7. Convolutional Neural Networks(CNN)

- Ideally, we would leverage our prior knowledge that **nearby pixels are typically related to each other**, to build efficient models for learning from image data.
- CNNs tend to be **computationally efficient**: -**require fewer parameters** than fully connected architectures
  - **convolutions are easy to parallelize** across GPU cores(컨볼루션 연산 - 여러 픽셀을 독립적으로 처리할 수 있어 병렬화가 용이, GPU - 다수의 코어로 이러한 병렬 연산을 동시 수행하는 데 적합.)

7.1. From Fully Connected Layers to Convolutions

7.1.1. Invariance (불변성)

- **컴퓨터 비전 Neural Network Architecture 설계 원칙**
  1. **translation invariance(equivalance)** - 네트워크의 초기 층(earliest layers)은 이미지 내에서 동일한 패치에 대해 위치에 상관없이 유사하게 반응해야함
     - *(예) Pigs usually do not fly and planes usually do not swim. Nonetheless, we should still recognize a pig were one to appear at the top of the image.*
  2. **locality principl**e - earliest layers는 이미지의 먼 영역보다는 local regions에 집중해야 함. --> 이후에 이 local representations들을 이용하여 whole image level의 predictions를 예측.
  3. deeper layers로 갈수록, longer-range의 이미지 features을 포착할 수 있어야 함.

7.1.2. Constraining the MLP

- input images(2D) X와 hidden representations
  - 2D 텐서들로 표현(matrices임)
  - same shape
  - 모두 공간적 구조를 가짐.
  - input 이미지의 각 위치에 있는 픽셀 값이 hidden representations의 각 유닛에 영향을 미침 --> 4차원 가중치 텐서 W를 사용하여 표현.
  - re-index the subscripts such that k=i+a and l=j+b
  - ***a image (1 megapixel) is mapped to a hidden representation. This requires parameters, far beyond what computers currently can handle.

(principle 1) Translation Invariance

- input X에서의 shift는 hidden representation H에서도 같은 이동을 일으킴(단, V와 8 is not depend on (i,j))
- ==> convolution 연산 !
- 은닉 표현의 각 위치에서 주변 픽셀의 값을 가중치로 합산하여 출력이 결정된다.
- 이는 바로 컨볼루션 연산이며, 가중치 텐서의 매개변수 수를 크게 줄일 수 있다.
- 매개변수 수는 이제 입력 크기에 비례 X, 필터 크기에만 비례.

(principle 2) Locality

- hidden representation [H]_i,j 의 특정 위치에서 중요한 정보는 해당 위치 주변의 local 영역에서만 나온다고 가정.즉, 특정 거리 이상 떨어진 위치의 정보는 해당 위치에 영향 X
  - 이를 통해 가중치를 설정하는 범위를 제한.
- convolution 연산에서 필터의 크기를 제한하여 가중치 수를 더 줄일 수 있다.
  - 매개변수 수: 더 감소.

7.1.3. Convolutions

CNN

- convolution 레이어:
  - Rather than using (i+a, j+b), we are using the difference instead
  - cross-correlation
- 더 깊은 층에서는 점차 더 큰 영역의 정보를 처리하면서, 비선형성을 추가하여 복잡한 이미지 구조를 학습.

7.1.4. Channels

우리의 목표

- convolutional layer picks windows of a given size and weighs intensities according to the filter V
- learn a model:
    - “feature” is "highest"인 곳 == "peak"를 찾는 것이 목표임
    - in the hidden layer representations.
- 이미지 고려할 때, width와 height뿐만 아니라 RGB값이라는 channel도 있음을 고려한다면 --> 3차원 텐서를 써야됨! == feature map : each provides a spatialized set of learned features for the subsequent layer.
- -> 쓸모: specialized to recognize 'edges'인 것 / 'texture'인 것 등등 다 다른 특성.

느낀 점 & Exercises Discussion:

- 수학 표현을 복습 및 확실히 이해하도록 공부 복습 필요.
- 이미지에서 컨볼루션을 수행할 때 객체가 경계에 위치하면, 어떻게 될까?
    - 일부 픽셀이 커널의 영역 밖으로 나가기 때문에 정보가 손실될 수 있음. --> 해결책으로 패딩(padding)을 사용 가능.

