Introduction to Machine Learning

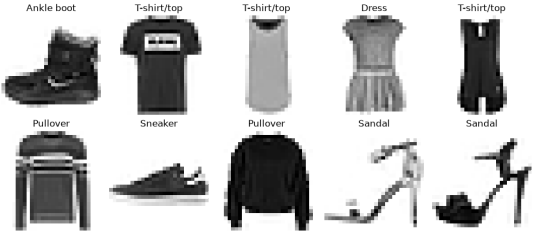
Day 14: Convolutional Neural Networks

Joanna Bieri DATA301

Get out your handwritten notes!

Mini Review - Fashion MNIST

REVIEW



70,000 black and white pictures, 28 by 28, in 10 classes. We train on 10,000 and validate on 2,000.

Mini Review - Scores and a New Loss Function

REVIEW

The network ends with **10 outputs**, one **score** for each class. The biggest score is the prediction.

```
loss_fn = nn.CrossEntropyLoss()
scores = model(X_valid)
y_pred = scores.argmax(dim=1)
```

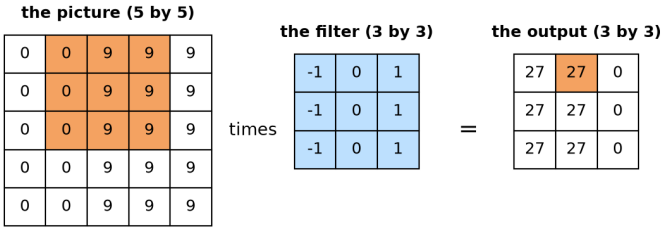
Mini Review - A Number to Beat

REVIEW

```
mlp = nn.Sequential(
    nn.Flatten(),           # 28 x 28 -> 784
    nn.Linear(784, 100),
    nn.ReLU(),
    nn.Linear(100, 10))
```

After 5 epochs the MLP gets **0.858** of the validation pictures right.

Mini Review - Convolution



Orange window, one row: (0 x -1) + (9 x 0) + (9 x 1) = 9

All three rows: 9 + 9 + 9 = 27

Then slide the window one step and do it again.

Multiply each pixel by the filter number on top of it, add it all up, then slide the window over one pixel.

Mini Review - nn.Conv2d

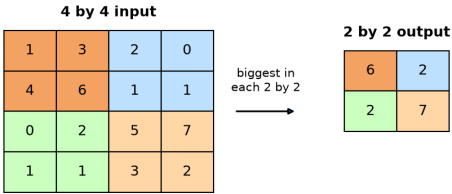
REVIEW

```
conv = nn.Conv2d(
    in_channels=1,
    out_channels=16,
    kernel_size=3,
    padding=1)

going in:   torch.Size([5, 1, 28, 28])
coming out: torch.Size([5, 16, 28, 28])
```

Mini Review - Pooling

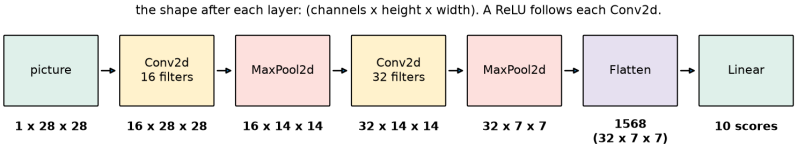
REVIEW



`nn.MaxPool2d(2)` keeps the biggest number in each 2 by 2 block. It has nothing to learn.

Mini Review - A Small CNN

REVIEW



Convolution, ReLU, pooling, twice, then flatten and one Linear layer for the 10 scores.

Why Not an MLP?

ASK

a tiny 4 by 4 picture

A	B	C	D
E	F	G	H
I	J	K	L
M	N	O	P

F and J are neighbors in the picture, one right on top of the other.

flatten

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P

In the flattened row they are 4 apart. In a 28 by 28 picture they would be 28 apart.

What does an MLP lose when we flatten a picture?

Why Not an MLP?

ANSWER

What does an MLP lose when we flatten a picture?

- Which pixels are next to each other.
- Two pixels on top of each other in the picture end up 28 places apart in the row.
- The MLP has to learn from scratch which pixels belong together.

