Fundamentals of machine learning

Generalization: The goal of machine learning

Underfitting and overfitting

Noisy training data

Ambiguous features

Rare features and spurious correlations

Adding white-noise channels or all-zeros channels to MNIST

```
from tensorflow.keras.datasets import mnist
import numpy as np

(train_images, train_labels), _ = mnist.load_data()
train_images = train_images.reshape((60000, 28 * 28))
train_images = train_images.astype("float32") / 255

train_images_with_noise_channels = np.concatenate(
    [train_images, np.random.random((len(train_images), 784))], axis=1)

train_images_with_zeros_channels = np.concatenate(
    [train_images, np.zeros((len(train_images), 784))], axis=1)
```

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/mn
11493376/11490434 [==============================] - 0s 0us/step
11501568/11490434 [==============================] - 0s 0us/step
```

Training the same model on MNIST data with noise channels or all-zero channels

```
from tensorflow import keras
from tensorflow.keras import layers

def get_model():
    model = keras.Sequential([
        layers.Dense(512, activation="relu"),
        layers.Dense(10, activation="softmax")
    ])
    model.compile(optimizer="rmsprop",
                  loss="sparse_categorical_crossentropy",
                  metrics=["accuracy"])
    return model

model = get_model()
history_noise = model.fit(
    train_images_with_noise_channels, train_labels,
    epochs=10,
    batch_size=128,
    validation_split=0.2)

model = get_model()
history_zeros = model.fit(
    train_images_with_zeros_channels, train_labels,
    epochs=10,
    batch_size=128,
    validation_split=0.2)
```

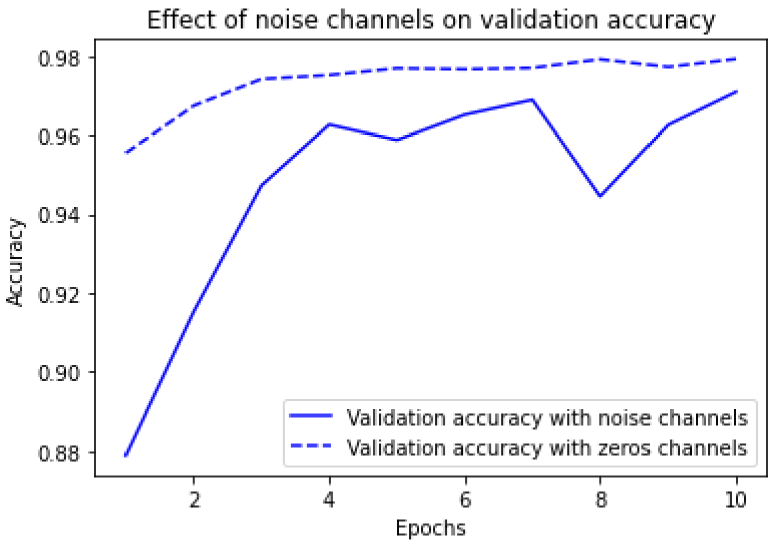
```
Epoch 1/10
375/375 [==============================] - 6s 14ms/step - loss: 0.6325 - accuracy: 0
Epoch 2/10
375/375 [==============================] - 5s 14ms/step - loss: 0.2474 - accuracy: 0
Epoch 3/10
375/375 [==============================] - 5s 14ms/step - loss: 0.1585 - accuracy: 0
Epoch 4/10
375/375 [==============================] - 5s 14ms/step - loss: 0.1110 - accuracy: 0
Epoch 5/10
375/375 [==============================] - 5s 14ms/step - loss: 0.0821 - accuracy: 0
Epoch 6/10
375/375 [==============================] - 5s 14ms/step - loss: 0.0589 - accuracy: 0
Epoch 7/10
375/375 [==============================] - 5s 14ms/step - loss: 0.0440 - accuracy: 0
Epoch 8/10
375/375 [==============================] - 5s 14ms/step - loss: 0.0334 - accuracy: 0
Epoch 9/10
375/375 [==============================] - 5s 14ms/step - loss: 0.0268 - accuracy: 0
Epoch 10/10
375/375 [==============================] - 5s 14ms/step - loss: 0.0192 - accuracy: 0
Epoch 1/10
375/375 [==============================] - 5s 14ms/step - loss: 0.2851 - accuracy: 0
Epoch 2/10
375/375 [==============================] - 5s 13ms/step - loss: 0.1184 - accuracy: 0
Epoch 3/10
375/375 [==============================] - 5s 13ms/step - loss: 0.0782 - accuracy: 0
Epoch 4/10
375/375 [==============================] - 5s 13ms/step - loss: 0.0555 - accuracy: 0
Epoch 5/10
375/375 [==============================] - 5s 13ms/step - loss: 0.0409 - accuracy: 0
Epoch 6/10
375/375 [==============================] - 5s 13ms/step - loss: 0.0309 - accuracy: 0
Epoch 7/10
375/375 [==============================] - 5s 13ms/step - loss: 0.0238 - accuracy: 0
Epoch 8/10
375/375 [==============================] - 5s 13ms/step - loss: 0.0180 - accuracy: 0
Epoch 9/10
375/375 [==============================] - 5s 13ms/step - loss: 0.0134 - accuracy: 0
```