7.2. Convolutions for Images

```
!pip install d2l
```

```
Collecting d2l
  Downloading d2l-1.0.3-py3-none-any.whl.metadata (556 bytes)
Collecting jupyter==1.0.0 (from d2l)
  Downloading jupyter-1.0.0-py2.py3-none-any.whl.metadata (995 bytes)
Collecting numpy==1.23.5 (from d2l)
  Downloading numpy-1.23.5-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (2.3 kB)
Collecting matplotlib==3.7.2 (from d2l)
  Downloading matplotlib-3.7.2-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (5.6 kB)
Collecting matplotlib-inline==0.1.6 (from d2l)
  Downloading matplotlib_inline-0.1.6-py3-none-any.whl.metadata (2.8 kB)
Collecting requests==2.31.0 (from d2l)
  Downloading requests-2.31.0-py3-none-any.whl.metadata (4.6 kB)
Collecting pandas==2.0.3 (from d2l)
  Downloading pandas-2.0.3-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (18 kB)
Collecting scipy==1.10.1 (from d2l)
  Downloading scipy-1.10.1-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (58 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 58.9/58.9 kB 4.4 MB/s eta 0:00:00
Requirement already satisfied: notebook in /usr/local/lib/python3.10/dist-packages (from jupyter==1.0.0->d2l) (6.5.5)
Collecting qtconsole (from jupyter==1.0.0->d2l)
  Downloading qtconsole-5.6.0-py3-none-any.whl.metadata (5.0 kB)
Requirement already satisfied: jupyter-console in /usr/local/lib/python3.10/dist-packages (from jupyter==1.0.0->d2l) (6.
Requirement already satisfied: nbconvert in /usr/local/lib/python3.10/dist-packages (from jupyter==1.0.0->d2l) (6.5.4)
Requirement already satisfied: ipykernel in /usr/local/lib/python3.10/dist-packages (from jupyter==1.0.0->d2l) (5.5.6)
Requirement already satisfied: ipywidgets in /usr/local/lib/python3.10/dist-packages (from jupyter==1.0.0->d2l) (7.7.1)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib==3.7.2->d2l)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-packages (from matplotlib==3.7.2->d2l) (0.
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib==3.7.2->d2l
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib==3.7.2->d2l
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib==3.7.2->d2l)
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib==3.7.2->d2l) (1
Collecting pyparsing<3.1,>=2.3.1 (from matplotlib==3.7.2->d2l)
  Downloading pyparsing-3.0.9-py3-none-any.whl.metadata (4.2 kB)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.10/dist-packages (from matplotlib==3.7.2->
Requirement already satisfied: traitlets in /usr/local/lib/python3.10/dist-packages (from matplotlib-inline==0.1.6->d2l)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas==2.0.3->d2l) (2024.2
Requirement already satisfied: tzdata>=2022.1 in /usr/local/lib/python3.10/dist-packages (from pandas==2.0.3->d2l) (2024
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests==2.31.
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests==2.31.0->d2l) (3.1
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests==2.31.0->d2l
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests==2.31.0->d2l
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.7->matplotli
Requirement already satisfied: ipython-genutils in /usr/local/lib/python3.10/dist-packages (from ipykernel->jupyter==1.0
Requirement already satisfied: ipython>=5.0.0 in /usr/local/lib/python3.10/dist-packages (from ipykernel->jupyter==1.0.0
Requirement already satisfied: jupyter-client in /usr/local/lib/python3.10/dist-packages (from ipykernel->jupyter==1.0.0
Requirement already satisfied: tornado>=4.2 in /usr/local/lib/python3.10/dist-packages (from ipykernel->jupyter==1.0.0->
Requirement already satisfied: widgetsnbextension~=3.6.0 in /usr/local/lib/python3.10/dist-packages (from ipywidgets->ju
Requirement already satisfied: jupyterlab-widgets>=1.0.0 in /usr/local/lib/python3.10/dist-packages (from ipywidgets->ju
Requirement already satisfied: prompt-toolkit!=3.0.0,!=3.0.1,<3.1.0,>=2.0.0 in /usr/local/lib/python3.10/dist-packages (
Requirement already satisfied: pygments in /usr/local/lib/python3.10/dist-packages (from jupyter-console->jupyter==1.0.0
Requirement already satisfied: lxml in /usr/local/lib/python3.10/dist-packages (from nbconvert->jupyter==1.0.0->d2l) (4.
Requirement already satisfied: beautifulsoup4 in /usr/local/lib/python3.10/dist-packages (from nbconvert->jupyter==1.0.0
Requirement already satisfied: bleach in /usr/local/lib/python3.10/dist-packages (from nbconvert->jupyter==1.0.0->d2l) (
Requirement already satisfied: defusedxml in /usr/local/lib/python3.10/dist-packages (from nbconvert->jupyter==1.0.0->d2
Requirement already satisfied: entrypoints>=0.2.2 in /usr/local/lib/python3.10/dist-packages (from nbconvert->jupyter==1
Requirement already satisfied: jinja2>=3.0 in /usr/local/lib/python3.10/dist-packages (from nbconvert->jupyter==1.0.0->d
Requirement already satisfied: jupyter-core>=4.7 in /usr/local/lib/python3.10/dist-packages (from nbconvert->jupyter==1.
Requirement already satisfied: jupyterlab-pygments in /usr/local/lib/python3.10/dist-packages (from nbconvert->jupyter==
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.10/dist-packages (from nbconvert->jupyter==1.0.
Requirement already satisfied: mistune<2,>=0.8.1 in /usr/local/lib/python3.10/dist-packages (from nbconvert->jupyter==1.
Requirement already satisfied: nbclient>=0.5.0 in /usr/local/lib/python3.10/dist-packages (from nbconvert->jupyter==1.0.
Requirement already satisfied: nbformat>=5.1 in /usr/local/lib/python3.10/dist-packages (from nbconvert->jupyter==1.0.0-
Requirement already satisfied: pandocfilters>=1.4.1 in /usr/local/lib/python3.10/dist-packages (from nbconvert->jupyter=
Requirement already satisfied: tinycss2 in /usr/local/lib/python3.10/dist-packages (from nbconvert->jupyter==1.0.0->d2l)
Requirement already satisfied: pyzmq<25,>=17 in /usr/local/lib/python3.10/dist-packages (from notebook->jupyter==1.0.0->
Requirement already satisfied: argon2-cffi in /usr/local/lib/python3.10/dist-packages (from notebook->jupyter==1.0.0->d2
Requirement already satisfied: nest-asyncio>=1.5 in /usr/local/lib/python3.10/dist-packages (from notebook->jupyter==1.0
Requirement already satisfied: Send2Trash>=1.8.0 in /usr/local/lib/python3.10/dist-packages (from notebook->jupyter==1.0
Requirement already satisfied: terminado>=0.8.3 in /usr/local/lib/python3.10/dist-packages (from notebook->jupyter==1.0.
Requirement already satisfied: prometheus-client in /usr/local/lib/python3.10/dist-packages (from notebook->jupyter==1.0
Requirement already satisfied: nbclassic>=0.4.7 in /usr/local/lib/python3.10/dist-packages (from notebook->jupyter==1.0.
Collecting qtpy>=2.4.0 (from qtconsole->jupyter==1.0.0->d2l)
  Downloading QtPy-2.4.1-py3-none-any.whl.metadata (12 kB)
Requirement already satisfied: setuptools>=18.5 in /usr/local/lib/python3.10/dist-packages (from ipython>=5.0.0->ipykern
Collecting jedi>=0.16 (from ipython>=5.0.0->ipykernel->jupyter==1.0.0->d2l)
  Using cached jedi-0.19.1-py2.py3-none-any.whl.metadata (22 kB)
Requirement already satisfied: decorator in /usr/local/lib/python3.10/dist-packages (from ipython>=5.0.0->ipykernel->jup
