■ 서식 1 : 결과보고서 요약본

Project Based Learning 결과보고서 요약본

<table>
<tr><td rowspan="2">교과목명</td><td>국문</td><td colspan="3">기계학습2</td></tr>
<tr><td>영문</td><td colspan="3">Machine Learning 2</td></tr>
<tr><td colspan="2">PBL 관련 능력단위
(능력단위코드)</td><td colspan="3">PBL 관련 능력단위요소 (능력단위요소코드)</td></tr>
<tr><td colspan="2" rowspan="2">기계학습(YCS-2001020910_21v1)</td><td colspan="3">신경망동작 이해하기</td></tr>
<tr><td colspan="3">프레임워크 사용하기</td></tr>
<tr><td colspan="2">학년 반</td><td>2-2</td><td>조원</td><td>이은우</td></tr>
<tr><td colspan="2">프로젝트 주제</td><td colspan="3">MxNet 프레임워크 사용하기</td></tr>
<tr><td colspan="2">지도교수</td><td>심효선</td><td>산업체 참가여부</td><td>□ 유 ■ 무</td></tr>
<tr><td colspan="2">참여학생</td><td colspan="3">이은우(202025016)</td></tr>
<tr><td colspan="2">작품 개요
(주제 선정 이유)</td><td colspan="3">==========
[1. 선형회귀 문제에서 학습한 모델의 패러미터 w, b 확인하기]
URL : https://d2l.ai/chapter_linear-regression/linear-regression-scratch.html
3장 Linear Neural Network for Regression
3-4. Linear Regression Implementation from Scratch

model = LinearRegressionScratch(2, lr=0.03)
data = d2l.SyntheticRegressionData(w=np.array([2, -3.4]), b=4.2)
trainer = d2l.Trainer(max_epochs=3)
trainer.fit(model, data)

==========
- 개요
교재의 LinearRegressionScratch 모델은 입력의 개수를 인자로 받는데 현재 예제에서는 입력 특징의 크기가 2로 되어 있다. 따라서 데이터세트를 만들때도 d2l.SyntheticRegressionData 함수는 인자 w의 길이가 2가 되어야 하고 데이터세트를 생성하기 위하여 인자 값 w는 [2, -3.4], b = 4.2 라고 가정하였다.

- 문제
모델의 입력특징의 크기가 3이고 데이터세트 생성을 위한 함수 d2l.SyntheticRegressionData의 인자의 값이 w = [3, 1, 2], b = 4.2 라고 설정하였을 때 모델을 훈련한 후에 모델의 패러미터 w, b의 값을 출력하여 초기에 설정했던 w = [3, 1, 2], b = 4.2 와 값이 얼마나 차이 나는지 확인하여 모델 학습이 잘 되었는지를 체크한다.</td></tr>
</table>

==========

[2. 프레임워크에서 제공하는 클래스의 API에 대해 이해하기]

URL : https://d2l.ai/chapter_linear-regression/index.html
3. Linear Neural Networks for Regression

==========

- 개요

프레임워크는 객체지향 기법을 이용하여 모델 클래스, 데이터 클래스, 트레이너 클래스를 제공하여 각 클래스가 가지고 있는 API를 이용하여 학습과 예측을 수행한다.

- 문제

1) 모델 클래스가 제공해주는 주요 함수의 API에 대해서 입력, 동작, 출력에 대하여 기술하시오

2) 데이터 클래스가 제공해주는 주요 함수의 API에 대해서 입력, 동작, 출력에 대하여 기술하시오

3) 트레이너 클래스가 제공해주는 주요 함수의 API에 대해서 입력, 동작, 출력에 대하여 기술하시오

==========

[3. 분류 문제에서 one-hot encoding, softmax argmax계산하기]

URL : https://d2l.ai/chapter_linear-classification/softmax-regression.html
4장 Linear Neural Network for Classification
4.1. Softmax Regression

==========

- 개요

분류 문제에서는 클래수의 개수가 n이면 출력층의 노드 개수도 n으로 지정하며 one-hot encoding 방식을 사용하며, 그 뒷단에 softmax 계층을 추가하는 것이 일반적이다. 출력단이 softmax의 형태인 경우에 손실함수로는 SoftmaxCrossEntrody 손실함수를 주로 적용한다.