Convolution

PREDICT

patch

1	1	5
1	1	5
1	1	5

filter

-1	0	1
-1	0	1
-1	0	1

Multiply each pixel by the filter number in the same spot and add everything up. What number comes out?

Convolution

ANSWER

The 3 by 3 patch (1, 1, 5 in every row) times the vertical edge filter. What comes out?

```
each row:  1*(-1) + 1*0 + 5*1 = 4
three rows: 4 + 4 + 4 = 12
```

The right side is bigger than the left, so the filter gives a big number. If the patch were all the same number it would give 0.

Convolution

PREDICT

`patch`

7	7	7
7	7	7
0	0	0

`filter`

-1	0	1
-1	0	1
-1	0	1

Same filter, new patch. This patch HAS an edge. What number comes out, and why?

Convolution

ANSWER

Same vertical edge filter on a patch with rows 7 7 7, 7 7 7, 0 0 0. What comes out, and why?

```
each row:  7*(-1) + 7*0 + 7*1 = 0
three rows: 0 + 0 + 0 = 0
```

0! This edge goes left to right (a horizontal edge), and this filter only notices changes from left to right. A horizontal edge needs a different filter, the same one turned on its side. That is why a CNN learns many filters.

Convolution

ASK

In a CNN, who picks

the numbers in the filters?

Convolution

ANSWER

In a CNN, who picks the numbers in the filters?

- Nobody picks them. They are weights, learned by gradient descent like the weights in nn.Linear.
- One filter might learn vertical edges, another horizontal edges, another a corner.
- The same filter is used at every spot in the picture, so a 3 by 3 filter has only 9 weights.

Pooling

PREDICT

```
color picture: 3 x 32 x 32

nn.Conv2d(3, 8, kernel_size=3, padding=1)
nn.MaxPool2d(2)
nn.Conv2d(8, 16, kernel_size=3, padding=1)
nn.MaxPool2d(2)
nn.Flatten()
nn.Linear(???, 10)
```

With your group: write the shape after every layer. What number goes in place of the ???

Pooling

ANSWER

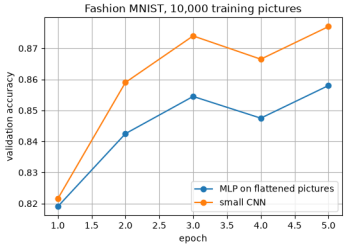
A 3 x 32 x 32 color picture goes through Conv2d(3, 8), MaxPool2d(2), Conv2d(8, 16), MaxPool2d(2), Flatten. What goes in Linear(???, 10)?

```
3 x 32 x 32  ->  8 x 32 x 32  ->  8 x 16 x 16
             -> 16 x 16 x 16  -> 16 x 8 x 8  ->  1024
```

16 * 8 * 8 = 1024, so nn.Linear(1024, 10). The number of filters becomes the number of channels, padding=1 keeps the size, and each pooling halves it. Challenge: what if we leave out padding=1?

A Small CNN

PREDICT



Now the pictures are color, 3 x 28 x 28. How many numbers does the MLP layer nn.Linear(3*28*28, 100) learn? The CNN layer nn.Conv2d(3, 16, kernel_size=3)? Why can the CNN do better with fewer?

A Small CNN

ANSWER

Color pictures, 3 x 28 x 28: how many numbers in nn.Linear(2352, 100) and in nn.Conv2d(3, 16, kernel_size=3)?

```
MLP:  2352 * 100 + 100      = 235,300
CNN:  16 * (3 * 3 * 3) + 16 = 448
```

- Each filter is small and reused at every spot in the picture.
- It looks at small patches, so it uses which pixels are neighbors.
- On our black and white pictures: CNN 0.877, MLP 0.858 after 5 epochs.

CNN Architectures

ASK

ResNet has 152 layers.

What can happen to the numbers

as they pass through 152 layers?

Why does that make

batch normalization so important?

CNN Architectures

ANSWER

ResNet has 152 layers. What can happen to the numbers on the way through, and why does batch normalization help?

- Through many layers the signal can shrink or grow (Day 11 and Day 12).
- Batch normalization keeps it a good size during training.
- Our 2 layer CNN does not have this problem, so we left it out.

Questions?