```
Epoch 10/10
375/375 [==============================] - 5s 13ms/step - loss: 0.0100 - accuracy: 0
```

Plotting a validation accuracy comparison

```
import matplotlib.pyplot as plt
val_acc_noise = history_noise.history["val_accuracy"]
val_acc_zeros = history_zeros.history["val_accuracy"]
epochs = range(1, 11)
plt.plot(epochs, val_acc_noise, "b-",
         label="Validation accuracy with noise channels")
plt.plot(epochs, val_acc_zeros, "b--",
         label="Validation accuracy with zeros channels")
plt.title("Effect of noise channels on validation accuracy")
plt.xlabel("Epochs")
plt.ylabel("Accuracy")
plt.legend()
```

```
<matplotlib.legend.Legend at 0x7f697a39b0d0>
```



The nature of generalization in deep learning

Fitting a MNIST model with randomly shuffled labels

```
(train_images, train_labels), _ = mnist.load_data()
train_images = train_images.reshape((60000, 28 * 28))
train_images = train_images.astype("float32") / 255

random_train_labels = train_labels[:]
np.random.shuffle(random_train_labels)

model = keras.Sequential([
    layers.Dense(512, activation="relu"),
    layers.Dense(10, activation="softmax")
])
```

```
model.compile(optimizer="rmsprop",
              loss="sparse_categorical_crossentropy",
              metrics=["accuracy"])
model.fit(train_images, random_train_labels,
          epochs=100,
          batch_size=128,
          validation_split=0.2)
```

