Misclassification Results

- Notebook 18Misclassification

Try this:

- Please print your misclassification results from your homework

Convolutional Neural Network Techniques

Dr. Mongkol Ekpanyapong

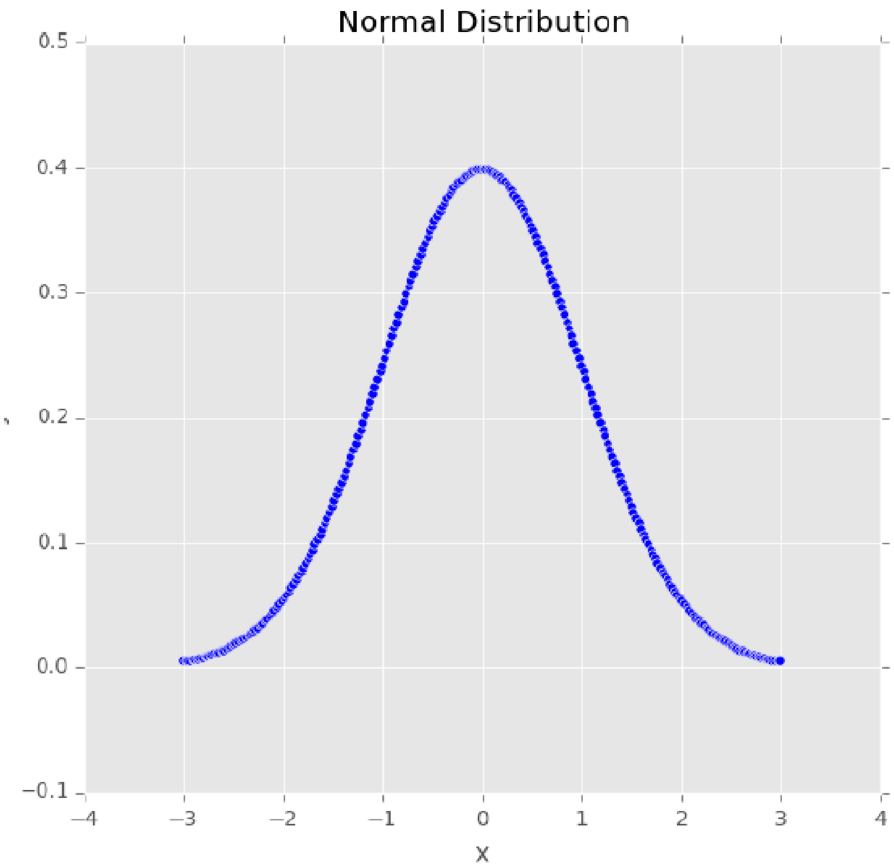
What is Data Augmentation?

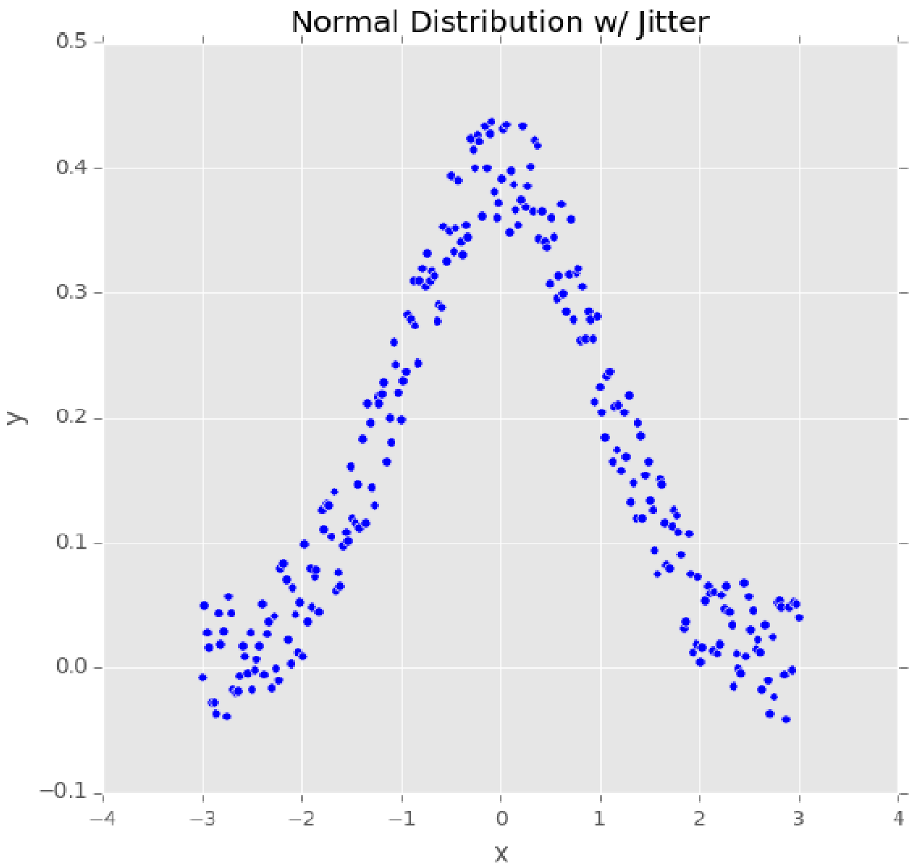
The goal is to increase the generalizability of the model.

