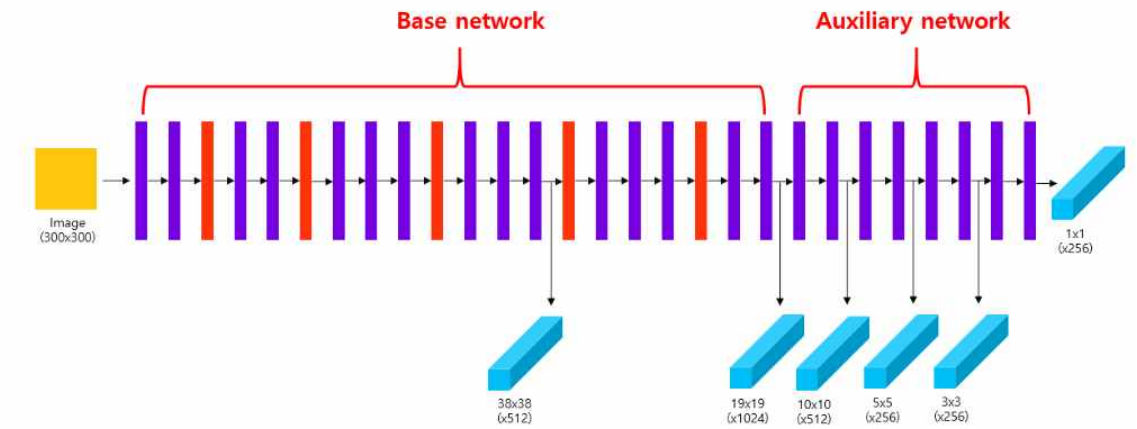
## SSD 아키텍처




(1) 기본 네트워크 (VGG-16 기반)
입력 이미지 크기: 300×300
SSD는 기존의 VGG-16 신경망의 Conv5_3 계층까지 사용하며, 이후 추가적인 컨볼루션 계층을 붙여 객체 탐지를 수행함.(VGG-16에서 분류 레이어 부분 제거하고탐지를 위한 보조 구조(Auxiliary Structure)가 추가)

(2) 추가 Feature Layers
기본 네트워크(VGG-16) 이후에 추가적인 컨볼루션 계층을 추가하여 탐지를 수행.
이러한 추가 계층들은 점점 크기가 작아지며(19×19 → 10×10 → 5×5 → 3×3 등)-> 작은 객체부터 큰 객체까지 탐지 가능(다중 스케일 예측 수행)

(3) 탐지 방식
각 Feature Map의 모든 위치에서 여러 개의 Default Boxes를 설정하여 탐지를 수행
작은 크기의 3×3 컨볼루션 필터를 이용해 각 박스의 객체 분류(classifier)와 바운딩 박스 조정(offset)을 예측
SSD의 탐지는 다중 스케일(Multi-Scale)에서 이루어짐 → 작은 feature map에서는 큰 객체, 큰 feature map에서는 작은 객체를 탐지
SSD network 중간에 존재하는 conv layer의 feature map들을 추출하여 detection 시 사용하는 방법을 제안




입력 이미지의 크기는 300x300 일때
1. base network conv4_3 layer에서 38x38x(512) 크기의 feature map을 추출
2. base network의 conv7 layer에서 19x19x(1024) 크기의 feature map을 추출
3. auxiliary network의 conv8_2, conv9_2, conv10_2, conv11_2 layer에서 각각 10x10x(512), 5x5x(256), 3x3x(256), 1x1x(256) 크기의 feature map을 추출
-> 총 6개의 scale을 가진 feature map을 얻음
결론: 다양한 scale의 feature map(Multiscale feature maps)을 사용, 다양한 크기의 객체를 탐지하는 것이 가능해짐

(4) 후처리 - Non-Maximum Suppression (NMS)
여러 박스가 같은 객체를 탐지했을 경우, 최고 신뢰도의 박스를 남기고 나머지를 제거하는 NMS 수행
최종적으로 8732개의 탐지 후보가 생성됨

(5) 성능
탐지 정확도 (mAP): 74.3%
속도 (FPS): 59 FPS
→ 높은 정확도를 유지하면서도 YOLO보다 빠름.