```
375/375 [==============================] - 3s 8ms/step - loss: 1.7924 - accuracy:
Epoch 21/100
375/375 [==============================] - 3s 8ms/step - loss: 1.7580 - accuracy:
Epoch 22/100
375/375 [==============================] - 3s 8ms/step - loss: 1.7281 - accuracy:
Epoch 23/100
375/375 [==============================] - 3s 8ms/step - loss: 1.6995 - accuracy:
Epoch 24/100
375/375 [==============================] - 3s 8ms/step - loss: 1.6686 - accuracy:
Epoch 25/100
375/375 [==============================] - 3s 8ms/step - loss: 1.6398 - accuracy:
Epoch 26/100
375/375 [==============================] - 3s 8ms/step - loss: 1.6094 - accuracy:
Epoch 27/100
375/375 [==============================] - 3s 7ms/step - loss: 1.5824 - accuracy:
Epoch 28/100
375/375 [==============================] - 3s 8ms/step - loss: 1.5566 - accuracy:
Epoch 29/100
375/375 [==============================] - 3s 8ms/step - loss: 1.5309 - accuracy:
Epoch 30/100
375/375 [==============================] - 3s 8ms/step - loss: 1.5039 - accuracy:
Epoch 31/100
375/375 [==============================] - 3s 8ms/step - loss: 1.4800 - accuracy:
Epoch 32/100
375/375 [==============================] - 3s 8ms/step - loss: 1.4533 - accuracy:
Epoch 33/100
375/375 [==============================] - 3s 8ms/step - loss: 1.4297 - accuracy:
Epoch 34/100
375/375 [==============================] - 3s 8ms/step - loss: 1.4068 - accuracy:
Epoch 35/100
375/375 [==============================] - 3s 8ms/step - loss: 1.3832 - accuracy:
Epoch 36/100
375/375 [==============================] - 3s 8ms/step - loss: 1.3597 - accuracy:
Epoch 37/100
375/375 [==============================] - 3s 8ms/step - loss: 1.3394 - accuracy:
Epoch 38/100
375/375 [==============================] - 3s 8ms/step - loss: 1.3167 - accuracy:
Epoch 39/100
375/375 [==============================] - 3s 8ms/step - loss: 1.2952 - accuracy:
Epoch 40/100
375/375 [==============================] - 3s 8ms/step - loss: 1.2764 - accuracy:
Epoch 41/100
375/375 [==============================] - 3s 8ms/step - loss: 1.2554 - accuracy:
Epoch 42/100
375/375 [==============================] - 3s 8ms/step - loss: 1.2345 - accuracy:
Epoch 43/100
375/375 [==============================] - 3s 8ms/step - loss: 1.2135 - accuracy:
Epoch 44/100
375/375 [==============================] - 3s 7ms/step - loss: 1.1956 - accuracy:
Epoch 45/100
375/375 [==============================] - 3s 8ms/step - loss: 1.1773 - accuracy:
Epoch 46/100
```

```
375/375 [==============================] - 3s 8ms/step - loss: 1.1589 - accuracy:
Epoch 47/100
375/375 [==============================] - 3s 8ms/step - loss: 1.1408 - accuracy:
Epoch 48/100
375/375 [==============================] - 3s 8ms/step - loss: 1.1239 - accuracy:
Epoch 49/100
```

The manifold hypothesis

Interpolation as a source of generalization

Why deep learning works

Training data is paramount

Evaluating machine-learning models

Training, validation, and test sets

Simple hold-out validation

K-fold validation

Iterated K-fold validation with shuffling

Beating a common-sense baseline

Things to keep in mind about model evaluation

Improving model fit

Tuning key gradient descent parameters

Training a MNIST model with an incorrectly high learning rate

```
(train_images, train_labels), _ = mnist.load_data()
train_images = train_images.reshape((60000, 28 * 28))
train_images = train_images.astype("float32") / 255

model = keras.Sequential([
    layers.Dense(512, activation="relu"),
    layers.Dense(10, activation="softmax")
])
model.compile(optimizer=keras.optimizers.RMSprop(1.),
              loss="sparse_categorical_crossentropy",
              metrics=["accuracy"])
model.fit(train_images, train_labels,
          epochs=10,
          batch_size=128,
          validation_split=0.2)
```

```
Epoch 1/10
375/375 [==============================] - 3s 8ms/step - loss: 1075.7363 - accuracy:
Epoch 2/10
375/375 [==============================] - 3s 8ms/step - loss: 4.2604 - accuracy: 0.
Epoch 3/10
375/375 [==============================] - 3s 8ms/step - loss: 2.9843 - accuracy: 0.
Epoch 4/10
375/375 [==============================] - 3s 8ms/step - loss: 3.6225 - accuracy: 0.
Epoch 5/10
375/375 [==============================] - 3s 8ms/step - loss: 4.7912 - accuracy: 0.
Epoch 6/10
375/375 [==============================] - 3s 8ms/step - loss: 3.1377 - accuracy: 0.
Epoch 7/10
375/375 [==============================] - 3s 8ms/step - loss: 2.6649 - accuracy: 0.
Epoch 8/10
375/375 [==============================] - 3s 8ms/step - loss: 2.5033 - accuracy: 0.
Epoch 9/10
375/375 [==============================] - 3s 8ms/step - loss: 2.5594 - accuracy: 0.
Epoch 10/10
375/375 [==============================] - 3s 8ms/step - loss: 2.6952 - accuracy: 0.
<keras.callbacks.History at 0x7f6973774450>
```

The same model with a more appropriate learning rate

