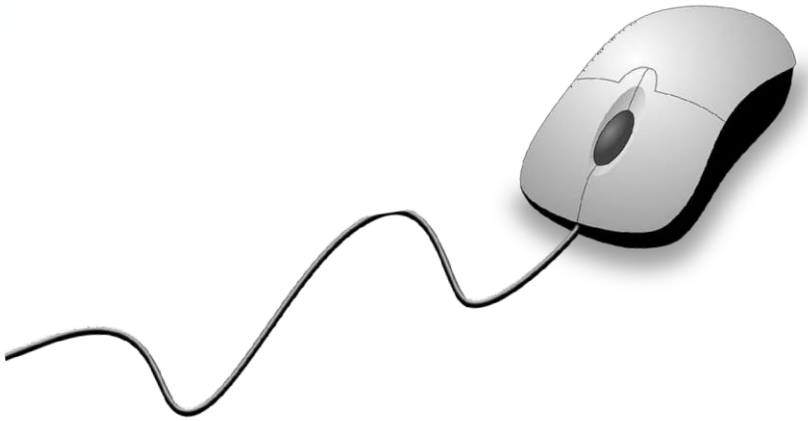
공개SW솔루션설치&활용가이드

기타 >AI



mxnet

# 제대로 배워보자

How to Use Open Source Software

Open Source Software Installation & Application Guide

OPENUP

오픈소스 소프트웨어 통합지원센터

Open Source Software Support Center

# CONTENTS

# 1. 개요

| | | | |
|---|---|---|---|
| **소개** | • 오픈소스기반의 분산 딥러닝 소프트웨어 프레임워크(distributed deep-learning framework)<br>• Amazon AWS, MS Azure 등 퍼블릭 클라우드에서 딥러닝 프레임워크로 사용 가능 | | |
| **주요기능** | • 자동 미분화<br>• GPU, mobile에서도 동작<br>• 혼합 패러다임 지원(symbolic/imperative) | | |
| **대분류** | • 기타 | **소분류** | • AI |
| **라이선스형태** | • Apache 2.0 | **사전설치 솔루션** | • 없음 |
| | | **버전** | • 1.5.1 (2019년 10월 기준) |
| **특징** | • 사용하는 GPU 숫자가 늘어남에 따라 비례하게 성능 개선을 기대할 수 있음<br>• 거의 대부분의 Device 를 지원<br>• Python, R, Scala, Clojure, Julia, Perl, JAVA, cpp 언어 지원 | | |
| **개발회사/커뮤니티** | • Apache Software Foundation | | |
| **공식 홈페이지** | • https://mxnet.incubator.apache.org/ | | |

# 2. 기능요약


**혼합 패러다임 지원**

Torch, Chiner 등의 딥러닝 프레임워크는 imperative 한 스타일을 지원하고, TensorFlow 는 Symbolic 스카일을 지원하는 것과 달리 MXNet 은 imperative 와 Symbolic 스타일을 섞어서 프로그래밍할 수 있습니다.

**분산 학습 지원**

MXNet은 대부분의 하드웨어를 지원하고 GPU 수 중가에 따라 학습 성능이 비례하게 증가하므로 분산 환경에 최적화 되어 있습니다. 즉, 매우 규모가 큰 프로젝트도 상대적으로 짧은 시간내에 처리할 수 있게 됩니다. 또한, MXNet은 최근 Uber 에서 개발한 분산 학습 프레임워크인 Horovod 를 지원하기로 하기도 했습니다.

**다양한 언어 지원**

MXNet은 C++, JavaScript, Python, R, Matlab, Julia, Scala, Clojure 및 Perl를 비롯한 다양한 프로그래밍 언어를 지원하므로 이미 익숙한 언어로 시작할 수 있습니다. 하지만 모델을 구축할 때 사용된 언어의 종류와 관계없이 최대 성능을 내기 위해 백엔드에서는 모든 코드가 C++로 컴파일 됩니다.

