Convolutional neural networks

Practice Session 8

Changho Suh

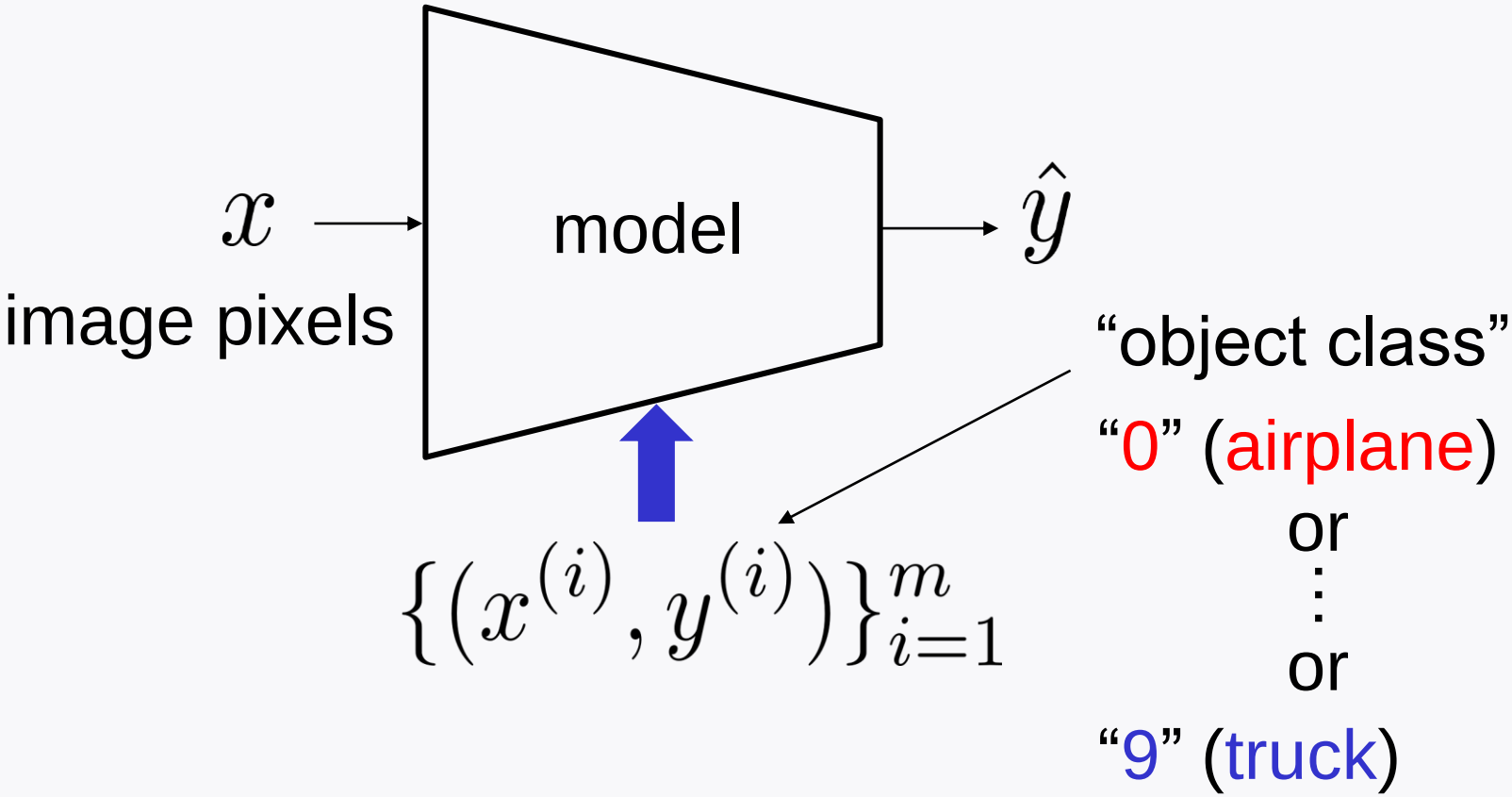
January 24, 2024

Outline

Will implement the simplified AlexNet.