Requirement already satisfied: pickleshare in /usr/local/lib/python3.10/dist-packages (from ipython>=5.0.0->ipykernel->j
Requirement already satisfied: backcall in /usr/local/lib/python3.10/dist-packages (from ipython>=5.0.0->ipykernel->jupy
Requirement already satisfied: pexpect>4.3 in /usr/local/lib/python3.10/dist-packages (from ipython>=5.0.0->ipykernel->j
Requirement already satisfied: platformdirs>=2.5 in /usr/local/lib/python3.10/dist-packages (from jupyter-core>=4.7->nbc
Requirement already satisfied: notebook-shim>=0.2.3 in /usr/local/lib/python3.10/dist-packages (from nbclassic>=0.4.7->n
Requirement already satisfied: fastjsonschema>=2.15 in /usr/local/lib/python3.10/dist-packages (from nbformat>=5.1->nbco
Requirement already satisfied: jsonschema>=2.6 in /usr/local/lib/python3.10/dist-packages (from nbformat>=5.1->nbconvert
Requirement already satisfied: wcwidth in /usr/local/lib/python3.10/dist-packages (from prompt-toolkit!=3.0.0,!=3.0.1,<3
Requirement already satisfied: ptyprocess in /usr/local/lib/python3.10/dist-packages (from terminado>=0.8.3->notebook->j
Requirement already satisfied: argon2-cffi-bindings in /usr/local/lib/python3.10/dist-packages (from argon2-cffi->notebo
Requirement already satisfied: soupsieve>1.2 in /usr/local/lib/python3.10/dist-packages (from beautifulsoup4->nbconvert-
Requirement already satisfied: webencodings in /usr/local/lib/python3.10/dist-packages (from bleach->nbconvert->jupyter=
Requirement already satisfied: parso<0.9.0,>=0.8.3 in /usr/local/lib/python3.10/dist-packages (from jedi>=0.16->ipython>
Requirement already satisfied: attrs>=22.2.0 in /usr/local/lib/python3.10/dist-packages (from jsonschema>=2.6->nbformat>
```

```
Requirement already satisfied: jsonschema-specifications>=2023.03.6 in /usr/local/lib/python3.10/dist-packages (from jso
Requirement already satisfied: referencing>=0.28.4 in /usr/local/lib/python3.10/dist-packages (from jsonschema>=2.6->nbf
Requirement already satisfied: rpds-py>=0.7.1 in /usr/local/lib/python3.10/dist-packages (from jsonschema>=2.6->nbformat
Requirement already satisfied: jupyter-server<3,>=1.8 in /usr/local/lib/python3.10/dist-packages (from notebook-shim>=0.
Requirement already satisfied: cffi>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from argon2-cffi-bindings->argon2
Requirement already satisfied: pycparser in /usr/local/lib/python3.10/dist-packages (from cffi>=1.0.1->argon2-cffi-bindi
Requirement already satisfied: anyio<4,>=3.1.0 in /usr/local/lib/python3.10/dist-packages (from jupyter-server<3,>=1.8->
Requirement already satisfied: websocket-client in /usr/local/lib/python3.10/dist-packages (from jupyter-server<3,>=1.8-
Requirement already satisfied: sniffio>=1.1 in /usr/local/lib/python3.10/dist-packages (from anyio<4,>=3.1.0->jupyter-se
Requirement already satisfied: exceptiongroup in /usr/local/lib/python3.10/dist-packages (from anyio<4,>=3.1.0->jupyter-
Downloading d2l-1.0.3-py3-none-any.whl (111 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 111.7/111.7 kB 4.5 MB/s eta 0:00:00
Downloading jupyter-1.0.0-py2.py3-none-any.whl (2.7 kB)
Downloading matplotlib-3.7.2-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (11.6 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 11.6/11.6 MB 100.8 MB/s eta 0:00:00
Downloading matplotlib_inline-0.1.6-py3-none-any.whl (9.4 kB)
Downloading numpy-1.23.5-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (17.1 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 17.1/17.1 MB 100.1 MB/s eta 0:00:00
Downloading pandas-2.0.3-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (12.3 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 12.3/12.3 MB 76.4 MB/s eta 0:00:00
Downloading requests-2.31.0-py3-none-any.whl (62 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 62.6/62.6 kB 5.6 MB/s eta 0:00:00
Downloading scipy-1.10.1-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (34.4 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 34.4/34.4 MB 19.8 MB/s eta 0:00:00
Downloading pyparsing-3.0.9-py3-none-any.whl (98 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 98.3/98.3 kB 8.4 MB/s eta 0:00:00
Downloading qtconsole-5.6.0-py3-none-any.whl (124 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 124.7/124.7 kB 11.1 MB/s eta 0:00:00
Downloading QtPy-2.4.1-py3-none-any.whl (93 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 93.5/93.5 kB 8.3 MB/s eta 0:00:00
Using cached jedi-0.19.1-py2.py3-none-any.whl (1.6 MB)
Installing collected packages: requests, qtpy, pyparsing, numpy, matplotlib-inline, jedi, scipy, pandas, matplotlib, qtc
  Attempting uninstall: requests
    Found existing installation: requests 2.32.3
    Uninstalling requests-2.32.3:
      Successfully uninstalled requests-2.32.3
  Attempting uninstall: pyparsing
    Found existing installation: pyparsing 3.1.4
    Uninstalling pyparsing-3.1.4:
      Successfully uninstalled pyparsing-3.1.4
  Attempting uninstall: numpy
    Found existing installation: numpy 1.26.4
    Uninstalling numpy-1.26.4:
      Successfully uninstalled numpy-1.26.4
  Attempting uninstall: matplotlib-inline
    Found existing installation: matplotlib-inline 0.1.7
    Uninstalling matplotlib-inline-0.1.7:
      Successfully uninstalled matplotlib-inline-0.1.7
  Attempting uninstall: scipy
    Found existing installation: scipy 1.13.1
    Uninstalling scipy-1.13.1:
      Successfully uninstalled scipy-1.13.1
  Attempting uninstall: pandas
    Found existing installation: pandas 2.2.2
    Uninstalling pandas-2.2.2:
      Successfully uninstalled pandas-2.2.2
  Attempting uninstall: matplotlib
    Found existing installation: matplotlib 3.7.1
    Uninstalling matplotlib-3.7.1:
      Successfully uninstalled matplotlib-3.7.1
ERROR: pip's dependency resolver does not currently take into account all the packages that are installed. This behaviou
albucore 0.0.16 requires numpy>=1.24, but you have numpy 1.23.5 which is incompatible.
albumentations 1.4.15 requires numpy>=1.24.4, but you have numpy 1.23.5 which is incompatible.
bigframes 1.21.0 requires numpy>=1.24.0, but you have numpy 1.23.5 which is incompatible.
chex 0.1.87 requires numpy>=1.24.1, but you have numpy 1.23.5 which is incompatible.
google-colab 1.0.0 requires pandas==2.2.2, but you have pandas 2.0.3 which is incompatible.
google-colab 1.0.0 requires requests==2.32.3, but you have requests 2.31.0 which is incompatible.
jax 0.4.33 requires numpy>=1.24, but you have numpy 1.23.5 which is incompatible.
jaxlib 0.4.33 requires numpy>=1.24, but you have numpy 1.23.5 which is incompatible.
mizani 0.11.4 requires pandas>=2.1.0, but you have pandas 2.0.3 which is incompatible.
plotnine 0.13.6 requires pandas<3.0.0,>=2.1.0, but you have pandas 2.0.3 which is incompatible.
xarray 2024.9.0 requires numpy>=1.24, but you have numpy 1.23.5 which is incompatible.
xarray 2024.9.0 requires pandas>=2.1, but you have pandas 2.0.3 which is incompatible.
Successfully installed d2l-1.0.3 jedi-0.19.1 jupyter-1.0.0 matplotlib-3.7.2 matplotlib-inline-0.1.6 numpy-1.23.5 pandas-
```

```
import torch
from torch import nn
from d2l import torch as d2l
```