# 3. 기본구성


세부 목차

1. 환경설정
2. MXNet 설치
3. MXNet 설치 확인

# 3. 기본구성

## 3.1 환경 설정(1)

Linux VM 에 CentOS 7 버전을 미리 설치하여 준비하고 최신 버전으로 업그레이드

```
Last login: Sun Oct 27 07:53:37 2019 from 192.168.56.1
[root@localhost ~]# yum upgrade -y
Loaded plugins: fastestmirror
Loading mirror speeds from cached hostfile
 * base: ty1.mirror.newmediaexpress.com
 * extras: data.aonenetworks.kr
 * updates: data.aonenetworks.kr
No packages marked for update
[root@localhost ~]#
```

# 3. 기본구성

## 3.1 환경 설정(2)

- Python 설치

```
[root@localhost ~]# yum install python python3
Loaded plugins: fastestmirror
Loading mirror speeds from cached hostfile
 * base: ty1.mirror.newmediaexpress.com
 * extras: data.aonenetworks.kr
 * updates: data.aonenetworks.kr
```

- Python 3.x version 사용을 위해 symbolic link 수정

```
[root@localhost bin]# unlink python
[root@localhost bin]# ls -al python*
lrwxrwxrwx. 1 root root     9 Oct 27 07:54 python2 -> python2.7
-rwxr-xr-x. 1 root root  7216 Aug  7 09:52 python2.7
lrwxrwxrwx. 1 root root     9 Oct 27 08:04 python3 -> python3.6
-rwxr-xr-x. 2 root root 11408 Aug  8 02:29 python3.6
-rwxr-xr-x. 2 root root 11408 Aug  8 02:29 python3.6m
[root@localhost bin]# ln -s /bin/python3.6 /bin/python
```

# 3. 기본구성

## 3.1 환경 설정(3)

- pip symbolic link 수정

```
[root@localhost bin]# ln -s /bin/pip3.6 /bin/pip
[root@localhost bin]# ls -al pip*
lrwxrwxrwx. 1 root root  11 Oct 27 08:11 pip -> /bin/pip3.6
lrwxrwxrwx. 1 root root   9 Oct 27 08:04 pip-3 -> ./pip-3.6
lrwxrwxrwx. 1 root root   8 Oct 27 08:04 pip-3.6 -> ./pip3.6
-rwxr-xr-x. 1 root root 407 Aug  8 02:05 pip3.6
[root@localhost bin]#
```

- Jupyter notebook 설치

  # Jupyter Notebook 은 오픈소스 웹 어플리케이션으로 라이브코드, 등식, 시각화와 설명을 포함한 문서를 만들고 공유할 수 있습니다. 주로 머신러닝, 통계 모델링 등에 사용됩니다.

```
[root@localhost bin]# pip install jupyter
WARNING: Running pip install with root privileges is generally n
--user` instead.
Requirement already satisfied: jupyter in /usr/local/lib/python3
Requirement already satisfied: jupyter-console in /usr/local/lib
pyter)
```

# 3. 기본구성

## 3.1 환경 설정(4)

- Jupyter notebook 서버 설정 파일 생성

```
[root@localhost bin]# jupyter notebook --generate-config
Writing default config to: /root/.jupyter/jupyter_notebook_config.py
```

- Jupyter notebook 접속 시 사용할 암호 생성

```
[root@localhost bin]# ipython
Python 3.6.8 (default, Aug  7 2019, 17:28:10)
Type 'copyright', 'credits' or 'license' for more information
IPython 7.9.0 -- An enhanced Interactive Python. Type '?' for help.

In [1]: from notebook.auth import passwd

In [2]: passwd()
Enter password:
Verify password:
Out[2]: 'sha1:d0a231c69e23:b59dc89694baa4460275a79be30afcacd9132466'
```

# 3. 기본구성

## 3.1 환경 설정(6)

- 원격 접속을 위해 jupyter notebook 컨피그 파일 수정
  - ✓ c.NotebookApp.ip = 'server ip address'
  - ✓ c.NotebookApp.password = '이전 단계에서 생성한 암호'
  - ✓ c.NotebookApp.open_browser = False

```
203 ## The IP address the notebook server will listen on.
204 c.NotebookApp.ip = '192.168.56.178'
```

```
276 c.NotebookApp.password = 'sha1:d0a231c69e23:b59dc89694baa4460275a79be30afcacd9132466'
277
```

```
267 c.NotebookApp.open_browser = False
268
```

- Jupiter Notebook 실행

```
[root@localhost home]# jupyter notebook --allow-root
[I 09:51:26.566 NotebookApp] Serving notebooks from local directory: /home
[I 09:51:26.566 NotebookApp] The Jupyter Notebook is running at:
[I 09:51:26.566 NotebookApp] http://192.168.56.178:8888/
[I 09:51:26.566 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice t
o skip confirmation).
```