- 문제

분류의 문제이고 클래스의 갯수는 3 (1:사과, 2:배, 3:감) 이고, 특정 훈련 데이터 (X, Y)를 가지고 모델에 적용하여 값을 예측하였다. 훈련데이터는 과일 사진이며 데이터세트는 (X, y) 인데 X는 사진, y는 레이블로서 1 ~ 3 중의 하나의 값을 가진다.

데이터의 y값이 2이고, 모델에 X를 입력으로 제공했을 때 출력의 결과가 [-0.4, 4.2, 0.3] 으로 나왔을 때,

1) 2에 해당하는 One-hot Encoding 구하기
2) [-0.4, 4.2, 0.3]에 해당하는 softmax 값 구하기. 이때 softmax 계산식을 구할 때 공학용 계산기를 이용하여 log 값을 구해야 됨
3) 2)번의 softmax 값을 입력으로 받아 argmax 값 계산하기
4) 참의값((1)에서 구한 One-hot Encoding값)과 예측값 ((2)에서 구한 softmax 값)을 이용하여 SoftmaxCrossEntropy 계산하기. 공학용 계산기를 이용하여 log 값을 구해야 됨

Y(데이터)	2		
OnehotEnc			
o1, o2, o3	-0.4	4.2	0.3
softmax			
argmax			
SoftmaxCrossEntrypy			

==========

[4. CNN 모델에서 Padding, Striding 개념 이해하기]

URL : https://d2l.ai/chapter_convolutional-neural-networks/index.html

7. Convolutional Neural Networks

==========

- 개요

CNN 모델에서는 2차원인 사진의 특성을 그대로 유지하기 위하여 2차원의 Kernel Window를 사용하여 원본 이미지에 convolution 연산을 적용한다. 이때 원본 이미지의 크기가 변하게 되는데 padding, striding의 연산을 통하여 출력의 크기를 조절하여 원본이미지의 특징을 추출한다.

- 문제

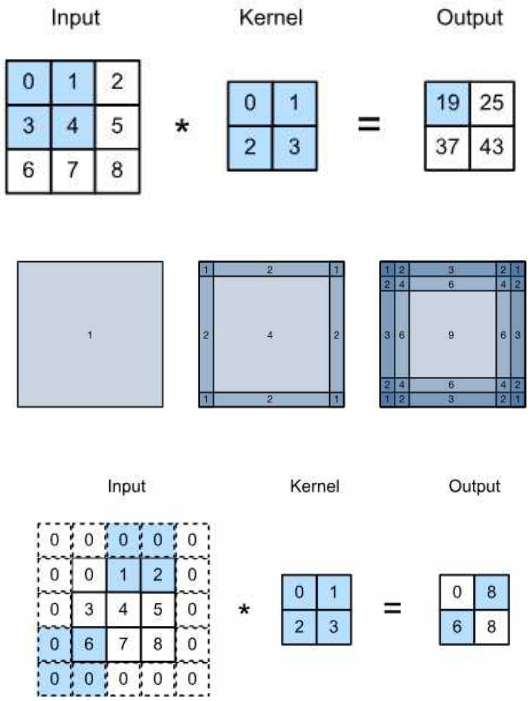
원본 사진의 크기는 320*240 이고 kernel window의 크기는 5*5 인 경우에

1) padding = 0, stride = 1 인 경우에 출력 결과의 크기를 계산하시오
stride가 1인 경우에 정상적으로 kernel window를 오른쪽으로,
그리고 아래로 1칸씩 이동한다.