7.2.1. The Cross-Correlation Operation

- 합성곱 윈도우가 특정 위치로 슬라이드하면 --> 해당 윈도우에 포함된 입력 서브텐서와 커널 텐서가 요소별로 곱해지고 --> 결과 텐서가 합산되어 단일 스칼라 값이 생성

```
def corr2d(X, K):
    """Compute 2D cross-correlation."""
    h, w = K.shape
    Y = torch.zeros((X.shape[0] - h + 1, X.shape[1] - w + 1))
    for i in range(Y.shape[0]):
        for j in range(Y.shape[1]):
            Y[i, j] = (X[i:i + h, j:j + w] * K).sum()
    return Y
```

```
X = torch.tensor([[0.0, 1.0, 2.0], [3.0, 4.0, 5.0], [6.0, 7.0, 8.0]])
K = torch.tensor([[0.0, 1.0], [2.0, 3.0]])
corr2d(X, K)
```

```
tensor([[19., 25.],
        [37., 43.]])
```

7.2.2. 합성곱 계층

- cross-correlates the input&kernel
- add a scalar bias
- -> produce an output.
- convolutional layer의 2개 파라미터: kernel & scalar bias

```
#2차원 합성곱 계층 구현

class Conv2D(nn.Module):
    def __init__(self, kernel_size):
        super().__init__()
        self.weight = nn.Parameter(torch.rand(kernel_size))
        self.bias = nn.Parameter(torch.zeros(1))

    def forward(self, x):
        return corr2d(x, self.weight) + self.bias
```

7.2.3. Object Edge Detection - in Images

- 픽셀 변경 위치를 찾아 이미지에서 객체의 모서리를 감지하는 것

```
# 1. '이미지' 구성
X = torch.ones((6, 8))
X[:, 2:6] = 0
X
```

```
tensor([[1., 1., 0., 0., 0., 0., 1., 1.],
        [1., 1., 0., 0., 0., 0., 1., 1.],
        [1., 1., 0., 0., 0., 0., 1., 1.],
        [1., 1., 0., 0., 0., 0., 1., 1.],
        [1., 1., 0., 0., 0., 0., 1., 1.],
        [1., 1., 0., 0., 0., 0., 1., 1.]])
```

```
# kernel: (height = 1, width = 2)
K = torch.tensor([[1.0, -1.0]])
```

```
Y = corr2d(X, K)
Y
```

```
tensor([[ 0.,  1.,  0.,  0.,  0., -1.,  0.],
        [ 0.,  1.,  0.,  0.,  0., -1.,  0.],
        [ 0.,  1.,  0.,  0.,  0., -1.,  0.],
        [ 0.,  1.,  0.,  0.,  0., -1.,  0.],
        [ 0.,  1.,  0.,  0.,  0., -1.,  0.],
        [ 0.,  1.,  0.,  0.,  0., -1.,  0.]])
```

```
# 확인 - apply the kernel to the transposed image. As expected, it vanishes. The kernel K only detects vertical edges.
corr2d(d2l.transpose(X), K)
```

```
tensor([[0., 0., 0., 0., 0.],
        [0., 0., 0., 0., 0.],
        [0., 0., 0., 0., 0.],
        [0., 0., 0., 0., 0.],
        [0., 0., 0., 0., 0.],
        [0., 0., 0., 0., 0.],
        [0., 0., 0., 0., 0.],
        [0., 0., 0., 0., 0.]])
```

7.2.4. Learning a Kernel

- as we look at larger kernels, and consider successive layers of convolutions, it might be impossible to specify precisely what each filter should be doing manually.(더 큰 커널을 살펴보고 연속적인 합성곱 층을 고려할 때, 각각의 필터가 정확히 무엇을 해야 하는지 수동으로 지정하는 것은 불가능할 수 있음.)
- whether we can learn the kernel that generated Y from X by looking at the input-output pairs only.(입력-출력 쌍만을 보고 X에서 Y를 생성한 커널을 학습할 수 있는지 여부.)

```
pip install flax==0.6.11 jax==0.4.11 jaxlib==0.4.11
```

```
Requirement already satisfied: flax==0.6.11 in /usr/local/lib/python3.10/dist-packages (0.6.11)
Requirement already satisfied: jax==0.4.11 in /usr/local/lib/python3.10/dist-packages (0.4.11)
Requirement already satisfied: jaxlib==0.4.11 in /usr/local/lib/python3.10/dist-packages (0.4.11)
Requirement already satisfied: numpy>=1.12 in /usr/local/lib/python3.10/dist-packages (from flax==0.6.11) (1.23.5)
Requirement already satisfied: msgpack in /usr/local/lib/python3.10/dist-packages (from flax==0.6.11) (1.0.8)
Requirement already satisfied: optax in /usr/local/lib/python3.10/dist-packages (from flax==0.6.11) (0.2.1)
Requirement already satisfied: orbax-checkpoint in /usr/local/lib/python3.10/dist-packages (from flax==0.6.11) (0.5.16)
Requirement already satisfied: tensorstore in /usr/local/lib/python3.10/dist-packages (from flax==0.6.11) (0.1.66)
Requirement already satisfied: rich>=11.1 in /usr/local/lib/python3.10/dist-packages (from flax==0.6.11) (13.9.1)
Requirement already satisfied: typing-extensions>=4.1.1 in /usr/local/lib/python3.10/dist-packages (from flax==0.6.11) (
Requirement already satisfied: PyYAML>=5.4.1 in /usr/local/lib/python3.10/dist-packages (from flax==0.6.11) (6.0.2)
Requirement already satisfied: ml-dtypes>=0.1.0 in /usr/local/lib/python3.10/dist-packages (from jax==0.4.11) (0.4.1)
Requirement already satisfied: opt-einsum in /usr/local/lib/python3.10/dist-packages (from jax==0.4.11) (3.4.0)
Requirement already satisfied: scipy>=1.7 in /usr/local/lib/python3.10/dist-packages (from jax==0.4.11) (1.10.1)
Requirement already satisfied: markdown-it-py>=2.2.0 in /usr/local/lib/python3.10/dist-packages (from rich>=11.1->flax==
Requirement already satisfied: pygments<3.0.0,>=2.13.0 in /usr/local/lib/python3.10/dist-packages (from rich>=11.1->flax
Requirement already satisfied: absl-py>=0.7.1 in /usr/local/lib/python3.10/dist-packages (from optax->flax==0.6.11) (1.4
Requirement already satisfied: chex>=0.1.7 in /usr/local/lib/python3.10/dist-packages (from optax->flax==0.6.11) (0.1.82
Requirement already satisfied: etils[epath,epy] in /usr/local/lib/python3.10/dist-packages (from orbax-checkpoint->flax=
Requirement already satisfied: nest_asyncio in /usr/local/lib/python3.10/dist-packages (from orbax-checkpoint->flax==0.6
Requirement already satisfied: protobuf in /usr/local/lib/python3.10/dist-packages (from orbax-checkpoint->flax==0.6.11)
Collecting numpy>=1.12 (from flax==0.6.11)
  Using cached numpy-1.26.4-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (61 kB)
Requirement already satisfied: toolz>=0.9.0 in /usr/local/lib/python3.10/dist-packages (from chex>=0.1.7->optax->flax==0
Requirement already satisfied: mdurl~=0.1 in /usr/local/lib/python3.10/dist-packages (from markdown-it-py>=2.2.0->rich>=
Requirement already satisfied: fsspec in /usr/local/lib/python3.10/dist-packages (from etils[epath,epy]->orbax-checkpoin
Requirement already satisfied: importlib_resources in /usr/local/lib/python3.10/dist-packages (from etils[epath,epy]->or
Requirement already satisfied: zipp in /usr/local/lib/python3.10/dist-packages (from etils[epath,epy]->orbax-checkpoint-
Using cached numpy-1.26.4-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (18.2 MB)
Installing collected packages: numpy
  Attempting uninstall: numpy
    Found existing installation: numpy 1.23.5
    Uninstalling numpy-1.23.5:
      Successfully uninstalled numpy-1.23.5
ERROR: pip's dependency resolver does not currently take into account all the packages that are installed. This behaviou
d2l 1.0.3 requires numpy==1.23.5, but you have numpy 1.26.4 which is incompatible.
mizani 0.11.4 requires pandas>=2.1.0, but you have pandas 2.0.3 which is incompatible.
plotnine 0.13.6 requires pandas<3.0.0,>=2.1.0, but you have pandas 2.0.3 which is incompatible.
xarray 2024.9.0 requires pandas>=2.1, but you have pandas 2.0.3 which is incompatible.
Successfully installed numpy-1.26.4
```

```
# Construct a two-dimensional convolutional layer with 1 output channel and a
# kernel of shape (1, 2). For the sake of simplicity, we ignore the bias here
conv2d = nn.LazyConv2d(1, kernel_size=(1, 2), bias=False)

# The two-dimensional convolutional layer uses four-dimensional input and
# output in the format of (example, channel, height, width), where the batch
# size (number of examples in the batch) and the number of channels are both 1
X = X.reshape((1, 1, 6, 8))
Y = Y.reshape((1, 1, 6, 7))
lr = 3e-2  # Learning rate

for i in range(10):
    Y_hat = conv2d(X)
    l = (Y_hat - Y) ** 2
    conv2d.zero_grad()
    l.sum().backward()
    # Update the kernel
    conv2d.weight.data[:] -= lr * conv2d.weight.grad
    if (i + 1) % 2 == 0:
        print(f'epoch {i + 1}, loss {l.sum():.3f}')
```

```
epoch 2, loss 10.382
epoch 4, loss 1.748
epoch 6, loss 0.296
epoch 8, loss 0.051
epoch 10, loss 0.009
```

```
conv2d.weight.data.reshape((1, 2)) #, the learned kernel tensor is remarkably close to the kernel tensor K we defined earlie
```

```
tensor([[ 0.9804, -0.9862]])
```