# 3. 기본구성

## 3.2 MXNet 설치

- Linux OS에 Python 을 활용하고 CPU를 사용하는 경우 아래 명령으로 MXNet 설치 가능
  - ✓ pip install mxnet
  - ✓ pip install mxnet-mkl

※ MXNet 공식 홈페이지 에서 MXNet 에서 제공되는 다양한 GPU 용 패키지들을 확인 가능

| Package | 1.5.1 | 1.4.1 | 1.3.1 | 1.2.1 | 1.1.0 | 1.0.0 | 0.12.1 | 0.11.0 |
|---|---|---|---|---|---|---|---|---|
| mxnet-cu101mkl | ✓ | ✓ | | | | | | |
| mxnet-cu101 | ✓ | ✓ | | | | | | |
| mxnet-cu100mkl | ✓ | ✓ | | | | | | |
| mxnet-cu100 | ✓ | ✓ | | | | | | |
| mxnet-cu92mkl | ✓ | ✓ | ✓ | ✓ | | | | |
| mxnet-cu92 | ✓ | ✓ | ✓ | ✓ | | | | |
| mxnet-cu90mkl | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | |
| mxnet-cu90 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | |
| mxnet-cu80mkl | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| mxnet-cu80 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| mxnet-mkl | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| mxnet | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |

# 3. 기본구성

## 3.3 MXNet 설치확인

- 아래 python 코드가 정상적으로 동작하는지 확인하여 MXNet 이 정상적으로 설치 되었는 지 검증 가능

```
[root@localhost home]# python
Python 3.6.8 (default, Aug  7 2019, 17:28:10)
[GCC 4.8.5 20150623 (Red Hat 4.8.5-39)] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import mxnet as mx
>>> a = mx.nd.ones((2,3))
>>> b = a * 2 + 1
>>> b.asnumpy()
array([[3., 3., 3.],
       [3., 3., 3.]], dtype=float32)
>>>
```

# 4. 기본 사용법

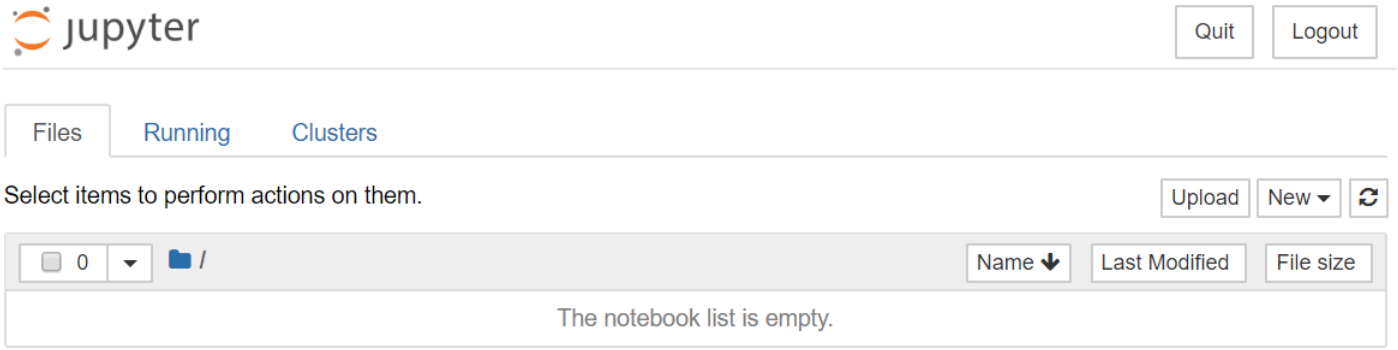

세부 목차

1. NDArray
2. Neural Network 생성 방법
3. GPU 사용
4. 학습

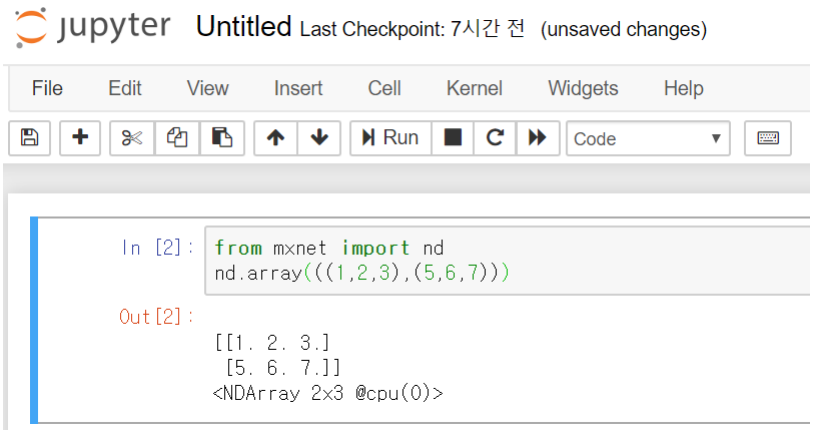
# 4. 기본 사용법

## 4.1 NDArray (1)

- MXNet 에서 Data 처리에 사용되는 NDArray 를 사용
- jupyter notebook 으로 접속 (http://jupiter_server_ip:8888)