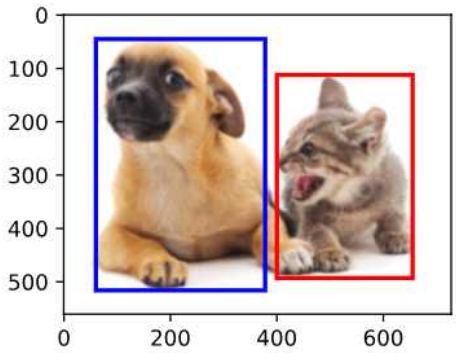
2) padding = 1, stride = 1 인 경우에 출력 결과의 크기를 계산하시오
padding이 1인 경우 상, 하, 좌, 우의 경계를 한 줄 늘이고 0의 값으로 채우게 된다.

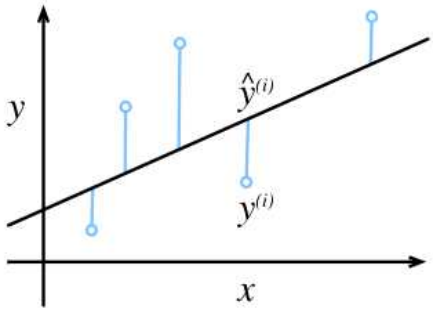
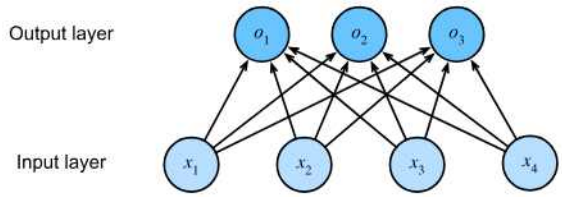
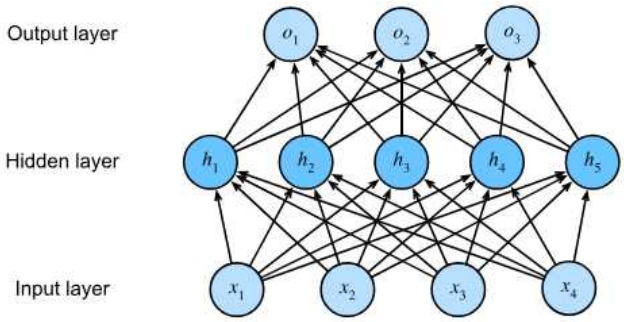
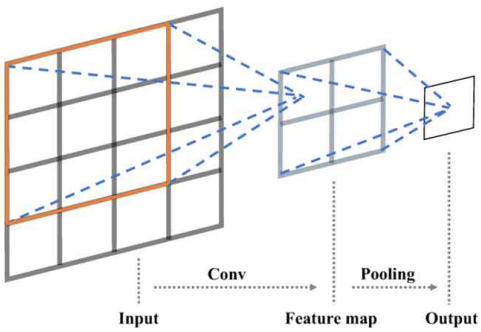
	========== [5. 객체인식 문제에서 화면에 Bounding Box 추가하기] URL : https://d2l.ai/chapter_computer-vision/bounding-box.html 14.3. Object Detection and Bounding Boxes ========== - 개요 분류 문제는 사진 속에 하나의 객체가 들어있다고 가정하고 그 객체의 분류를 찾는 문제이고, 객체인식은 분류를 하기 전에 화면의 어느 위치에 어떤 객체가 있는지 없는지부터 찾는 문제이다. 객체가 있는 것이 확인되면 그 객체의 위치를 bounding box로 값을 구해낸다. - 문제 객체가 2개 이상 들어 있는 사진을 한 장 구하고 화면상에 들어 있는 객체를 포함하는 박스의 위치를 눈대중으로 구해 낸다. 프레임워크에서 제공해주는 API 함수를 이용하여 색을 구분하여 bounding box를 추가하여 화면에 표시한다. 박스의 위치가 정확하지 않으면 위치를 수정하고 다시 boudning box를 화면에 그린다.
작품 구조도 (문제점 제시 및 개선방안)	- 프레임 워크를 구성하는 Module, Data, Trainer 클래스의 API를 이해한다 `class Module(nn.Block, d2l.HyperParameters):  #@save` `    """The base class of models."""` `    def __init__(self, plot_train_per_epoch=2, plot_valid_per_epoch=1):` `        super().__init__()` `        self.save_hyperparameters()` `        self.board = ProgressBoard()` `    def loss(self, y_hat, y):` `        raise NotImplementedError` `    def forward(self, X):` `        assert hasattr(self, 'net'), 'Neural network is defined'` `        return self.net(X)` `class DataModule(d2l.HyperParameters):  #@save` `    """The base class of data."""` `    def __init__(self, root='../data', num_workers=4):` `        self.save_hyperparameters()` `    def get_dataloader(self, train):` `        raise NotImplementedError` `class Trainer(d2l.HyperParameters):  #@save` `    """The base class for training models with data."""` `    def __init__(self, max_epochs, num_gpus=0, gradient_clip_val=0):` `        self.save_hyperparameters()` `        assert num_gpus == 0, 'No GPU support yet'` `    def prepare_data(self, data):` `        self.train_dataloader = data.train_dataloader()` `        self.val_dataloader = data.val_dataloader()` `        self.num_train_batches = len(self.train_dataloader)` `        self.num_val_batches = (len(self.val_dataloader)` `                                if self.val_dataloader is not None else 0)`