```
model = keras.Sequential([
    layers.Dense(512, activation="relu"),
    layers.Dense(10, activation="softmax")
])
model.compile(optimizer=keras.optimizers.RMSprop(1e-2),
              loss="sparse_categorical_crossentropy",
              metrics=["accuracy"])
model.fit(train_images, train_labels,
          epochs=10,
          batch_size=128,
          validation_split=0.2)
```

```
Epoch 1/10
375/375 [==============================] - 3s 8ms/step - loss: 0.3734 - accuracy: 0.
Epoch 2/10
```

```
375/375 [==============================] - 3s 8ms/step - loss: 0.1435 - accuracy: 0.
Epoch 3/10
375/375 [==============================] - 3s 8ms/step - loss: 0.1121 - accuracy: 0.
Epoch 4/10
375/375 [==============================] - 3s 8ms/step - loss: 0.0956 - accuracy: 0.
Epoch 5/10
375/375 [==============================] - 3s 8ms/step - loss: 0.0844 - accuracy: 0.
Epoch 6/10
375/375 [==============================] - 3s 8ms/step - loss: 0.0784 - accuracy: 0.
Epoch 7/10
375/375 [==============================] - 3s 8ms/step - loss: 0.0736 - accuracy: 0.
Epoch 8/10
375/375 [==============================] - 3s 8ms/step - loss: 0.0693 - accuracy: 0.
Epoch 9/10
375/375 [==============================] - 3s 8ms/step - loss: 0.0615 - accuracy: 0.
Epoch 10/10
375/375 [==============================] - 3s 8ms/step - loss: 0.0613 - accuracy: 0.
<keras.callbacks.History at 0x7f6973600050>
```

Leveraging better architecture priors

Increasing model capacity

A simple logistic regression on MNIST

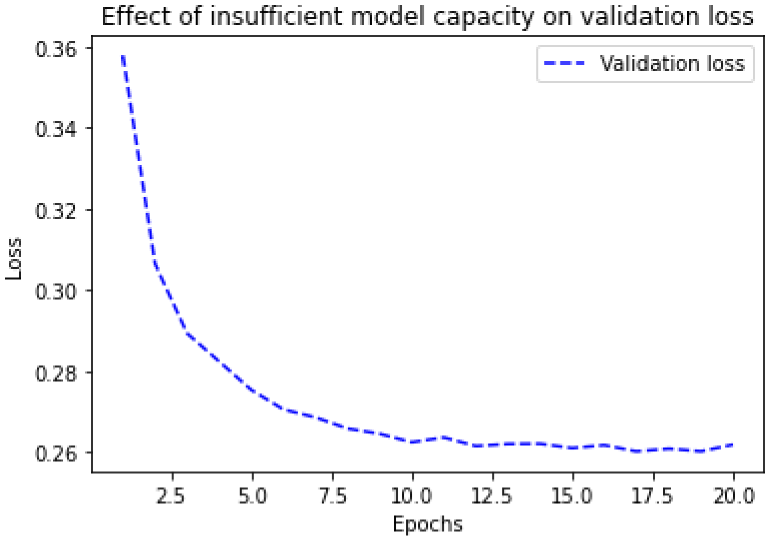
```
model = keras.Sequential([layers.Dense(10, activation="softmax")])
model.compile(optimizer="rmsprop",
              loss="sparse_categorical_crossentropy",
              metrics=["accuracy"])
history_small_model = model.fit(
    train_images, train_labels,
    epochs=20,
    batch_size=128,
    validation_split=0.2)
```