Task: Image recognition (CIFAR10)

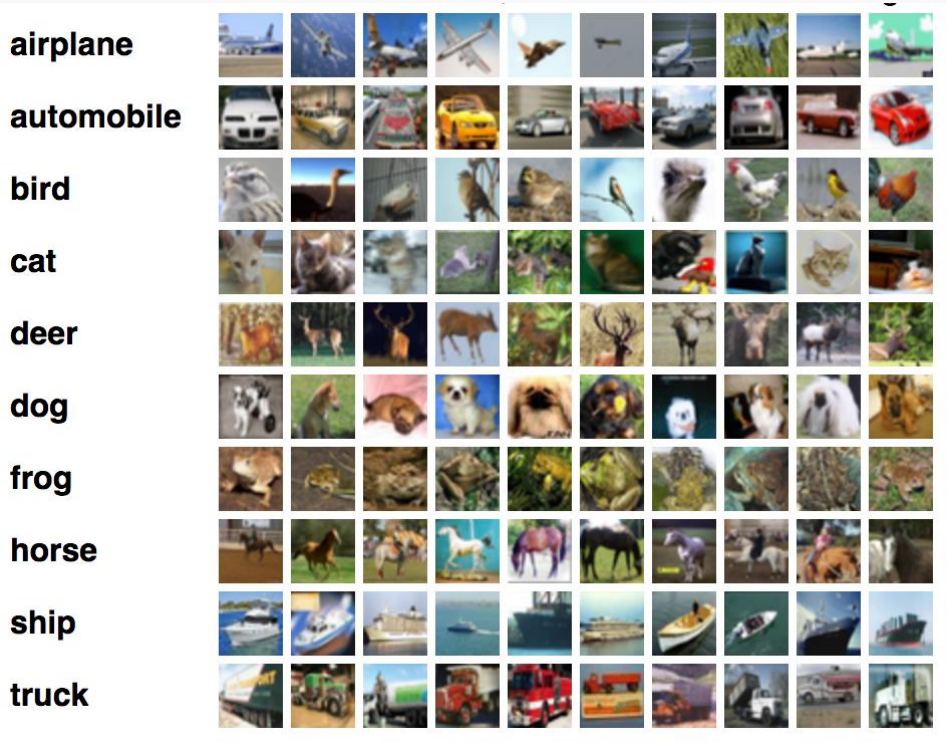
Image recognition



CIFAR10

Canadian **I**nstitute **F**or **A**dvanced **R**esearch

10 classes:



$$m = 50,000 \qquad m_{\text{test}} = 10,000$$

CIFAR10

Data: RGB scale **image pixels** $\in \mathbf{R}^{32\times32\times3}$

Label: human-annotated labels $\in \{0, \ldots, 9\}$

(10 classes)

Example:

(, 6) “frog”

Loading CIFAR10

```
from tensorflow.keras.datasets import cifar10

(X_train, y_train), (X_test, y_test) = cifar10.load_data()

print(X_train.shape)
print(X_test.shape)
print(y_train.shape)
print(y_test.shape)
```

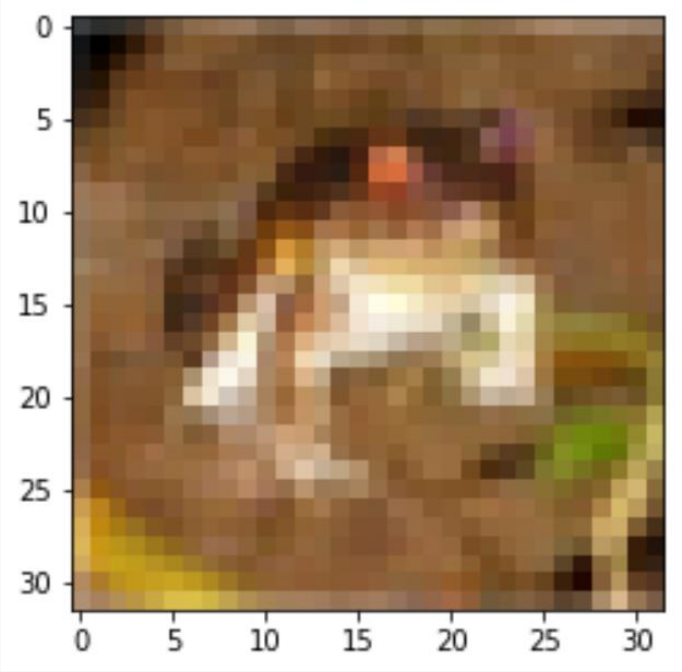
```
(50000, 32, 32, 3)
(10000, 32, 32, 3)
(50000, 1)
(10000, 1)
```

```
# normalization
X_train, X_test = X_train/255.0, X_test/255.0
```