## Default Boxes and Aspect Ratios (기본 박스와 종횡비)

# 기본 박스(Default Boxes) 개념
네트워크는 여러 feature map 위치마다 다양한 크기와 종횡비를 가진 '기본 박스'들을 미리 정의합니다. 이 박스들은 잠재적인 객체 위치를 나타내며, 예측 과정에서 각 기본 박스에 대해 객체가 존재할 확률과 객체의 정확한 위치 조정값을 산출

기본 박스 배치 방식

- 각 Feature Map 위치(셀)마다 여러 개의 기본 박스를 배치
- 각 기본 박스는 다른 크기(Scale)와 종횡비(Aspect Ratio)를 가짐

- Feature Map 크기가 다르면 기본 박스의 크기도 달라짐 (다중 스케일 적용)
- 모든 기본 박스의 위치는 해당 Feature Map의 셀과 정렬됨 (Grid 방식으로 배치됨)
-> 기본 박스는 컨볼루션 방식으로 배치되며, 모든 위치에 균등하게 적용됨

기본 박스의 예측 방식

각 기본 박스마다 다음 두 가지를 예측
1. 객체 클래스 점수(Class Scores) → 객체가 포함될 확률 예측
2. 위치 보정 값(Offsets) → 기본 박스의 위치를 조정하여 객체에 맞춤

SSD는 각 **Feature Map에서 총** $(c+4)k$ **개의 필터를 사용하여 예측을 수행**

- $c$ = 클래스 개수
- $4$ = 바운딩 박스 오프셋(x, y, w, h)
- $k$ = 하나의 Feature Map 위치에서 설정된 기본 박스 개수

**최종 출력 크기**

- Feature Map 크기가 $m \times n$ 일 때,
- **총** $(c+4)kmn$ **개의 예측값이 생성됨**

-> 기본 박스를 활용하여 빠르게 객체 탐지 수행 가능!

## training

1. 기존 객체 탐지 방식과의 차이점

기존 방식 (Faster R-CNN) 객체 탐지를 위해 두 단계(two-stage)로 진행
먼저 후보 바운딩 박스를 생성 → 후보 바운딩 박스를 평가하여 최종 결과 도출 -> 속도가 상대적으로 느림.
SSD 방식
단일 신경망(single-shot network)에서 객체 탐지를 수행 영역 제안 없이, 기본 박스(Default Boxes)와 매칭되는 Ground Truth 박스를 직접 예측
Faster R-CNN처럼 영역 제안 후 평가하는 과정이 없음 → 속도가 훨씬 빠름.

## 매칭 전략 (Matching Strategy)

SSD는 기본 박스(Default Boxes)와 Ground Truth 바운딩 박스를 매칭하는 방식이 기존 방법(MultiBox)과 다릅니다.

1. 기존 방식 (MultiBox)
- 각 Ground Truth 박스는 IoU 기준으로 가장 잘 맞는 하나의 기본 박스와 매칭
  즉, GT 박스당 하나의 기본 박스만 선택하여 학습

2. SSD 방식
- SSD는 IoU가 0.5 이상인 모든 기본 박스를 GT 박스와 매칭
  따라서, 하나의 GT 박스가 여러 개의 기본 박스와 매칭될 수 있음
  이 방식 덕분에 네트워크가 여러 기본 박스에서 높은 점수를 예측하는 방법을 학습할 수 있음.

#ground truth 박스 : 정답

$$IoU = \frac{\text{교집합 영역}}{\text{합집합 영역}}$$

IoU 값이 1.00이면 두 박스가 완벽하게 겹침
IoU 값이 0.5 이상이면 일정 부분이 겹친다고 판단하여 매칭

## Training objective

SSD의 학습 목표는 MultiBox 방식 [7,8]에서 확장된 형태로, **다중 객체(Class) 탐지를 지원하도록 설계**되었습니다.
각 기본 박스(Default Box) $i$ 가 Ground Truth(정답) 박스 $j$ 와 매칭되었는지를 나타내는 **지표** $x_{ij}^{p}$ 는 **0 또는 1** 값을 가집니다.