- During the training, we provide slightly modified versions of the input data points

- At testing time, we do not apply data augmentation

Networks as Feature Extractors



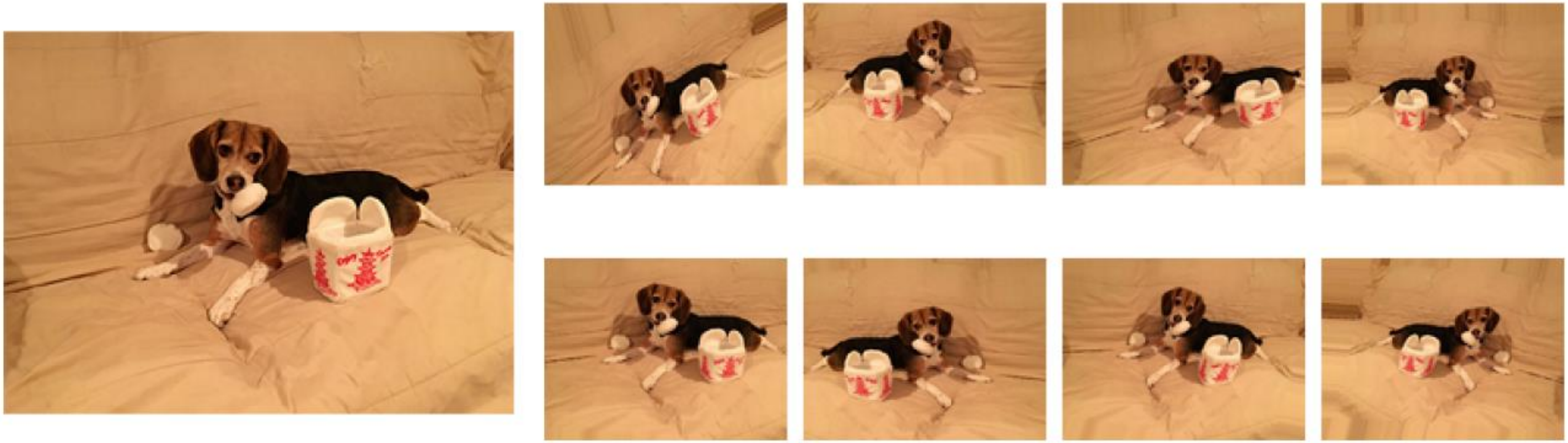


A normal distribution

With jitter

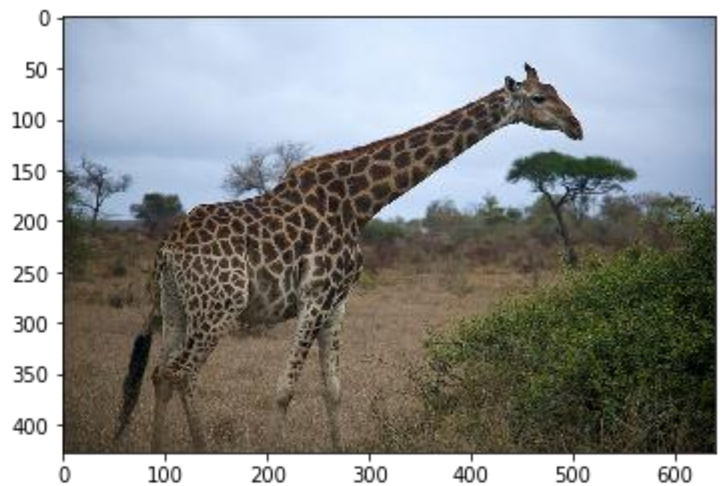
Data Augmentation

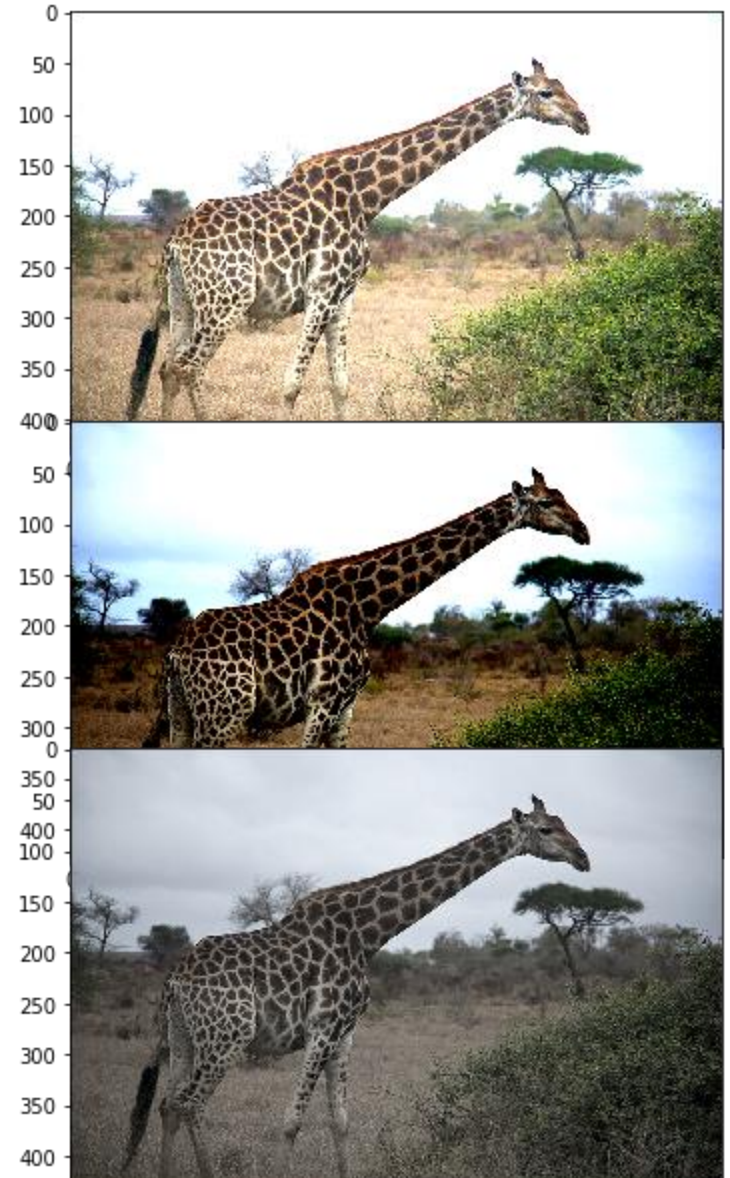
- One of the most effective way of reducing overfitting
- It also helps addressing of not having enough data or enough variation



Data Augmentation

- Color shifting





Data Augmentation Variety

- Flipping (Horizontal/Vertical)
- Brightness
- Rotations
- Zoom
- Cropping
- Skew/Sheer

Data Augmentation

- Only done during training
- Turn it off during validation/testing
- Supports are provided both Keras and Pytorch