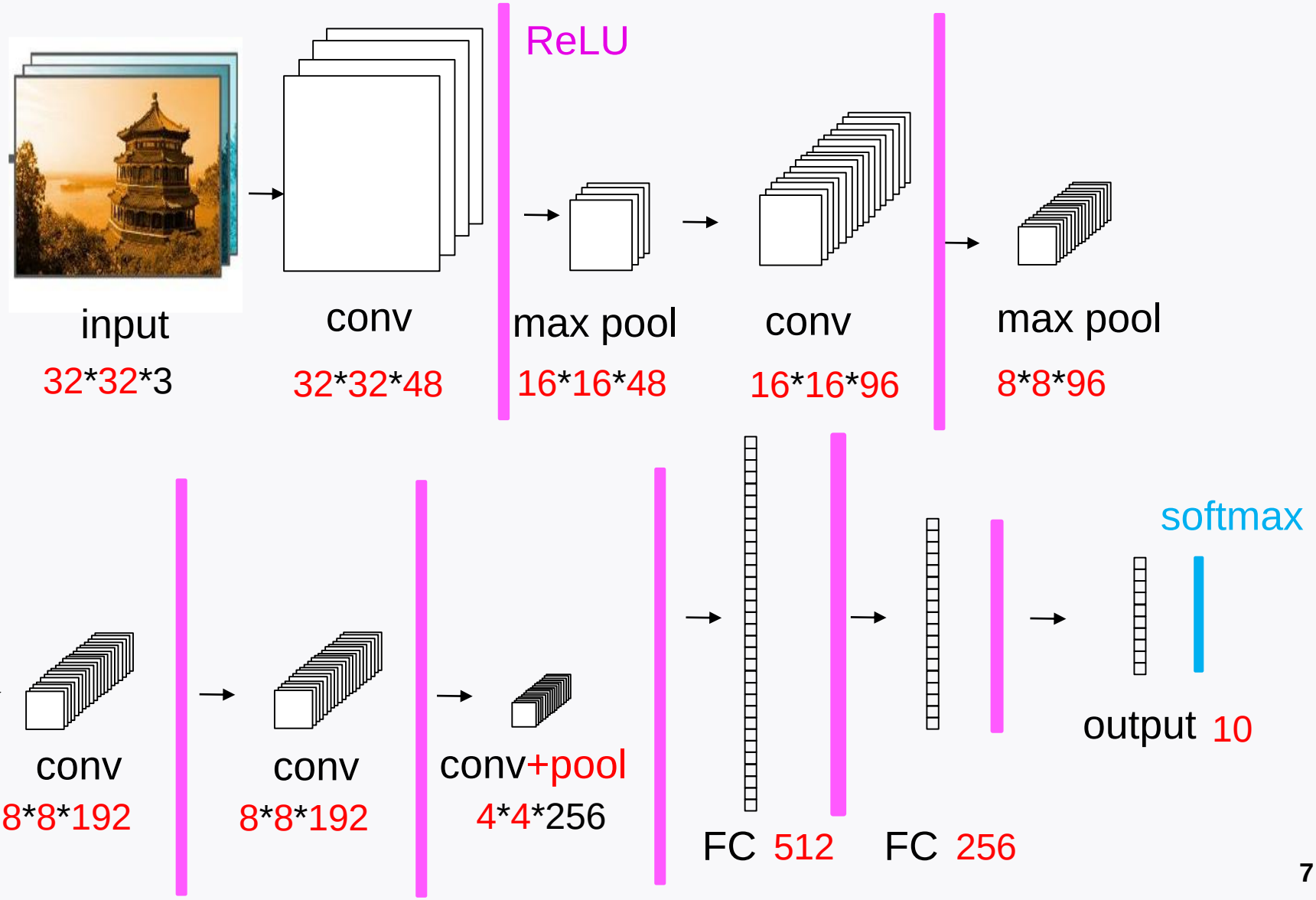
Data visualization

```
import matplotlib.pyplot as plt

plt.figure()
plt.imshow(X_train[0])
plt.show()
```



Simplified AlexNet

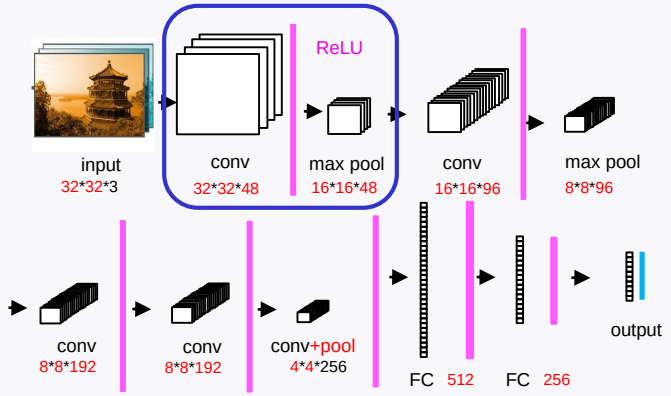


Simplified AlexNet: Tensorflow coding

```
from keras.models import Sequential
from keras.layers import Flatten, Dense
from keras.layers import Conv2D
from keras.layers import MaxPool2D

model_alexnet = Sequential()

#1st stack: [conv]+[ReLU]+[pool]
model_alexnet.add(Conv2D(input_shape = (32,32,3),
                         kernel_size = (3,3),
                         strides=(1,1),
                         filters=48,
                         padding='same',
                         activation='relu'))
model_alexnet.add(MaxPool2D(pool_size = (2,2),
                            strides = (2,2),
                            padding = 'valid'))
```