CNN의 주요 개념들을 이해한다

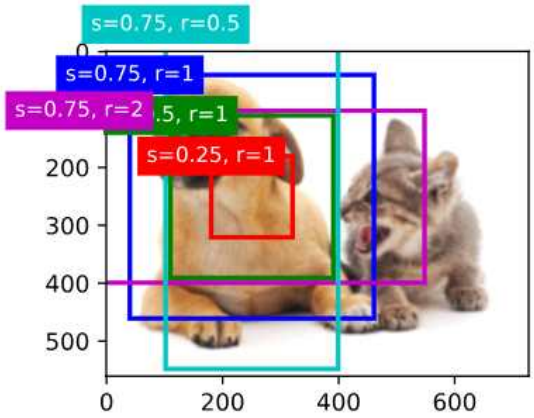


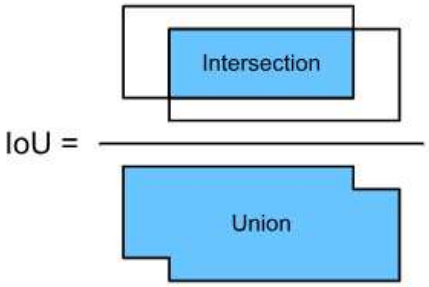
객체인식의 Bounding Box 사용법을 이해한다

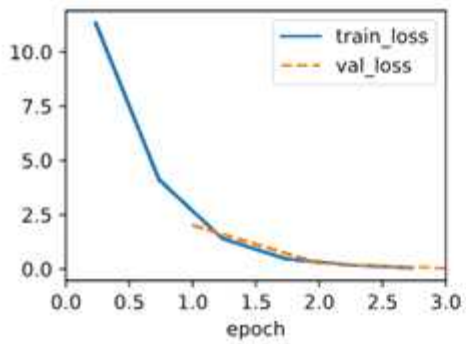


관련 이론	선형회귀에 대하여 이해한다.  분류에 대하여 이해한다.  히든층이 있는 MLP에 대하여 이해한다.  컴퓨터비젼의 CNN 모델에 대하여 이해한다. 

객체 인식 모델에 대하여 이해를 한다.





결과물 제작 (문제점 개선사항)	========== [1. 선형회귀 문제에서 학습한 모델의 패러미터 w, b 확인하기] URL : https://d2l.ai/chapter_linear-regression/linear-regression-scratch.html 3장 Linear Neural Network for Regression 3-4. Linear Regression Implementation from Scratch `model = LinearRegressionScratch(2, lr=0.03)` `data = d2l.SyntheticRegressionData(w=np.array([2, -3.4]), b=4.2)` `trainer = d2l.Trainer(max_epochs=3)` `trainer.fit(model, data)` 예 : `print(f'error in estimating w: {data.w - model.w.reshape(data.w.shape)}')` `print(f'error in estimating b: {data.b - model.b}')` `error in estimating w: [ 0.11080897 -0.12691855]` `error in estimating b: [0.19214153]` ========== [결과] 위의 예와 같이 출력명령을 캡처하여 붙여 넣고, 오차가 얼마나 발생했는지에 대해 간단히 설명 한다.  `print(f'error in estimating w: {data.w - model.w.reshape(data.w.shape)}')` `print(f'error in estimating b: {data.b - model.b}')` `error in estimating w: [ 0.1143676  -0.12299228]` `error in estimating b: [0.18997431]` 1.선형 회귀 모델을 정의하고, 학습률을 0.03으로 지정한다. 2.가상데이터셋으로 w=[2, -3.4]와 b=4.2인 모델의 패러미터를 생성한다. 3. fit( )함수를 이용해 모델을 학습시키고, fit함수를 통해 3번의 주기로 훈련을 진행함.