```
Epoch 1/20
375/375 [==============================] - 1s 2ms/step - loss: 0.6608 - accuracy: 0.
Epoch 2/20
375/375 [==============================] - 1s 2ms/step - loss: 0.3510 - accuracy: 0.
Epoch 3/20
375/375 [==============================] - 1s 2ms/step - loss: 0.3154 - accuracy: 0.
Epoch 4/20
375/375 [==============================] - 1s 2ms/step - loss: 0.2995 - accuracy: 0.
Epoch 5/20
375/375 [==============================] - 1s 2ms/step - loss: 0.2896 - accuracy: 0.
Epoch 6/20
375/375 [==============================] - 1s 2ms/step - loss: 0.2832 - accuracy: 0.
Epoch 7/20
375/375 [==============================] - 1s 2ms/step - loss: 0.2779 - accuracy: 0.
Epoch 8/20
375/375 [==============================] - 1s 2ms/step - loss: 0.2745 - accuracy: 0.
Epoch 9/20
375/375 [==============================] - 1s 2ms/step - loss: 0.2714 - accuracy: 0.
```

```
Epoch 10/20
375/375 [==============================] - 1s 2ms/step - loss: 0.2685 - accuracy: 0.
Epoch 11/20
375/375 [==============================] - 1s 2ms/step - loss: 0.2667 - accuracy: 0.
Epoch 12/20
375/375 [==============================] - 1s 2ms/step - loss: 0.2646 - accuracy: 0.
Epoch 13/20
375/375 [==============================] - 1s 2ms/step - loss: 0.2631 - accuracy: 0.
Epoch 14/20
375/375 [==============================] - 1s 2ms/step - loss: 0.2614 - accuracy: 0.
Epoch 15/20
375/375 [==============================] - 1s 2ms/step - loss: 0.2605 - accuracy: 0.
Epoch 16/20
375/375 [==============================] - 1s 2ms/step - loss: 0.2593 - accuracy: 0.
Epoch 17/20
375/375 [==============================] - 1s 2ms/step - loss: 0.2581 - accuracy: 0.
Epoch 18/20
375/375 [==============================] - 1s 2ms/step - loss: 0.2573 - accuracy: 0.
Epoch 19/20
375/375 [==============================] - 1s 2ms/step - loss: 0.2563 - accuracy: 0.
Epoch 20/20
375/375 [==============================] - 1s 2ms/step - loss: 0.2556 - accuracy: 0.
```

```
import matplotlib.pyplot as plt
val_loss = history_small_model.history["val_loss"]
epochs = range(1, 21)
plt.plot(epochs, val_loss, "b--",
         label="Validation loss")
plt.title("Effect of insufficient model capacity on validation loss")
plt.xlabel("Epochs")
plt.ylabel("Loss")
plt.legend()
```

```
<matplotlib.legend.Legend at 0x7f696a873190>
```



```
model = keras.Sequential([
    layers.Dense(96, activation="relu"),
    layers.Dense(96, activation="relu"),
    layers.Dense(10, activation="softmax"),
])
```

```
model.compile(optimizer="rmsprop",
              loss="sparse_categorical_crossentropy",
              metrics=["accuracy"])
history_large_model = model.fit(
    train_images, train_labels,
    epochs=20,
    batch_size=128,
    validation_split=0.2)
```