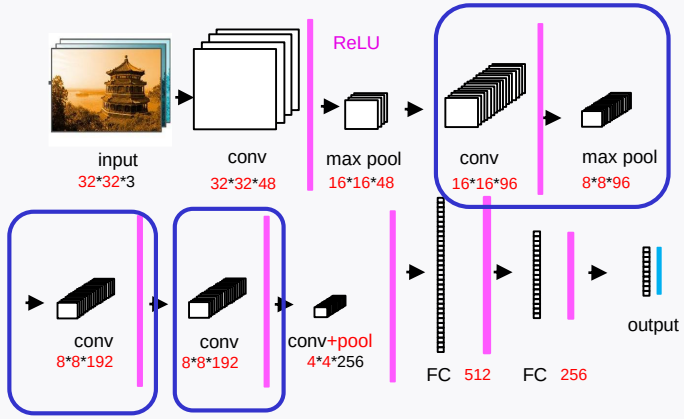


Simplified AlexNet: Tensorflow coding

```
#2nd stack: [conv]+[ReLU]+[pool]
model_alexnet.add(Conv2D(kernel_size = (3,3),
                         strides=(1,1),
                         filters=96,
                         padding='same',
                         activation='relu'))
model_alexnet.add(MaxPool2D(pool_size = (2,2),
                            strides = (2,2),
                            padding = 'valid'))
```

```
# 3rd stack: [conv]+[ReLU]
model_alexnet.add(Conv2D(kernel_size = (3,3),
                         strides=(1,1),
                         filters=192,
                         padding='same',
                         activation='relu'))
```

```
# 4th stack: [conv]+[ReLU]
model_alexnet.add(Conv2D(kernel_size = (3,3),
                         strides=(1,1),
                         filters=192,
                         padding='same',
                         activation='relu'))
```

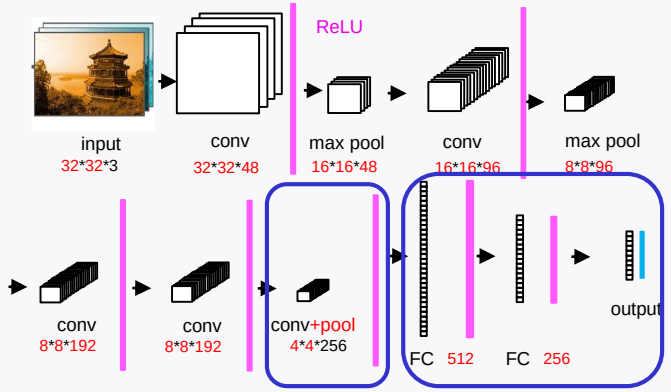


Simplified AlexNet: Tensorflow coding

```
# 5th stack: [conv]+[ReLU]+[pool]
model_alexnet.add(Conv2D(kernel_size = (3,3),
                         strides=(1,1),
                         filters=256,
                         padding='same',
                         activation='relu'))
model_alexnet.add(MaxPool2D(pool_size = (2,2),
                            strides = (2,2),
                            padding = 'valid'))
```

```
# 6th layer: 512 Fully connected
model_alexnet.add(Flatten())
model_alexnet.add(Dense(512,activation='relu'))
# 7th layer: 256 fully connected
model_alexnet.add(Dense(256,activation='relu'))
```

```
# 8th layer: 10 output
model_alexnet.add(Dense(10,activation='softmax'))
```



Compile

```
from tensorflow.keras.optimizers import Adam

opt = Adam(learning_rate = 0.001,
           beta_1 = 0.9,
           beta_2 = 0.999)

model_alexnet.compile(optimizer = opt,
                      loss='sparse_categorical_crossentropy',
                      metrics=['acc'])
```

Training & evaluation

```
# Training
model_alexnet.fit(X_train,y_train,epochs=20)
```

```
# Evaluation
test_performance = model_alexnet.evaluate(X_test,y_test)
print(test_performance)
```

```
[1.6277161836624146, 0.7396000027656555]
```

Look ahead

Will implement simplified ResNet in the context of image recognition (CIFAR10).