7.2.5. Cross-Correlation and Convolution

- "합성곱(convolution)"
  - 필터를 먼저 가로와 세로로 뒤집은 후 적용하는 과정
- "교차 상관(cross-correlation)"
  - 이미지에 필터(커널)를 그대로 적용하는 과정
- 딥러닝에서는 학습 과정에서 필터(커널)를 데이터로부터 배움 -> convolution이든 cross-correlation이든 동일한 출력 얻을 수 있음
- 즉 최종 결과에 큰 영향이 없으므로 크게 구분하지 않고 사용.

7.2.6. Feature Map and Receptive Field

- Receptive Field(수용 영역):
  - 신경망의 어떤 출력 요소가 얼마나 넓은 범위의 입력 데이터를 고려하는지를 나타냄
  - in other words, CNN의 한 출력 요소가 계산될 때, 그 요소는 이전 층의 여러 요소에 의해 영향을 받음. 이 영향을 주는 범위를 수용 영역이라고 하는 것.
  - deeper layer, larger receptive field
  - 예를 들어, 어떤 특정 요소가 이전 층의 4개의 요소로부터 영향을 받았다면, 그 4개의 요소가 수용 영역에 해당
  - 더 넓은 영역에서 입력 데이터를 분석하고 특징을 학습하는 데 중요한 역할
- feature map: CNN에서 합성곱을 통해 얻은 출력 -> input image부터 학습된 중요한 특징(feature)들을 공간적 차원(width & height)에서 나타냄. -> 이 피처 맵은 다음 층에서 사용됨.

7.3. Padding and Stride

- techniques that offer more control over the size of the output
- kernels가 일반적으로.. have width and height greater than 1, --> 여러 successive convolutions 적용 후의 output은 input보다 considerably smaller
  - 즉 경계에 있는 정보들이 지워짐
  - 이를 해결하기 위해 padding!
- 차원을 줄이고 싶은 경우(we may want to reduce the dimensionality drastically)
  - Strided convolutions!

```
import torch
from torch import nn
```

7.3.1. Padding

```
# We define a helper function to calculate convolutions. It initializes the
# convolutional layer weights and performs corresponding dimensionality
# elevations and reductions on the input and output
def comp_conv2d(conv2d, X):
    # (1, 1) indicates that batch size and the number of channels are both 1
    X = X.reshape((1, 1) + X.shape)
    Y = conv2d(X)
    # Strip the first two dimensions: examples and channels
    return Y.reshape(Y.shape[2:])

# 1 row and column is padded on either side, so a total of 2 rows or columns
# are added
conv2d = nn.LazyConv2d(1, kernel_size=3, padding=1)
X = torch.rand(size=(8, 8))
comp_conv2d(conv2d, X).shape
```

```
torch.Size([8, 8])
```

```
# We use a convolution kernel with height 5 and width 3. The padding on either
# side of the height and width are 2 and 1, respectively
```

```
conv2d = nn.LazyConv2d(1, kernel_size=(5, 3), padding=(2, 1))
comp_conv2d(conv2d, X).shape
```

```
torch.Size([8, 8])
```

7.3.2. Strides

```
conv2d = nn.LazyConv2d(1, kernel_size=3, padding=1, stride=2)
comp_conv2d(conv2d, X).shape
```

```
torch.Size([4, 4])
```

```
conv2d = nn.LazyConv2d(1, kernel_size=(3, 5), padding=(0, 1), stride=(3, 4))
comp_conv2d(conv2d, X).shape
```

```
torch.Size([2, 2])
```

Discussion

- 1보다 큰 보폭을 사용하면...
  - (1) 계산상의 이점:
    - 필터가 입력 데이터 위에서 더 크게 점프하면서 이동함
    - -> 연산 복잡도 줄어듦
    - -> 메모리 절약 및 연산 시간 단축 가능
    - -> 속도 향상 가능
  - (2) 통계적 이점
    - 보폭을 크게 하면 입력 데이터를 더 큰 간격으로 샘플링 -> 불필요한 세부 사항 걸러냄 -> 노이즈 줄임
      - (downsampling)
    - larger reception field
    - overfitting 방지하는 효과 있을 수 있음(세부 사항 생략하므로)