위의 매칭 전략에서, 우리는 다음과 같은 조건을 가질 수 있습니다:

$$\sum_{i} x_{ij}^{p} \geq 1$$

**즉, 하나의 Ground Truth 박스(GT 박스)가 여러 기본 박스와 매칭될 수 있음**을 의미합니다.
전체 손실 함수(Objective Loss Function)는 **위치 손실(Localization Loss,** $L_{\text{loc}}$**)** 과 **신뢰도 손실(Confidence Loss,** $L_{\text{conf}}$**)** 의 가중합(weighted sum)으로 정의됩니다.

$$L(x, c, l, g) = \frac{1}{N}\left(L_{\text{conf}}(x, c) + \alpha L_{\text{loc}}(x, l, g)\right)$$

여기서,

- $N$ = **매칭된 기본 박스(Default Boxes)의 개수**
  - 만약 $N = 0$ (즉, GT가 없는 이미지)인 경우, 손실을 0으로 설정.
- $L_{\text{conf}}$ = **신뢰도 손실 (Confidence Loss)**
- $L_{\text{loc}}$ = **위치 손실 (Localization Loss)**
- $\alpha$ = **두 손실을 조정하는 가중치 (교차 검증을 통해** $\alpha = 1$ **로 설정됨)**

### 위치 손실 (Localization Loss, $L_{\text{loc}}$)

위치 손실은 예측된 바운딩 박스($l$) 와 GT 박스($g$) 의 차이를 최소화하는 역할을 하며,
이는 **Smooth L1 Loss** 를 사용하여 계산됩니다.

$$L_{\text{loc}}(x, l, g) = \sum_{i \in Pos} \sum_{m \in \{cx, cy, w, h\}} x_{ij}^{k} \text{smooth}_{L1}(l_m^i - \hat{g}_m^j)$$

여기서,

- $i \in Pos$ : **양성(Positive) 샘플**, 즉 Ground Truth와 매칭된 기본 박스만 고려.
- **위치 조정값 (Offset) 계산**
  - Faster R-CNN과 유사하게, **중심 좌표(center), 너비(width), 높이(height)를 회귀(regression) 방식으로 조정.**

$$\hat{g}_{cx}^{j} = \frac{(g_{cx}^{j} - d_{cx}^{i})}{d_w^i}, \quad \hat{g}_{cy}^{j} = \frac{(g_{cy}^{j} - d_{cy}^{i})}{d_h^i}$$

$$\hat{g}_w^j = \log\left(\frac{g_w^j}{d_w^i}\right), \quad \hat{g}_h^j = \log\left(\frac{g_h^j}{d_h^i}\right)$$

즉, 바운딩 박스의 **중심(cx, cy)** 를 정규화하여 학습하며, 너비(w)와 높이(h)에 대해 **로그 변환 적용** →
작은 변화에도 민감하게 학습 가능.

### 신뢰도 손실 (Confidence Loss, $L_{\text{conf}}$)

신뢰도 손실은 **객체 클래스 분류(Classification)** 를 위한 손실 함수이며,
**Softmax Cross-Entropy Loss** 를 사용합니다.

$$L_{\text{conf}}(x, c) = -\sum_{i \subset Pos} x_{ij}^{p} \log(c_i^p) - \sum_{i \subset Neg} \log(c_i^0)$$

여기서,

- **양성 샘플(Positive, $Pos$)**: GT와 매칭된 기본 박스 → 정답 클래스에 대한 Softmax Loss 적용.
- **음성 샘플(Negative, $Neg$)**: 배경(Background)으로 간주되는 기본 박스 → 배경 클래스(0번 클래스)에 대한 Softmax Loss 적용.

$$c_i^p = \frac{\exp(c_i^p)}{\sum_p \exp(c_i^p)}$$

## 기본 박스의 스케일(Scale) 및 종횡비(Aspect Ratio) 선택

기존 방식
–이미지를 여러 크기로 변환한 후, 각각 객체 탐지를 수행하고 최종 결과를 결합.
각 이미지 크기에 맞는 모델을 별도로 학습해야 하므로 연산량이 많고, 학습이 복잡

SSD
- 이미지 크기를 변환하지 않고, 네트워크의 서로 다른 계층(Layers)에 위치한 여러 Feature Map을 활용하여 다양한 크기의 객체를 탐지.
- 모든 객체 크기에 대해 동일한 파라미터를 공유 → 학습 효율성이 높아지고 연산량이 적음

SSD에서 기본 박스의 역할

1. 낮은 계층의 Feature Map → 작은 객체 탐지
   낮은 계층의 Feature Map은 해상도가 높고, 객체의 세부 정보(Fine Details)를 잘 보존
   따라서 작은 객체(Small Objects)를 탐지하는 데 적합

2. 깊은 계층의 Feature Map → 큰 객체 탐지
깊은 계층의 Feature Map은 더 넓은 영역(Global Context)를 포함
따라서 큰 객체(Large Objects) 탐지에 적합
SSD는 각 계층의 특징을 활용하여 작은 객체부터 큰 객체까지 효과적으로 탐지할 수 있도록 설계됨

## 기본 박스 배치와 스케일 설정

SSD에서 **각 Feature Map이 가지는 수용 필드(Receptive Field)는 서로 다름**이 알려져 있습니다 [13].
그러나, SSD에서는 기본 박스들이 특정 Feature Map의 실제 수용 필드에 정확히 대응할 필요는 없습니다.
대신, **기본 박스를 특정 객체 크기에 적응할 수 있도록 배치**합니다.