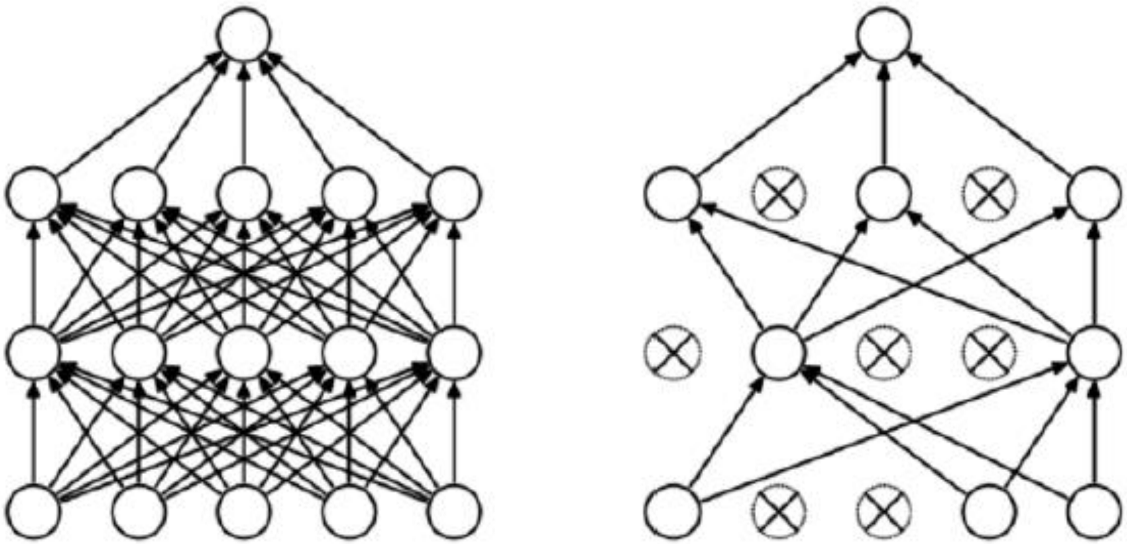
Example

```
train_datagen = ImageDataGenerator(
        rescale = 1./255,
        rotation_range=10,
        width_shift_range=0.1,
        height_shift_range=0.1,
        shear_range=0.1,
        zoom_range=0.1,
        horizontal_flip=True,
        fill_mode='nearest')
```

Dropout

- The dropout is a regularization technique
- The idea is to keep turn off some neural so that we use many good neural nodes for prediction (not relying only on one)
- It provides multiple redundant nodes

Dropout Example



(a) Standard Neural Net (b) After applying dropout.

Dropout

- Note that we randomly add dropout only during the training time,(testing time, we activate back all nodes)

- Model:

INPUT => CONV => RELU => BN => POOL => FC => DO => FC => DO

Example

```
model.add(Dropout(0.2))
```

Batch Normalization (BN)

- Recall- the output of CNN is

Batch size x Feature Map Height x Feature Map Width x Channels

- BN calculates the mean and standard deviation of each input variable, to a layer per mini-batch and uses this to perform the standardization

Batch Normalization

- It is introduced by Ioffe and Szegedy in 2015 paper, Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift to add BN layer
- The idea is to normalize the data where x_i is mini-batch
- The equation is as follow:

$$\hat{x}_i = \frac{x_i - \mu_\beta}{\sqrt{\sigma_\beta^2 + \varepsilon}}$$

when

$$\mu_\beta = \frac{1}{M}\sum_{i=1}^{m} x_i \qquad \sigma_\beta^2 = \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_\beta)^2$$

Batch Normalization

- Advantage:
  - Help reduce the number of epochs for training and help for regularization
  - Recommend to put wherever we can
- Drawback:
  - Slow down the system
- Model:

INPUT => CONV => RELU => BN => POOL => FC

Example

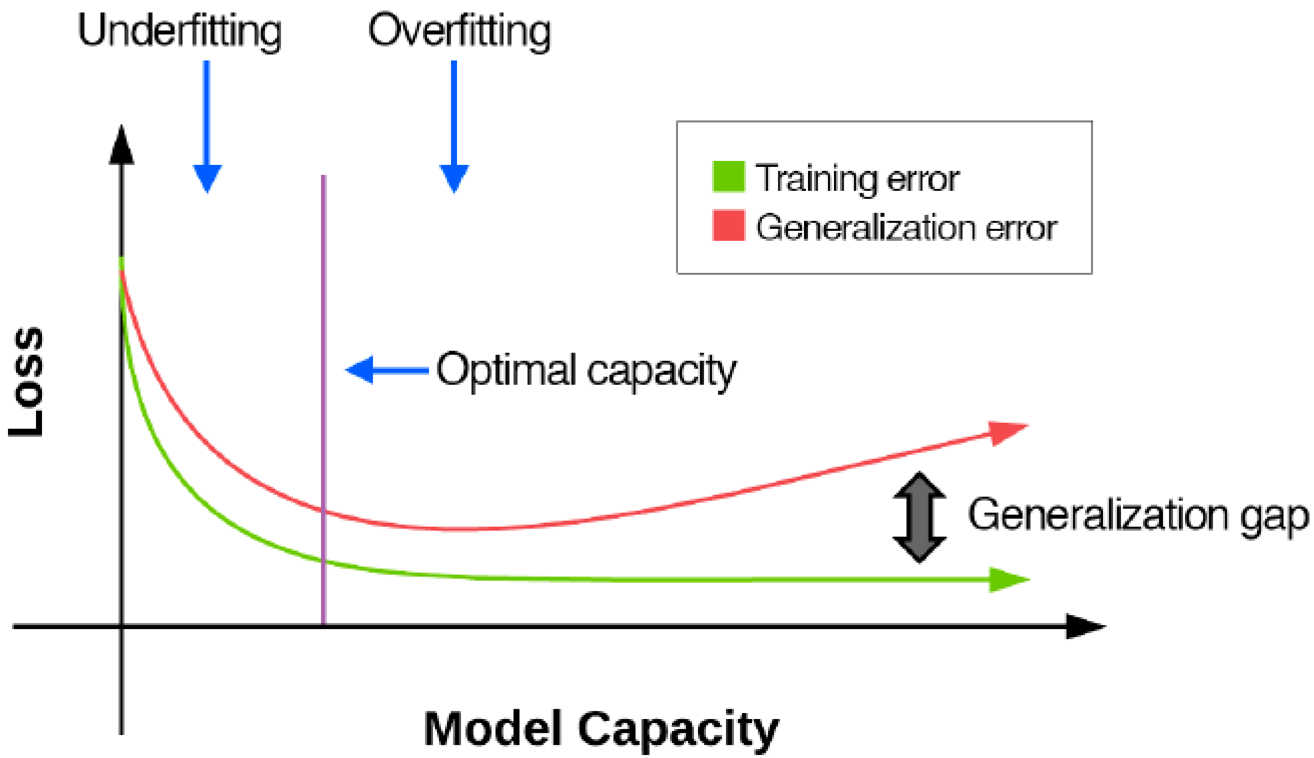
```
model.add(BatchNormalization())
```

Rule of Training The Model

1. Reduce the training loss as much as possible

2. Ensuring the gap between the training and testing is small

Controlling A Model



Model capacity is the capacity of the neural network, e.g., # layers

Model Capacity

- We can increase capacity by:
  - adding more layers and neurons

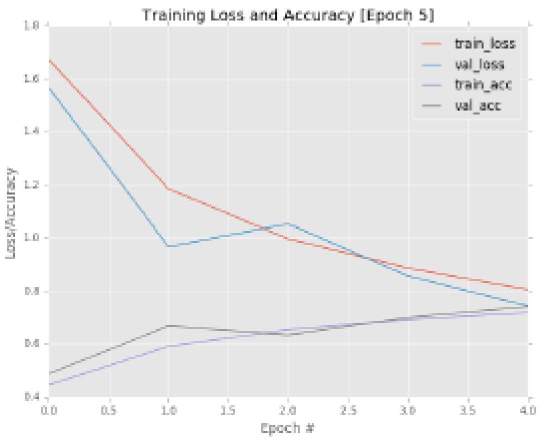
- We can decrease capacity by:
  - Removing layers and neurons
  - Applying regularization techniques (weight decay, dropout, data augmentation, early stopping, etc.)

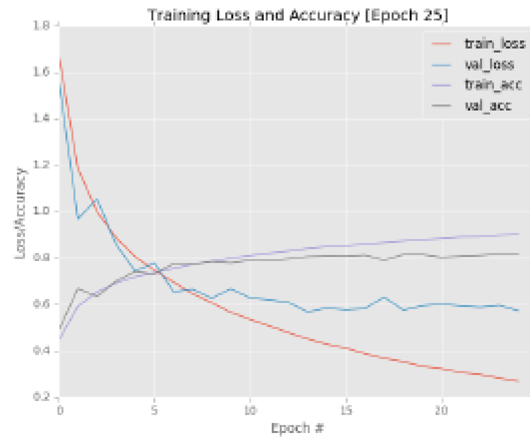
Can validation loss be lower than training loss?