7.4. Multiple Input and Multiple Output Channels

- ***채널 개념

```
import torch
from d2l import torch as d2l
```

7.4.1. Multiple Input Channels

- 입력 데이터에 여러 채널이 포함되어 있는 경우 입력 데이터와 동일한 수의 입력 채널을 갖는 합성곱 커널을 구성하여 입력 데이터와 교차 상관을 수행

```
def corr2d_multi_in(X, K):
    # Iterate through the 0th dimension (channel) of K first, then add them up
    return sum(d2l.corr2d(x, k) for x, k in zip(X, K))
```

```
# 검증
X = torch.tensor([[[0.0, 1.0, 2.0], [3.0, 4.0, 5.0], [6.0, 7.0, 8.0]],
               [[1.0, 2.0, 3.0], [4.0, 5.0, 6.0], [7.0, 8.0, 9.0]]])
K = torch.tensor([[[0.0, 1.0], [2.0, 3.0]], [[1.0, 2.0], [3.0, 4.0]]])

corr2d_multi_in(X, K)
```

```
tensor([[ 56.,  72.],
        [104., 120.]])
```

7.4.2. Multiple Output Channels

```
# 여러 채널의 출력을 계산하기 위한 교차 상관 함수 구현
def corr2d_multi_in_out(X, K):
    # Iterate through the 0th dimension of K, and each time, perform
    # cross-correlation operations with input X. All of the results are
    # stacked together
    return torch.stack([corr2d_multi_in(X, k) for k in K], 0)
```

```
# convolution kernel with three output channels
K = torch.stack((K, K + 1, K + 2), 0)
K.shape
```

```
torch.Size([3, 2, 2, 2])
```

```
# perform cross-correlation operations
corr2d_multi_in_out(X, K)
```

```
tensor([[[ 56.,  72.],
         [104., 120.]],

        [[ 76., 100.],
         [148., 172.]],

        [[ 96., 128.],
         [192., 224.]]])
```

7.4.3. 1x1 Convolutional Layer [사용 이유]

- 채널 간 정보 결합: 1x1 합성곱은 공간적인 정보(가로, 세로 픽셀) 대신, 채널 차원에서의 정보를 결합->여러 채널에서 추출된 특징들을 통합
- 차원 축소: 입력의 채널 수를 줄여 연산 비용을 절감
- 비선형성 추가: 1x1 합성곱 뒤에 활성화 함수(ReLU 등)를 사용하면 네트워크에 비선형성을 추가 가능 -> 모델이 더 복잡한 패턴을 학습할 수 있게 됨
- 모델의 깊이 확장: 네트워크의 깊이를 확장하면서도 각 층이 너무 많은 공간적 정보를 잃지 않도록 1x1 합성곱을 사용하여 공간 정보를 유지하며 채널 차원에서만 연산을 수행
- 병목 계층 역할: 네트워크에서 연산 부담을 줄이고 중요한 특징만 남기기 위한 병목 계층(bottleneck layer)으로 1x1 합성곱을 사용

```
def corr2d_multi_in_out_1x1(X, K):
    c_i, h, w = X.shape
    c_o = K.shape[0]
    X = X.reshape((c_i, h * w))
    K = K.reshape((c_o, c_i))
    # Matrix multiplication in the fully connected layer
    Y = torch.matmul(K, X)
    return Y.reshape((c_o, h, w))
```

```
X = torch.normal(0, 1, (3, 3, 3))
K = torch.normal(0, 1, (2, 3, 1, 1))
Y1 = corr2d_multi_in_out_1x1(X, K)
Y2 = corr2d_multi_in_out(X, K)
assert float(torch.abs(Y1 - Y2).sum()) < 1e-6
```

7.5. Pooling

- sensitivity of convolutional layers to location을 완화
- spatially downsampling representations.

```
import torch
from torch import nn
from d2l import torch as d2l
```

7.5.1. Maximum Pooling and Average Pooling

```
# max pooling
def pool2d(X, pool_size, mode='max'):
    p_h, p_w = pool_size
    Y = torch.zeros((X.shape[0] - p_h + 1, X.shape[1] - p_w + 1))
    for i in range(Y.shape[0]):
        for j in range(Y.shape[1]):
            if mode == 'max':
                Y[i, j] = X[i: i + p_h, j: j + p_w].max()
            elif mode == 'avg':
                Y[i, j] = X[i: i + p_h, j: j + p_w].mean()
    return Y
```

```
# 검증
X = torch.tensor([[0.0, 1.0, 2.0], [3.0, 4.0, 5.0], [6.0, 7.0, 8.0]])
pool2d(X, (2, 2))
```

```
tensor([[4., 5.],
        [7., 8.]])
```

```
# 실험 - average pooling
pool2d(X, (2, 2), 'avg')
```

```
tensor([[2., 3.],
        [5., 6.]])
```

7.5.2. Padding and Stride

```
X = torch.arange(16, dtype=torch.float32).reshape((1, 1, 4, 4))
X
```

```
tensor([[[[ 0.,  1.,  2.,  3.],
          [ 4.,  5.,  6.,  7.],
          [ 8.,  9., 10., 11.],
          [12., 13., 14., 15.]]]])
```

```
pool2d = nn.MaxPool2d(3)
# Pooling has no model parameters, hence it needs no initialization
pool2d(X)
```

```
tensor([[[[10.]]]])
```

```
# padding과 stride 수동 지정
pool2d = nn.MaxPool2d(3, padding=1, stride=2)
pool2d(X)
```

```
tensor([[[[ 5.,  7.],
          [13., 15.]]]])
```

```
pool2d = nn.MaxPool2d((2, 3), stride=(2, 3), padding=(0, 1))
pool2d(X)
```

```
tensor([[[[ 5.,  7.],
          [13., 15.]]]])
```

7.5.3. Multiple Channels

```
X = torch.cat((X, X + 1), 1)
X
```

```
tensor([[[[ 0.,  1.,  2.,  3.],
          [ 4.,  5.,  6.,  7.],
          [ 8.,  9., 10., 11.],
          [12., 13., 14., 15.]],

         [[ 1.,  2.,  3.,  4.],
          [ 5.,  6.,  7.,  8.],
          [ 9., 10., 11., 12.],
          [13., 14., 15., 16.]]]])
```

```
pool2d = nn.MaxPool2d(3, padding=1, stride=2)
pool2d(X)
```

```
tensor([[[[ 5.,  7.],
          [13., 15.]],

         [[ 6.,  8.],
          [14., 16.]]]])
```

7.6. Convolutional Neural Networks (LeNet)

```
import torch
from torch import nn
from d2l import torch as d2l
```

7.6.1. LeNet

(1) a convolutional encoder consisting of two convolutional layers; (2) a dense block consisting of three fully connected layers.