```
Epoch 1/20
375/375 [==============================] - 2s 4ms/step - loss: 0.3675 - accuracy: 0.
Epoch 2/20
375/375 [==============================] - 1s 4ms/step - loss: 0.1655 - accuracy: 0.
Epoch 3/20
375/375 [==============================] - 1s 3ms/step - loss: 0.1147 - accuracy: 0.
Epoch 4/20
375/375 [==============================] - 1s 4ms/step - loss: 0.0877 - accuracy: 0.
Epoch 5/20
375/375 [==============================] - 1s 4ms/step - loss: 0.0710 - accuracy: 0.
Epoch 6/20
375/375 [==============================] - 1s 4ms/step - loss: 0.0576 - accuracy: 0.
Epoch 7/20
375/375 [==============================] - 1s 4ms/step - loss: 0.0500 - accuracy: 0.
Epoch 8/20
375/375 [==============================] - 1s 3ms/step - loss: 0.0416 - accuracy: 0.
Epoch 9/20
375/375 [==============================] - 1s 4ms/step - loss: 0.0359 - accuracy: 0.
Epoch 10/20
375/375 [==============================] - 1s 3ms/step - loss: 0.0297 - accuracy: 0.
Epoch 11/20
375/375 [==============================] - 1s 4ms/step - loss: 0.0247 - accuracy: 0.
Epoch 12/20
375/375 [==============================] - 1s 3ms/step - loss: 0.0230 - accuracy: 0.
Epoch 13/20
375/375 [==============================] - 1s 3ms/step - loss: 0.0191 - accuracy: 0.
Epoch 14/20
375/375 [==============================] - 1s 4ms/step - loss: 0.0150 - accuracy: 0.
Epoch 15/20
375/375 [==============================] - 1s 3ms/step - loss: 0.0135 - accuracy: 0.
Epoch 16/20
375/375 [==============================] - 1s 3ms/step - loss: 0.0111 - accuracy: 0.
Epoch 17/20
375/375 [==============================] - 1s 3ms/step - loss: 0.0100 - accuracy: 0.
Epoch 18/20
375/375 [==============================] - 1s 3ms/step - loss: 0.0079 - accuracy: 0.
Epoch 19/20
375/375 [==============================] - 1s 3ms/step - loss: 0.0073 - accuracy: 0.
Epoch 20/20
375/375 [==============================] - 1s 4ms/step - loss: 0.0069 - accuracy: 0.
```

Improving generalization

Dataset curation

Feature engineering

Using early stopping

Regularizing your model

Reducing the network's size

Original model

```
from tensorflow.keras.datasets import imdb
(train_data, train_labels), _ = imdb.load_data(num_words=10000)

def vectorize_sequences(sequences, dimension=10000):
    results = np.zeros((len(sequences), dimension))
    for i, sequence in enumerate(sequences):
        results[i, sequence] = 1.
    return results
train_data = vectorize_sequences(train_data)