우리가 **m 개의 Feature Map을 사용하여 예측한다고 가정**할 때,
각 Feature Map에서 사용할 기본 박스의 스케일은 다음과 같이 계산됩니다:

$$s_k = s_{\min} + \frac{s_{\max} - s_{\min}}{m - 1}(k - 1), \quad k \in [1, m]$$

여기서,

- $s_{\min} = 0.2$, $s_{\max} = 0.9$ → **최소 Feature Map 스케일은 0.2, 최대는 0.9**
- 가장 작은 Feature Map (즉, 네트워크의 최상위 계층)은 **스케일 0.2** 를 사용하고,
- 가장 큰 Feature Map (즉, 네트워크의 최하위 계층)은 **스케일 0.9** 를 사용하며,
- 그 사이의 계층들은 **균등한 간격(Regularly Spaced)으로 배치됨.**

### 📌 수용 필드 (Receptive Field)란?

수용 필드( **Receptive Field** )란 **신경망(Neural Network)의 특정 뉴런이 입력 이미지에서 영향을 받을 수 있는 영역(픽셀 범위)** 을 의미합니다.
즉, **한 개의 뉴런이 얼마나 넓은 영역을 보고 있는지를 나타내는 개념**입니다.

#### 💡 예제

- 초기에 입력 이미지가 **256×256 크기**라고 가정
- 네트워크의 첫 번째 컨볼루션 필터(3×3)를 통과하면 **각 뉴런은 3×3 영역만큼의 정보를 보게 됨**
- 더 깊은 계층(예: 5층)에서는 이전 레이어에서 합쳐진 정보를 보기 때문에 **더 넓은 영역을 수용 필드로 가짐**
- **즉, 네트워크의 깊이가 깊을수록 수용 필드 크기가 커짐!**

📌 ➡️ **깊은 레이어로 갈수록 더 넓은 범위의 정보를 수용할 수 있음!**

## 기본 박스의 종횡비 설정

각 기본 박스는 **다양한 종횡비(Aspect Ratio)** 를 가져야 합니다.
우리는 기본 박스의 종횡비를 **다음과 같이 설정**합니다:

$$a_r \in \{1, 2, 3, \frac{1}{2}, \frac{1}{3}\}$$

각 기본 박스의 **너비(Width)와 높이(Height)** 는 다음과 같이 계산됩니다:

$$w_k^a = s_k\sqrt{a_r}, \quad h_k^a = s_k/\sqrt{a_r}$$

📌 **추가적인 기본 박스(Default Box)**

- **Aspect Ratio가 1인 경우**, 우리는 추가적인 기본 박스를 배치합니다.
- 이 기본 박스의 크기는 **기하 평균(Geometric Mean)** 을 사용하여 설정됨:

$$s_k' = \sqrt{s_k s_{k+1}}$$

- 결과적으로, **각 Feature Map의 위치마다 총 6개의 기본 박스(Default Boxes)가 사용됨.**

## 기본 박스의 위치 배치

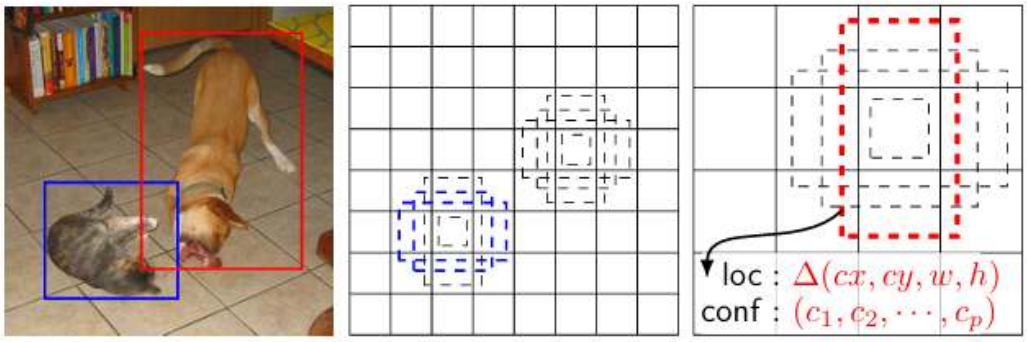
각 기본 박스는 **Feature Map의 격자(Grid) 위치마다 배치**되며,
그 중심 좌표는 다음과 같이 설정됩니다:

$$\left(\frac{i+0.5}{|f_k|}, \frac{j+0.5}{|f_k|}\right)$$

여기서,

- $|f_k|$ 는 **k번째 Feature Map의 크기** (예: 8×8, 4×4 등)
- $i, j$ 는 **Feature Map 내에서의 격자 좌표(Grid Position)**

즉, **각 Feature Map의 셀 중심(center)에 기본 박스를 배치**하여, 객체 탐지를 수행할 수 있도록 설계됩니다.




(a) Image with GT boxes (b) $8 \times 8$ feature map (c) $4 \times 4$ feature map

여러 크기의 기본 박스 결합 → 다양한 크기의 객체 탐지 가능

- 모든 Feature Map의 기본 박스들을 조합하여 예측을 수행하면, 다양한 크기와 종횡비의 객체를 탐지할 수 있음
- 그림 1을 보면, 개(Dog)는 8×8 Feature Map에서 탐지되고, 고양이(Cat)는 4×4 Feature Map에서 탐지됨
- 즉, 작은 객체는 작은 Feature Map에서, 큰 객체는 큰 Feature Map에서 탐지될 수 있도록 배치됨

# Hard Negative Mining

SSD와 같은 객체 탐지 모델에서는, 모델이 기본 박스를 기반으로 객체를 탐지
하지만 대부분의 기본 박스는 객체가 아닌 배경(Background)으로 분류되어 음성(Negative) 샘플이 됨.

예)
전체 기본 박스 수: 10,000개
Ground Truth와 매칭된 기본 박스(양성 샘플): 500개
객체가 없는 기본 박스(음성 샘플): 9,500개
➡ 양성(Positive) 샘플보다 음성(Negative) 샘플이 너무 많음
➡ 학습 데이터가 심각한 불균형(Class Imbalance) 문제를 가지게 됨
➡ 모델이 배경을 더 중요하게 학습할 가능성이 높아짐 (즉, 객체 탐지 성능이 낮아질 수 있음)

**Hard Negative Mining** 적용

음성 샘플이 너무 많으면 모든 음성 샘플을 학습하는 것이 비효율적 -> 손실이 가장 큰(즉, 모델이 가장 실수한) 음성 샘플을 선택하여 학습하는 방식이 Hard Negative Mining

과정
1. 각 기본 박스마다 Confidence Loss(신뢰도 손실)를 계산
2. 객체가 있을 확률을 예측한 값과 실제 Ground Truth(정답) 비교
3. Confidence Loss가 가장 높은(즉, 모델이 틀린) 음성 샘플부터 정렬

양성 샘플과 음성 샘플 비율을 3:1로 유지하여 학습에 사용
예: 양성 500개 → 음성 1,500개만 사용
➡ 이렇게 하면 불필요한 배경 학습을 줄이고, 모델이 중요한 음성 샘플을 학습할 수 있도록 유도함.

## 데이터 증강 (Data Augmentation)

각 학습 이미지에 대해 다음 옵션 중 하나를 랜덤하게 적용하여 샘플링(Sampling)을 수행니
1. 원본 이미지 전체를 그대로 사용
2. 객체와 최소 Jaccard Overlap이 0.1, 0.3, 0.5, 0.7, 또는 0.9가 되도록 패치를 샘플링
3. 랜덤하게 패치를 샘플링

- 샘플링된 패치(Patch)의 크기는 원본 이미지 크기의 10%~100%(즉, [0.1, 1]) 범위 내에서 무작위로 설정
- 종횡비(Aspect Ratio)는 1/2 ~ 2 사이에서 무작위로 선택
- 패치 내 GT 박스는, 중심이 해당 패치 내부에 포함된 경우 유지,
- 샘플링된 패치는 고정된 크기로 Resize되며, 50% 확률로 좌우 반전(Horizontal Flip)이 적용
- 기존 연구 [14]에서 제안된 포토메트릭(Photo-metric) 왜곡 효과(밝기, 대비 조정 등) 도 추가로 적용