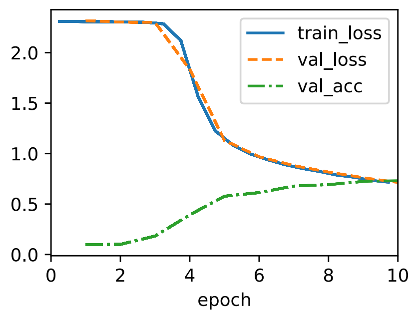
```
def init_cnn(module):
    """Initialize weights for CNNs."""
    if type(module) == nn.Linear or type(module) == nn.Conv2d:
        nn.init.xavier_uniform_(module.weight)
```

```
class LeNet(d2l.Classifier):
    """The LeNet-5 model."""
    def __init__(self, lr=0.1, num_classes=10):
        super().__init__()
        self.save_hyperparameters()
        self.net = nn.Sequential(
            nn.LazyConv2d(6, kernel_size=5, padding=2), nn.Sigmoid(),
            nn.AvgPool2d(kernel_size=2, stride=2),
            nn.LazyConv2d(16, kernel_size=5), nn.Sigmoid(),
            nn.AvgPool2d(kernel_size=2, stride=2),
            nn.Flatten(),
            nn.LazyLinear(120), nn.Sigmoid(),
            nn.LazyLinear(84), nn.Sigmoid(),
            nn.LazyLinear(num_classes))


@d2l.add_to_class(d2l.Classifier)
def layer_summary(self, X_shape):
    X = torch.randn(*X_shape)
    for layer in self.net:
        X = layer(X)
        print(layer.__class__.__name__, 'output shape:\t', X.shape)

model = LeNet()
model.layer_summary((1, 1, 28, 28))
```

```
Conv2d output shape:     torch.Size([1, 6, 28, 28])
Sigmoid output shape:    torch.Size([1, 6, 28, 28])
AvgPool2d output shape:  torch.Size([1, 6, 14, 14])
Conv2d output shape:     torch.Size([1, 16, 10, 10])
Sigmoid output shape:    torch.Size([1, 16, 10, 10])
AvgPool2d output shape:  torch.Size([1, 16, 5, 5])
Flatten output shape:    torch.Size([1, 400])
Linear output shape:     torch.Size([1, 120])
Sigmoid output shape:    torch.Size([1, 120])
Linear output shape:     torch.Size([1, 84])
Sigmoid output shape:    torch.Size([1, 84])
Linear output shape:     torch.Size([1, 10])
```

7.6.2. Training

```
trainer = d2l.Trainer(max_epochs=10, num_gpus=1)
data = d2l.FashionMNIST(batch_size=128)
model = LeNet(lr=0.1)
model.apply_init([next(iter(data.get_dataloader(True)))[0]], init_cnn)
trainer.fit(model, data)
```



Discussions

- What happens to the activations when you feed significantly different images into the network (e.g., cats, cars, or even random noise)?
    - 1. Early Layers: Detect basic features like edges or textures, leading to different activations based on the input (e.g., car lines, cat fur).
    - 2. Deeper Layers: Produce weaker or erratic activations since they focus on high-level patterns specific to the trained classes.
    - 3. Final Layer: Outputs uncertain or random predictions for unfamiliar inputs like noise or images outside the training set.

8. Modern Convolutional Neural Networks

- VGG network:
    - 'blocks' of layers를 repeat한다는 배경에서, foundation model(대규모로 사전 훈련된 모델)을 다양한 관련된 작업에 재활용한다는 concept.
    - 루프와 서브루틴을 사용

8.2. Networks Using Blocks(VGG)

```
import torch
from torch import nn
from d2l import torch as d2l
```

8.2.1. VGG Blocks

- a sequence of convolutions with kernels with padding of 1 followed by a max-pooling layer with stride of 2로 구성됨

```
def vgg_block(num_convs, out_channels):
    layers = []
    for _ in range(num_convs):
        layers.append(nn.LazyConv2d(out_channels, kernel_size=3, padding=1))
        layers.append(nn.ReLU())
    layers.append(nn.MaxPool2d(kernel_size=2,stride=2))
    return nn.Sequential(*layers)
```

8.2.2. VGG Network

- 2 parts
  - 1) mostly of convolutional and pooling layers
  - 2) fully connected layers that are identical to those in AlexNet but without linearlity

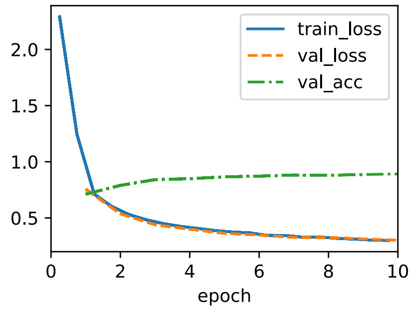
```
class VGG(d2l.Classifier):
    def __init__(self, arch, lr=0.1, num_classes=10):
        super().__init__()
        self.save_hyperparameters()
        conv_blks = []
        for (num_convs, out_channels) in arch:
            conv_blks.append(vgg_block(num_convs, out_channels))
        self.net = nn.Sequential(
            *conv_blks, nn.Flatten(),
            nn.LazyLinear(4096), nn.ReLU(), nn.Dropout(0.5),
            nn.LazyLinear(4096), nn.ReLU(), nn.Dropout(0.5),
            nn.LazyLinear(num_classes))
        self.net.apply(d2l.init_cnn)
```

```
VGG(arch=((1, 64), (1, 128), (2, 256), (2, 512), (2, 512))).layer_summary(
    (1, 1, 224, 224))
```

```
Sequential output shape:         torch.Size([1, 64, 112, 112])
Sequential output shape:         torch.Size([1, 128, 56, 56])
Sequential output shape:         torch.Size([1, 256, 28, 28])
Sequential output shape:         torch.Size([1, 512, 14, 14])
Sequential output shape:         torch.Size([1, 512, 7, 7])
Flatten output shape:    torch.Size([1, 25088])
Linear output shape:     torch.Size([1, 4096])
ReLU output shape:       torch.Size([1, 4096])
Dropout output shape:    torch.Size([1, 4096])
Linear output shape:     torch.Size([1, 4096])
ReLU output shape:       torch.Size([1, 4096])
Dropout output shape:    torch.Size([1, 4096])
Linear output shape:     torch.Size([1, 10])
```

- 8.2.3. Training
  - channel 수가 AlexNet 보다 적음

```
model = VGG(arch=((1, 16), (1, 32), (2, 64), (2, 128), (2, 128)), lr=0.01)
trainer = d2l.Trainer(max_epochs=10, num_gpus=1)
data = d2l.FashionMNIST(batch_size=128, resize=(224, 224))
model.apply_init([next(iter(data.get_dataloader(True)))[0]], d2l.init_cnn)
trainer.fit(model, data)
```