- 테스트를 위해 **"New"** 버튼을 클릭하여 새로운 노트북을 만들고, 2D array 를 생성



# 4. 기본 사용법

## 4.1 NDArray (2)

- 숫자 1로 채워진 2 x 3의 NDArray 를 만드는 방법

```
In [4]: x = nd.ones((2,3))
        x

Out[4]:
        [[1. 1. 1.]
         [1. 1. 1.]]
        <NDArray 2x3 @cpu(0)>
```

- 1에서 9 사이의 랜덤한 값을 가지는 2 x 3 의 NDArray 를 만드는 방법

```
In [5]: y = nd.random.uniform(1,9,(2,3))
        y

Out[5]:
        [[5.390508  5.742757  6.7215147]
         [7.754126  5.822107  7.863565 ]]
        <NDArray 2x3 @cpu(0)>
```

- 2 로 채워진 2 x 3 의 NDArray 를 생성하는 방법

```
In [9]: x = nd.full((2,3), 2.0)
        x

Out[9]:
        [[2. 2. 2.]
         [2. 2. 2.]]
        <NDArray 2x3 @cpu(0)>
```

# 4. 기본 사용법

## 4.2 Neural Network생성방법

- 2개의 Output 을 가진 Dense Layer 생성

In [1]:
```
from mxnet import nd
from mxnet.gluon import nn
layer = nn.Dense(2)
layer
```

Out[1]:
```
Dense(None -> 2, linear)
```

- LeNet을 생성 코드

In [20]:
```
net = nn.Sequential()
# Add a sequence of layers.
net.add(nn.Conv2D(channels=6, kernel_size=5, activation='relu'),
        nn.MaxPool2D(pool_size=2, strides=2),
        nn.Conv2D(channels=16, kernel_size=3, activation='relu'),
        nn.MaxPool2D(pool_size=2, strides=2),
        nn.Dense(120, activation="relu"),
        nn.Dense(84, activation="relu"),
        nn.Dense(10))
net
```

Out[20]:
```
Sequential(
  (0): Conv2D(None -> 6, kernel_size=(5, 5), stride=(1, 1), Activation(relu))
  (1): MaxPool2D(size=(2, 2), stride=(2, 2), padding=(0, 0), ceil_mode=False, global_pool=False, pool_type=max, layout=NCHW)
  (2): Conv2D(None -> 16, kernel_size=(3, 3), stride=(1, 1), Activation(relu))
  (3): MaxPool2D(size=(2, 2), stride=(2, 2), padding=(0, 0), ceil_mode=False, global_pool=False, pool_type=max, layout=NCHW)
  (4): Dense(None -> 120, Activation(relu))
  (5): Dense(None -> 84, Activation(relu))
  (6): Dense(None -> 10, linear)
)
```

# 4. 기본 사용법

## 4.3 GPU 사용

- GPU를 사용하면 CPU를 사용했을 때에 비해 빠르게 계산 가능
- 머신에 Nvidia GPU가 장착되어 있고, CUDA가 정상적으로 설치되어 있다면, MXNet 에서 GPU를 사용 가능
  - ✓ pip uninstall mxnet
  - ✓ pip install mxnet-cu101
- 아직까지는 AMD나 Intel 과 같은 GPU는 아직 지원하지 않음
- GPU 사용에 필요한 플러그인을 불러오고 GPU에 데이터를 배정하는 코드

In [2]:
```
from mxnet import nd, gpu, gluon, autograd
from mxnet.gluon import nn
from mxnet.gluon.data.vision import datasets, transforms
import time

x = nd.ones((3,4), ctx=gpu())
x
```

# 4. 기본 사용법

## 4.4 학습

- 데이터를 준비하고 Neural Network 를 준비했다면 Gluon Trainer 를 활용하여 학습을 진행 가능

In [3]:
```
gpus = mx.test_utils.list_gpus()
ctx =  [mx.gpu()] if gpus else [mx.cpu(0), mx.cpu(1)]
net.initialize(mx.init.Xavier(magnitude=2.24), ctx=ctx)
trainer = gluon.Trainer(net.collect_params(), 'sgd', {'learning_rate': 0.02})
```