model = keras.Sequential([
    layers.Dense(16, activation="relu"),
    layers.Dense(16, activation="relu"),
    layers.Dense(1, activation="sigmoid")
])
model.compile(optimizer="rmsprop",
              loss="binary_crossentropy",
              metrics=["accuracy"])
history_original = model.fit(train_data, train_labels,
                             epochs=20, batch_size=512, validation_split=0.4)
```

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/im
17465344/17464789 [==============================] - 0s 0us/step
17473536/17464789 [==============================] - 0s 0us/step
Epoch 1/20
30/30 [==============================] - 1s 33ms/step - loss: 0.5090 - accuracy: 0.7
Epoch 2/20
30/30 [==============================] - 1s 25ms/step - loss: 0.3036 - accuracy: 0.9
Epoch 3/20
30/30 [==============================] - 1s 26ms/step - loss: 0.2251 - accuracy: 0.9
Epoch 4/20
30/30 [==============================] - 1s 26ms/step - loss: 0.1768 - accuracy: 0.9
Epoch 5/20
30/30 [==============================] - 1s 26ms/step - loss: 0.1445 - accuracy: 0.9
Epoch 6/20
30/30 [==============================] - 1s 26ms/step - loss: 0.1188 - accuracy: 0.9
Epoch 7/20
30/30 [==============================] - 1s 27ms/step - loss: 0.0992 - accuracy: 0.9
Epoch 8/20
30/30 [==============================] - 1s 26ms/step - loss: 0.0819 - accuracy: 0.9
Epoch 9/20
```

```
30/30 [==============================] - 1s 27ms/step - loss: 0.0675 - accuracy: 0.9
Epoch 10/20
30/30 [==============================] - 1s 26ms/step - loss: 0.0560 - accuracy: 0.9
Epoch 11/20
30/30 [==============================] - 1s 26ms/step - loss: 0.0483 - accuracy: 0.9
Epoch 12/20
30/30 [==============================] - 1s 26ms/step - loss: 0.0355 - accuracy: 0.9
Epoch 13/20
30/30 [==============================] - 1s 26ms/step - loss: 0.0299 - accuracy: 0.9
Epoch 14/20
30/30 [==============================] - 1s 25ms/step - loss: 0.0247 - accuracy: 0.9
Epoch 15/20
30/30 [==============================] - 1s 26ms/step - loss: 0.0207 - accuracy: 0.9
Epoch 16/20
30/30 [==============================] - 1s 26ms/step - loss: 0.0134 - accuracy: 0.9
Epoch 17/20
30/30 [==============================] - 1s 25ms/step - loss: 0.0123 - accuracy: 0.9
Epoch 18/20
30/30 [==============================] - 1s 26ms/step - loss: 0.0110 - accuracy: 0.9
Epoch 19/20
30/30 [==============================] - 1s 27ms/step - loss: 0.0053 - accuracy: 0.9
Epoch 20/20
30/30 [==============================] - 1s 26ms/step - loss: 0.0056 - accuracy: 0.9
```

Version of the model with lower capacity

```
model = keras.Sequential([
    layers.Dense(4, activation="relu"),
    layers.Dense(4, activation="relu"),
    layers.Dense(1, activation="sigmoid")
])
model.compile(optimizer="rmsprop",
              loss="binary_crossentropy",
              metrics=["accuracy"])
history_smaller_model = model.fit(
    train_data, train_labels,
    epochs=20, batch_size=512, validation_split=0.4)
```

```
Epoch 1/20
30/30 [==============================] - 1s 32ms/step - loss: 0.6496 - accuracy: 0.7
Epoch 2/20
30/30 [==============================] - 1s 25ms/step - loss: 0.5606 - accuracy: 0.8
Epoch 3/20
30/30 [==============================] - 1s 26ms/step - loss: 0.4684 - accuracy: 0.8
Epoch 4/20
30/30 [==============================] - 1s 26ms/step - loss: 0.3857 - accuracy: 0.9
Epoch 5/20
30/30 [==============================] - 1s 26ms/step - loss: 0.3198 - accuracy: 0.9
Epoch 6/20
30/30 [==============================] - 1s 26ms/step - loss: 0.2700 - accuracy: 0.9
Epoch 7/20
30/30 [==============================] - 1s 26ms/step - loss: 0.2318 - accuracy: 0.9
Epoch 8/20
30/30 [==============================] - 1s 26ms/step - loss: 0.2025 - accuracy: 0.9
Epoch 9/20
30/30 [==============================] - 1s 26ms/step - loss: 0.1791 - accuracy: 0.9
Epoch 10/20
```

```
30/30 [==============================] - 1s 26ms/step - loss: 0.1590 - accuracy: 0.9
Epoch 11/20
30/30 [==============================] - 1s 25ms/step - loss: 0.1434 - accuracy: 0.9
Epoch 12/20
30/30 [==============================] - 1s 25ms/step - loss: 0.1287 - accuracy: 0.9
Epoch 13/20
30/30 [==============================] - 1s 27ms/step - loss: 0.1157 - accuracy: 0.9
Epoch 14/20
30/30 [==============================] - 1s 27ms/step - loss: 0.1019 - accuracy: 0.9
Epoch 15/20
30/30 [==============================] - 1s 26ms/step - loss: 0.0906 - accuracy: 0.9
Epoch 16/20
30/30 [==============================] - 1s 26ms/step - loss: 0.0814 - accuracy: 0.9
Epoch 17/20
30/30 [==============================] - 1s 26ms/step - loss: 0.0732 - accuracy: 0.9
Epoch 18/20
30/30 [==============================] - 1s 26ms/step - loss: 0.0657 - accuracy: 0.9
Epoch 19/20
30/30 [==============================] - 1s 25ms/step - loss: 0.0585 - accuracy: 0.9
Epoch 20/20
30/30 [==============================] - 1s 25ms/step - loss: 0.0526 - accuracy: 0.9
```