Discussions

- AlexNet 대비 VGG는 계산 측면에서 훨씬 느리고, 더 많은 GPU 메모리도 필요
    - 이유: 더 많은 층과 작은 필터를 사용하기 때문에, 더 많은 매개변수를 필요로 함

8.6. Residual Networks(ResNet) and ResNeXt

```
import torch
from torch import nn
from torch.nn import functional as F
from d2l import torch as d2l
```

8.6.1. Function Classes

- identity function 도입
    - 모든 추가 레이어가 항등 함수를 요소 중 하나로 더 쉽게 포함해야 한다는 아이디어
    - residual block을 통해 각 layer가 identity function 학습
- residual block 개념은 Transformers, GNN 등에 활용

8.6.2. Residual Blocks

- 1) add the input to the output before applying the ReLU onlinearity whenever use_1x1conv=False
- 2) adjust channels and resolution by means of a 1x1 convolution before adding

```
class Residual(nn.Module):
    """The Residual block of ResNet models."""
    def __init__(self, num_channels, use_1x1conv=False, strides=1):
        super().__init__()
        self.conv1 = nn.LazyConv2d(num_channels, kernel_size=3, padding=1,
                                   stride=strides)
        self.conv2 = nn.LazyConv2d(num_channels, kernel_size=3, padding=1)
        if use_1x1conv:
            self.conv3 = nn.LazyConv2d(num_channels, kernel_size=1,
                                       stride=strides)
        else:
            self.conv3 = None
        self.bn1 = nn.LazyBatchNorm2d()
        self.bn2 = nn.LazyBatchNorm2d()

    def forward(self, X):
        Y = F.relu(self.bn1(self.conv1(X)))
        Y = self.bn2(self.conv2(Y))
        if self.conv3:
            X = self.conv3(X)
        Y += X
        return F.relu(Y)
```

```
blk = Residual(3)
X = torch.randn(4, 3, 6, 6)
blk(X).shape
```

```
torch.Size([4, 3, 6, 6])
```

8.6.3. ResNet Model

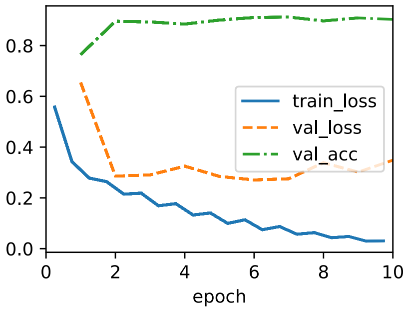
- ResNet의 처음 두 계층은 GoogLeNet의 계층과 동일

```
class ResNet(d2l.Classifier):
    def b1(self):
        return nn.Sequential(
            nn.LazyConv2d(64, kernel_size=7, stride=2, padding=3),
            nn.LazyBatchNorm2d(), nn.ReLU(),
            nn.MaxPool2d(kernel_size=3, stride=2, padding=1))


@d2l.add_to_class(ResNet)
def block(self, num_residuals, num_channels, first_block=False):
    blk = []
    for i in range(num_residuals):
        if i == 0 and not first_block:
            blk.append(Residual(num_channels, use_1x1conv=True, strides=2))
        else:
            blk.append(Residual(num_channels))
    return nn.Sequential(*blk)


@d2l.add_to_class(ResNet)
def __init__(self, arch, lr=0.1, num_classes=10):
    super(ResNet, self).__init__()
    self.save_hyperparameters()
    self.net = nn.Sequential(self.b1())
    for i, b in enumerate(arch):
        self.net.add_module(f'b{i+2}', self.block(*b, first_block=(i==0)))
    self.net.add_module('last', nn.Sequential(
        nn.AdaptiveAvgPool2d((1, 1)), nn.Flatten(),
        nn.LazyLinear(num_classes)))
    self.net.apply(d2l.init_cnn)


class ResNet18(ResNet):
    def __init__(self, lr=0.1, num_classes=10):
        super().__init__(((2, 64), (2, 128), (2, 256), (2, 512)),
                       lr, num_classes)

ResNet18().layer_summary((1, 1, 96, 96))
```

```
Sequential output shape:         torch.Size([1, 64, 24, 24])
Sequential output shape:         torch.Size([1, 64, 24, 24])
Sequential output shape:         torch.Size([1, 128, 12, 12])
Sequential output shape:         torch.Size([1, 256, 6, 6])
Sequential output shape:         torch.Size([1, 512, 3, 3])
Sequential output shape:         torch.Size([1, 10])
```

8.6.4. Training

```
model = ResNet18(lr=0.01)
trainer = d2l.Trainer(max_epochs=10, num_gpus=1)
data = d2l.FashionMNIST(batch_size=128, resize=(96, 96))
model.apply_init([next(iter(data.get_dataloader(True)))[0]], d2l.init_cnn)
trainer.fit(model, data)
```



8.6.5. ResNeXt

```
class ResNeXtBlock(nn.Module):
    def __init__(self, num_channels, groups, bot_mul, use_1x1conv=False,
                 strides=1):
        super().__init__()
        bot_channels = int(round(num_channels * bot_mul))
        self.conv1 = nn.LazyConv2d(bot_channels, kernel_size=1, stride=1
        self.conv2 = nn.LazyConv2d(bot_channels, kernel_size=3,
                                   stride=strides, padding=1,
                                   groups=bot_channels//groups)
        self.conv3 = nn.LazyConv2d(num_channels, kernel_size=1, stride=1
```

```python
        self.bn1 = nn.LazyBatchNorm2d()
        self.bn2 = nn.LazyBatchNorm2d()
        self.bn3 = nn.LazyBatchNorm2d()
        if use_1x1conv:
            self.conv4 = nn.LazyConv2d(num_channels, kernel_size=1,
                                       stride=strides)
            self.bn4 = nn.LazyBatchNorm2d()
        else:
            self.conv4 = None

    def forward(self, X):
        Y = F.relu(self.bn1(self.conv1(X)))
        Y = F.relu(self.bn2(self.conv2(Y)))
        Y = self.bn3(self.conv3(Y))
        if self.conv4:
            X = self.bn4(self.conv4(X))
```

```python
blk = ResNeXtBlock(32, 16, 1)
X = torch.randn(4, 32, 96, 96)
blk(X).shape
```

```
torch.Size([4, 32, 96, 96])
```

Discussions

- Inception 블록 과 residual 블록:
  - Inception uses multiple paths while resnet uses one single path with X.
  - Inception 블록: multi-scale feature를 통해 복잡하고 다양한 패턴을 인식, 병렬 경로로 인한 계산량과 복잡도가 high
  - Residual 블록: skip connection을 통해 매우 깊은 네트워크에서도 안정적인 학습, 계산 효율성 측면에서 더 유리