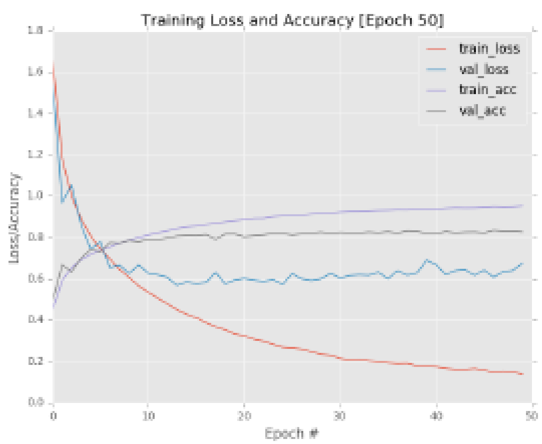
Why?

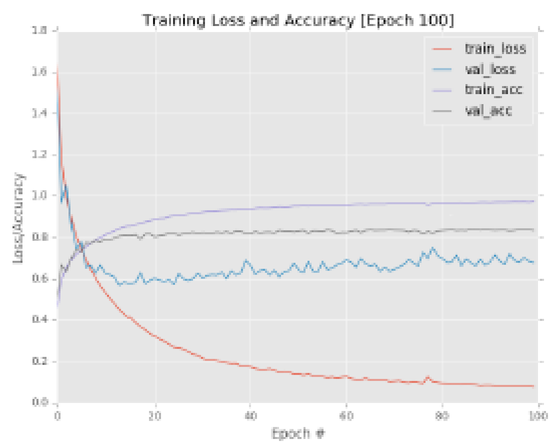
- Your training data is seeing all the “hard” examples while your validation data consists of easy data points
- You perform data augmentation during the training
- You are not training hard enough

Spotting the Training/Validation







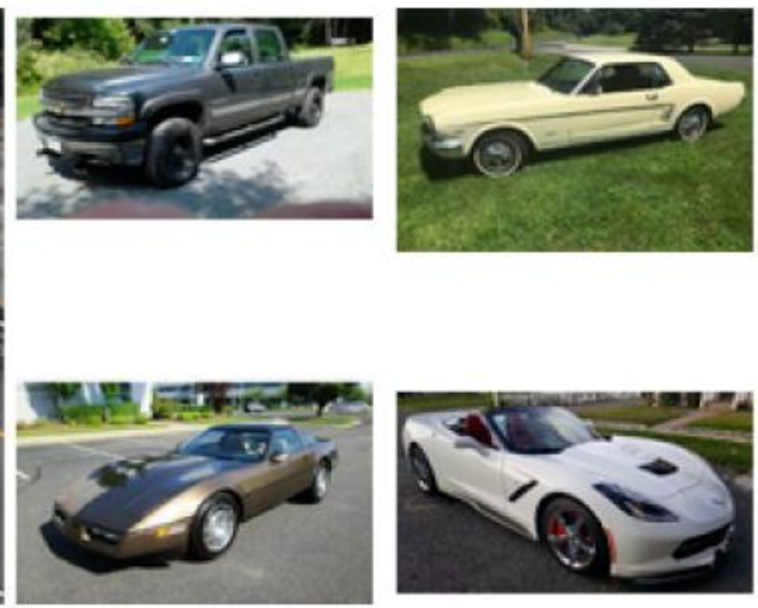


Example of Regularization in Keras

- Notebook 19 Regularization

Example of Data Training

- Left: cars on a highway (our problem)
- Right: train images



What is the problem?

Optimal Pathway to Apply Deep Learning

"Most issues in applied deep learning come from training data/testing data mismatch. In some scenarios this issue just doesn't come up, but you'd be surprised how often applied machine learning projects use training data (which is easy to collect and annotate) that is different from the target application.",