In [ ]:
```
epoch = 10
metric = mx.metric.Accuracy()
softmax_cross_entropy_loss = gluon.loss.SoftmaxCrossEntropyLoss()
for i in range(epoch):
    train_data.reset()
    for batch in train_data:
        data = gluon.utils.split_and_load(batch.data[0], ctx_list=ctx, batch_axis=0)
        label = gluon.utils.split_and_load(batch.label[0], ctx_list=ctx, batch_axis=0)
        outputs = []
        with ag.record():
            for x, y in zip(data, label):
                z = net(x)
                loss = softmax_cross_entropy_loss(z, y)
                loss.backward()
                outputs.append(z)
        metric.update(label, outputs)
        trainer.step(batch.data[0].shape[0])
    name, acc = metric.get()
    metric.reset()
    print('training acc at epoch %d: %s=%f'%(i, name, acc))
```

# 5. 활용 예제


세부 목차

1. 데이터 준비
2. Neural Network 설정
3. 학습

# 5. 활용 예제

## 5.1 데이터준비(1)

- 28 x 28 사이즈의 그레이 스케일 이미지인 Fashion-Mnist 데이터셋을 사용하여 Image Recognition 하는 예제
- 필요한 라이브러리 설치
  - ✓ Pip install matplotlib
- 아래 코드를 통해 필요한 라이브러리를 부르고 학습에 사용할 데이터셋을 준비

In [2]:
```
from mxnet import nd, gluon, init, autograd
from mxnet.gluon import nn
from mxnet.gluon.data.vision import datasets, transforms
from IPython import display
import matplotlib.pyplot as plt
import time
```

In [3]:
```
mnist_train = datasets.FashionMNIST(train=True)
X, y = mnist_train[0]
('X shape: ', X.shape, 'X dtype', X.dtype, 'y:', y)
```

```
Downloading /root/.mxnet/datasets/fashion-mnist/train-images-idx3-ubyte.gz from https://apache-mxnet.s3-accelerate.dualstack.amazonaws.com/gluon/dataset/fashion-mnist/train-images-idx3-ubyte.gz...
Downloading /root/.mxnet/datasets/fashion-mnist/train-labels-idx1-ubyte.gz from https://apache-mxnet.s3-accelerate.dualstack.amazonaws.com/gluon/dataset/fashion-mnist/train-labels-idx1-ubyte.gz...
```

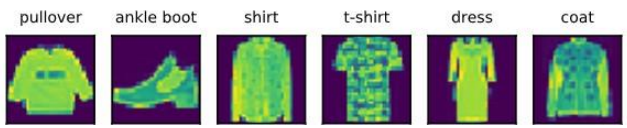
Out[3]:
```
('X shape: ', (28, 28, 1), 'X dtype', numpy.uint8, 'y:', 2)
```

# 5. 활용 예제

## 5.1 데이터준비(2)

- NDArray 에 (height, width, channel) 로 저장하고, Lable 은 지정하고,
  첫번째 6개의 샘플을 보여주는 코드

In [22]:
```
text_labels = ['t-shirt', 'trouser', 'pullover', 'dress', 'coat',
               'sandal', 'shirt', 'sneaker', 'bag', 'ankle boot']
X, y = mnist_train[0:6]
# plot images
display.set_matplotlib_formats('svg')
_, figs = plt.subplots(1, X.shape[0], figsize=(5, 5))
for f,x,yi in zip(figs, X,y):
    # 3D->2D by removing the last channel dim
    f.imshow(x.reshape((28,28)).asnumpy())
    ax = f.axes
    ax.set_title(text_labels[int(yi)])
    ax.title.set_fontsize(7)
    ax.get_xaxis().set_visible(False)
    ax.get_yaxis().set_visible(False)
plt.show()
```



# 5. 활용 예제

## 5.1 데이터준비(3)

- Gluon model 에 준비한 데이터셋을 활용하기 위해 데이터를 (channel, height, width) 형태의 실수로 변환하고, 데이터를 재배치하고 검증용 데이터를 만드는 코드

In [5]:
```
transformer = transforms.Compose([
    transforms.ToTensor(),
    transforms.Normalize(0.13, 0.31)])
mnist_train = mnist_train.transform_first(transformer)
```