4. 학습이 완료된 후, 추정된 예측데이터 model.w , model.b들과 실제 데이터의 값을 비교한다.

그렇게 해서 나온 w의 오차는 w=[2, -3.4]에서 예측된 값이 약 0.11, -0.12만큼 차이가 나며, b의 오차는 b=4.2에서 예측된 값이 약 0.19만큼 차이가 나고 있다.

전체적으로 w[가중치]와 b 모두 정확하게 추정되었으며 오차는 작은값이 나왔으므로 학습이 잘 되었다고 볼 수 있다.

==========

[2. 프레임워크에서 제공하는 클래스의 API에 대해 이해하기]

URL : https://d2l.ai/chapter_linear-regression/index.html
3. Linear Neural Networks for Regression

==========

[결과]

1) 모델 클래스가 제공해주는 주요 함수의 API에 대해서 입력, 동작, 출력에 대하여 기술하시오

① init( ) 함수
- 입력 : 인자로 훈련과 검증데이터를 plot할 주기를 입력받는다.
- 동작 : 상속받은 클래스의 초기화를 수행하고 save_hyperparameters를 통해 넘어온 인자값들을 객체 내에 저장함.
- 출력 : 출력 값은 없다.

② loss( ) 함수
- 입력 : 예측값인 y_hat과 실제값인 y를 입력받는다.
- 동작 : 모델의 예측값과 실제값의 손실을 계산함.
- 출력 : 출력 값은 계산된 손실값이다.

③ forward( ) 함수
- 입력 : 모델에 입력할 데이터 X를 입력받음.
- 동작 : 입력데이터를 통해 전방전파 연산을 수행함
- 출력 : 입력값을 통해 전방전파 연산을 수행하여 나온 예측값

④ training_step( ), validation_step( ) 함수
- 입력 : batch 데이터를 입력받음
- 동작 : 입력 데이터를 forward( ) 함수에 전달해 예측값을 계산함. 예측값과 실제값을 통해 loss( )함수에 전달해 손실값을 계산함.
- 출력 : 손실값을 출력함.

2) 데이터 클래스가 제공해주는 주요 함수의 API에 대해서 입력, 동작, 출력에 대하여 기술하시오

① init( ) 함수 [syntheticRegressionData 클래스]

- 입력 : w, b, noise, num_train, num_val, batch_size 모델의 패러미터와 노이즈값, 훈련/검증 데이터의 개수, 배치데이터의 크기를 인자로 받는다.
- 동작 : d2l.DataModule 클래스를 상속받고, 하이퍼 패러미터를 이용해 변수들을 저장하고 총 샘플의 개수[num_train + num_val = 2000]를 n에 저장함, 그리고 self.X는 n개의 총 샘플과 len(w)의 크기 (2000,2)의 크기로 랜덤하게 생성함. Noise는 (2000, 1)의 크기로 랜덤하게 생성함. 마지막으로 self.Y는 y=Xw + b + noise라는 수식에 따라 계산.
- 출력 : 출력 값은 없다.