-Andrew Ng

Training on Mnist-cloth

```
import numpy as np
import os

import sys
assert sys.version_info >= (3, 5)

# Scikit-Learn ≥0.20 is required
import sklearn


import matplotlib as mpl
import matplotlib.pyplot as plt
import tensorflow as tf
from tensorflow import keras

mpl.rc('axes', labelsize=14)
mpl.rc('xtick', labelsize=12)
mpl.rc('ytick', labelsize=12)
```

```
# Where to save the figures
PROJECT_ROOT_DIR = "."
CHAPTER_ID = "ann"
IMAGES_PATH = os.path.join(PROJECT_ROOT_DIR, "images", CHAPTER_ID)
os.makedirs(IMAGES_PATH, exist_ok=True)

def save_fig(fig_id, tight_layout=True, fig_extension="png", resolution=300):
    path = os.path.join(IMAGES_PATH, fig_id + "." + fig_extension)
    print("Saving figure", fig_id)
    if tight_layout:
        plt.tight_layout()
    plt.savefig(path, format=fig_extension, dpi=resolution)

tf.__version__
keras.__version__
```

```
fashion_mnist = keras.datasets.fashion_mnist
(X_train_full, y_train_full), (X_test, y_test) = fashion_mnist.load_data()
X_train_full.shape
X_train_full.dtype
X_valid, X_train = X_train_full[:5000] / 255., X_train_full[5000:] / 255.
y_valid, y_train = y_train_full[:5000], y_train_full[5000:]
X_test = X_test / 255.
plt.imshow(X_train[0], cmap="binary")
plt.axis('off')
plt.show()

y_train
class_names = ["T-shirt/top", "Trouser", "Pullover", "Dress", "Coat",
               "Sandal", "Shirt", "Sneaker", "Bag", "Ankle boot"]
```

```python
class_names[y_train[0]]
X_valid.shape
X_test.shape
n_rows = 4
n_cols = 10
plt.figure(figsize=(n_cols * 1.2, n_rows * 1.2))
for row in range(n_rows):
    for col in range(n_cols):
        index = n_cols * row + col
        plt.subplot(n_rows, n_cols, index + 1)
        plt.imshow(X_train[index], cmap="binary", interpolation="nearest")
        plt.axis('off')
        plt.title(class_names[y_train[index]], fontsize=12)
plt.subplots_adjust(wspace=0.2, hspace=0.5)
save_fig('fashion_mnist_plot', tight_layout=False)
plt.show()
```

```
model = keras.models.Sequential()
model.add(keras.layers.Flatten(input_shape=[28, 28]))
model.add(keras.layers.Dense(300, activation="relu"))
model.add(keras.layers.Dense(100, activation="relu"))
model.add(keras.layers.Dense(10, activation="softmax"))
keras.backend.clear_session()
np.random.seed(42)
tf.set_random_seed(42)
model = keras.models.Sequential([
    keras.layers.Flatten(input_shape=[28, 28]),
    keras.layers.Dense(300, activation="relu"),
    keras.layers.Dense(100, activation="relu"),
    keras.layers.Dense(10, activation="softmax")
])
model.layers
model.summary()
from keras.utils import plot_model
```

```
# plot_model(model, "my_fashion_mnist_model.png")
hidden1 = model.layers[1]
hidden1.name
model.get_layer(hidden1.name) is hidden1
weights, biases = hidden1.get_weights()
weights
weights.shape
biases
biases.shape
model.compile(loss="sparse_categorical_crossentropy",
              optimizer="sgd",
              metrics=["accuracy"])
history = model.fit(X_train, y_train, epochs=30,
                    validation_data=(X_valid, y_valid))
history.params
print(history.epoch)
history.history.keys()
import pandas as pd
```

```
pd.DataFrame(history.history).plot(figsize=(8, 5))
plt.grid(True)
plt.gca().set_ylim(0, 1)
save_fig("keras_learning_curves_plot")
plt.show()
model.evaluate(X_test, y_test)
X_new = X_test[:3]
y_proba = model.predict(X_new)
y_proba.round(2)
y_pred = model.predict_classes(X_new)
y_pred
np.array(class_names)[y_pred]
y_new = y_test[:3]
y_new
```

```
plt.figure(figsize=(7.2, 2.4))
for index, image in enumerate(X_new):
    plt.subplot(1, 3, index + 1)
    plt.imshow(image, cmap="binary", interpolation="nearest")
    plt.axis('off')
    plt.title(class_names[y_test[index]], fontsize=12)
plt.subplots_adjust(wspace=0.2, hspace=0.5)
save_fig('fashion_mnist_images_plot', tight_layout=False)
plt.show()
```

Homework

- From Mnist-cloth, try to build your own CNN model to outperform FCN given in the example.

Questions?