In [6]:
```
batch_size = 256
train_data = gluon.data.DataLoader(
    mnist_train, batch_size=batch_size, shuffle=True, num_workers=4)
```

In [7]:
```
for data, label in train_data:
    print(data.shape, label.shape)
    break
```

```
(256, 1, 28, 28) (256,)
```

In [8]:
```
mnist_valid = gluon.data.vision.FashionMNIST(train=False)
valid_data = gluon.data.DataLoader(
    mnist_valid.transform_first(transformer),
    batch_size=batch_size, num_workers=4)
```

```
Downloading /root/.mxnet/datasets/fashion-mnist/t10k-images-idx3-ubyte.gz from https://apache-mxnet.s3-accelerate.dualstack.amazonaws.com/gluon/dataset/fashion-mnist/t10k-images-idx3-ubyte.gz...
Downloading /root/.mxnet/datasets/fashion-mnist/t10k-labels-idx1-ubyte.gz from https://apache-mxnet.s3-accelerate.dualstack.amazonaws.com/gluon/dataset/fashion-mnist/t10k-labels-idx1-ubyte.gz...
```

# 5. 활용 예제

## 5.2 Neural Network설정

- LeNet 을 생성하고 weight initialize method 로 Xavier 를 사용하는 코드

```
In [10]: net = nn.Sequential()
         net.add(nn.Conv2D(channels=6, kernel_size=5, activation='relu'),
                 nn.MaxPool2D(pool_size=2, strides=2),
                 nn.Conv2D(channels=16, kernel_size=3, activation='relu'),
                 nn.MaxPool2D(pool_size=2, strides=2),
                 nn.Flatten(),
                 nn.Dense(120, activation="relu"),
                 nn.Dense(84, activation="relu"),
                 nn.Dense(10))
         net.initialize(init=init.Xavier())
```

- Loss fuction 과 optimization method 를 정의하는 코드

```
In [11]: softmax_cross_entropy = gluon.loss.SoftmaxCrossEntropyLoss()

In [12]: trainer = gluon.Trainer(net.collect_params(), 'sgd', {'learning_rate': 0.1})
```

# 5. 활용 예제

## 5.3 학습

- 학습 모델의 정확도를 계산하기 위한 함수를 아래와 같이 정의

```
In [13]: def acc(output, label):
             # output: (batch, num_output) float32 ndarray
             # label: (batch, ) int32 ndarray
             return (output.argmax(axis=1) ==
                     label.astype('float32')).mean().asscalar()
```

- 학습을 수행

```
In [14]: for epoch in range(10):
             train_loss, train_acc, valid_acc = 0., 0., 0.
             tic = time.time()
             for data, label in train_data:
                 # forward + backward
                 with autograd.record():
                     output = net(data)
                     loss = softmax_cross_entropy(output, label)
                 loss.backward()
                 # update parameters
                 trainer.step(batch_size)
                 # calculate training metrics
                 train_loss += loss.mean().asscalar()
                 train_acc += acc(output, label)
             # calculate validation accuracy
             for data, label in valid_data:
                 valid_acc += acc(net(data), label)
             print("Epoch %d: loss %.3f, train acc %.3f, test acc %.3f, in %.1f sec" % (
                     epoch, train_loss/len(train_data), train_acc/len(train_data),
                     valid_acc/len(valid_data), time.time()-tic))
```

```
Epoch 0: loss 0.822, train acc 0.695, test acc 0.812, in 19.8 sec
Epoch 1: loss 0.467, train acc 0.827, test acc 0.844, in 21.6 sec
Epoch 2: loss 0.399, train acc 0.854, test acc 0.865, in 19.6 sec
Epoch 3: loss 0.362, train acc 0.868, test acc 0.874, in 19.3 sec
Epoch 4: loss 0.339, train acc 0.876, test acc 0.880, in 20.0 sec
Epoch 5: loss 0.317, train acc 0.885, test acc 0.891, in 20.6 sec
Epoch 6: loss 0.304, train acc 0.888, test acc 0.890, in 19.7 sec
Epoch 7: loss 0.291, train acc 0.894, test acc 0.894, in 19.2 sec
Epoch 8: loss 0.280, train acc 0.896, test acc 0.894, in 19.1 sec
Epoch 9: loss 0.271, train acc 0.900, test acc 0.898, in 19.2 sec
```

- 약 90% 정도의 정확도를 가지는 학습 모델을 만듬

# Open Source Software Installation & Application Guide