① init( )함수

- 입력 : 데이터가 저장될 기본 디렉터리 경로를'../data'로 지정하고, 데이터 로딩에 사용할 작업자의 수 'num_workers'를 인자로 받는다.
- 동작 : 입력받은 하이퍼 파라미터를 저장한다.
- 출력 : 출력값 없음[생성자 함수]

② get_dataloader( )함수

- 입력 : 데이터를 받을 때 훈련데이터인지 아닌지를 알기 위해 train이라는 인자를 받는다.
- 동작 : indicies 리스트를 만들고, 훈련데이터이면 셔플을 적용하며, 훈련데이터가 아닌[검증데이터] 경우 순차적인 순서를 사용한다. 미니배치의 크기만큼 batch_indices를 만듬. batch_indices를 이용하여 미니배치 크기의 X리스트와 Y리스트를 구한다.
- 출력 : yield 함수를 이용해 for루프를 돌며 연속된 묶음데이터를 리턴한다.

③ train_dataloader( )함수

- 입력 : 입력값 없음
- 동작 : "train=True" 패러미터를 전달하여 학습 데이터를 가져온다.
- 출력 : 학습 데이터로더 객체를 반환함

④ val_dataloader( )함수

- 입력 : 입력값 없음
- 동작 : "train=False" 패러미터를 전달하여 검증 데이터를 가져온다.
- 출력 : 학습 데이터로더 객체를 반환함

3) 트레이너 클래스가 제공해주는 주요 함수의 API에 대해서 입력, 동작, 출력에 대하여 기술하시오

① init( )함수

- 입력 : max_epoch, num_gpus, gradient_clip_val을 인자로 입력받는다.
- 동작 : 하이퍼 패러미터를 저장하고, gpu를 사용하지 않는다는 것을 명시
- 출력 : 출력값 없음

② prepare_data( )함수

- 입력 : train_dataloader(), val_dataloader()를 가진 데이터셋 객체를 입력

	받음. • 동작 : 훈련, 검증에 사용할 데이터 로더를 설정하고, 각 데이터 로더의 배치 수를 계산하고 저장함. • 출력 : 출력값 없음. ③ prepare_model( )함수 • 입력 : 훈련할 model 객체를 입력받는다. • 동작 : 모델의 trainer속성에 현재 Trainer객체를 할당하고, 모델 객체를 인스턴스 변수로 저장한다. • 출력 : 출력값 없음 ④ fit( )함수 • 입력 : model, data을 준비한다. • 동작 : 입력받은 model과 data를 준비하고, 최적화할 sgd도 준비한다. 그리고 fit_epoch()를 호출하여 epoch의 횟수만큼 학습을 시작한다. • 출력 : 출력값 없음 ⑤ fit_epoch( )함수 • 입력 : 없음 • 동작 : 미니배치 단위로 수행하고 training_step() 함수를 실행하여 전방전파 계산을 진행하고 손실함수를 리턴한다. 그리고 loss.backward()함수를 실행하여 손실함수를 가지고 후방전파 계산을 진행해 패러미터의 미분을 계산하고 step함수를 호출해 SGD최적화 기법을 이용해 패러미터를 업데이트한다. • 출력 : 출력값 없음

==========

[3. 분류 문제에서 one-hot encoding, softmax argmax계산하기]

URL : https://d2l.ai/chapter_linear-classification/softmax-regression.html

4장 Linear Neural Network for Classification

4.1. Softmax Regression

==========

[결과]

계산시 계산식도 표기한다.

1) 2에 해당하는 One-hot Encoding 구하기

▶ [0, 1, 0]

2) [-0.4, 4.2, 0.3]에 해당하는 softmax 값 구하기. 이때 softmax 계산식을 구할 때 공학용 계산기를 이용하여 log 값을 구해야 됨

Y(데이터)	3		
OnehotEnc	(0, 1, 0)		
o1, o2, o3	-0.4	4.2	0.3
softmax	0.01	0.97	0.02
argmax	2		
SoftmaxCrossEntrypy	-1*(0*log(0.01) + 1*log(0.97) + 0*log(0.02)) = 0.01		

▶softmax 계산식은 다음과 같다.

$$\sigma(x_j) = \frac{e^{x_j}}{\sum_i e^{x_i}}$$

이를 보기 쉽게 풀이하면

o1의 softmax = log(0.4)/log(0.4)+log(4.2)+log(0.3)을 계산

o2의 softmax = log(4.2)/log(0.4)+log(4.2)+log(0.3)을 계산

o3의 softmax = log(0.3)/log(0.4)+log(4.2)+log(0.3)을 계산

3) 2)번의 softmax 값을 입력으로 받아 argmax 값 계산하기

▶softmax에서 가장 큰값은 0.97이며 이는 인덱스 2에 해당하므로, argmax = 2이다.

4) 참의값 1)에서 구한 One-hot Encoding값과 예측값 2)에서 구한 softmax 값을 이용하여 SoftmaxCrossEntropy 계산하기. 공학용 계산기를 이용하여 log 값을 구해야 됨

-1*(0*log(0.01) + 1*log(0.97) + 0*log(0.02)) = 0.01

표의 식처럼 계산하여 나온 결과는 0.01이 된다.

==========

[4. CNN 모델에서 Padding, Striding 개념 이해하기]

URL : https://d2l.ai/chapter_convolutional-neural-networks/index.html

7. Convolutional Neural Networks

==========

[결과]

원본 사진의 크기는 320*240 이고 kernel window의 크기는 5*5 인 경우에

1) padding = 0, stride = 1 인 경우에 출력 결과의 크기를 계산하시오

CNN에서 출력 크기를 계산하는 방법은 다음식과 같이 진행되며,

nh/nw = 원본사진의 높이와 너비, kh/hw = kernal window의 크기, ph/pw = 패딩의 크기, sh/sw = 스트라이드의 크기이다.

$$\lfloor (n_{\mathrm{h}} - k_{\mathrm{h}} + p_{\mathrm{h}} + s_{\mathrm{h}})/s_{\mathrm{h}} \rfloor \times \lfloor (n_{\mathrm{w}} - k_{\mathrm{w}} + p_{\mathrm{w}} + s_{\mathrm{w}})/s_{\mathrm{w}} \rfloor.$$

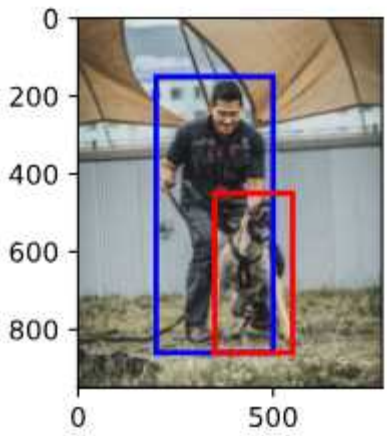
해당 식에 대입하면 [(320-5+0+1)/1] × [(240-5+0+1)/1]이 되며

출력 결과의 크기는 316×236의 크기를 갖는다.

2) padding = 1, stride = 1 인 경우에 출력 결과의 크기를 계산하시오

마찬가지로 식에 대입하면 [(320-5+1+1)/1] × [(240-5+1+1)/1]이며

출력 결과의 크기는 318×238의 크기를 갖는다.

	========== [5. 객체인식 문제에서 화면에 Bounding Box 추가하기] URL : https://d2l.ai/chapter_computer-vision/bounding-box.html 14.3. Object Detection and Bounding Boxes ========== [결과] bounding box가 그려진 화면을 캡처하여 추가 하시오 ▶사람과 개가 같이 있는 사진에서 사람과 개를 구분하기 위해 다음과 같이 x,y값을 지정하여 bounding box를 그림 `# Here `bbox` is the abbreviation for bounding box` `people_bbox, dog_bbox = [200, 150, 500, 860], [350.0, 450.0, 550.0, 860.0]` 
기대 효과 및 활용방안	기계학습은 일반적으로 프레임워크를 이용하여 애플리케이션을 구현하는데 사용되기 때문에 프레임워크를 제대로 사용하는 방법을 배운다. 기능의 기본원리를 이해하는 것도 중요하며 최종적으로는 애플리케이션으로 구현 할 수 있는 능력을 키우는 것이 필요하다. 다양한 실습의 구현을 통해서 인공지능 기능 구현에 활용 가능하다.
2024 년 11 월 27 일 **지도교수 심 효 선 (인)**	