Version of the model with higher capacity

```
model = keras.Sequential([
    layers.Dense(512, activation="relu"),
    layers.Dense(512, activation="relu"),
    layers.Dense(1, activation="sigmoid")
])
model.compile(optimizer="rmsprop",
              loss="binary_crossentropy",
              metrics=["accuracy"])
history_larger_model = model.fit(
    train_data, train_labels,
    epochs=20, batch_size=512, validation_split=0.4)
```

```
Epoch 1/20
30/30 [==============================] - 8s 259ms/step - loss: 0.5403 - accuracy: 0.
Epoch 2/20
30/30 [==============================] - 7s 246ms/step - loss: 0.2700 - accuracy: 0.
Epoch 3/20
30/30 [==============================] - 8s 252ms/step - loss: 0.1523 - accuracy: 0.
Epoch 4/20
30/30 [==============================] - 7s 246ms/step - loss: 0.0948 - accuracy: 0.
Epoch 5/20
30/30 [==============================] - 7s 248ms/step - loss: 0.1024 - accuracy: 0.
Epoch 6/20
30/30 [==============================] - 7s 245ms/step - loss: 0.0070 - accuracy: 0.
Epoch 7/20
30/30 [==============================] - 7s 246ms/step - loss: 8.1871e-04 - accuracy
Epoch 8/20
30/30 [==============================] - 7s 246ms/step - loss: 1.2237e-04 - accuracy
Epoch 9/20
30/30 [==============================] - 7s 243ms/step - loss: 2.0267e-05 - accuracy
Epoch 10/20
30/30 [==============================] - 7s 243ms/step - loss: 4.4091e-06 - accuracy
Epoch 11/20
```

```
30/30 [==============================] - 7s 246ms/step - loss: 1.1361e-06 - accuracy
Epoch 12/20
30/30 [==============================] - 7s 243ms/step - loss: 3.4746e-07 - accuracy
Epoch 13/20
30/30 [==============================] - 7s 246ms/step - loss: 1.2056e-07 - accuracy
Epoch 14/20
30/30 [==============================] - 7s 246ms/step - loss: 5.2926e-08 - accuracy
Epoch 15/20
30/30 [==============================] - 7s 242ms/step - loss: 2.9216e-08 - accuracy
Epoch 16/20
30/30 [==============================] - 7s 244ms/step - loss: 1.9402e-08 - accuracy
Epoch 17/20
30/30 [==============================] - 7s 243ms/step - loss: 1.4464e-08 - accuracy
Epoch 18/20
30/30 [==============================] - 7s 241ms/step - loss: 1.1557e-08 - accuracy
Epoch 19/20
30/30 [==============================] - 7s 243ms/step - loss: 9.6278e-09 - accuracy
Epoch 20/20
30/30 [==============================] - 7s 244ms/step - loss: 8.2596e-09 - accuracy
```

Adding weight regularization

Adding L2 weight regularization to the model

```
from tensorflow.keras import regularizers
model = keras.Sequential([
    layers.Dense(16,
                 kernel_regularizer=regularizers.l2(0.002),
                 activation="relu"),
    layers.Dense(16,
                 kernel_regularizer=regularizers.l2(0.002),
                 activation="relu"),
    layers.Dense(1, activation="sigmoid")
])
model.compile(optimizer="rmsprop",
              loss="binary_crossentropy",
              metrics=["accuracy"])
history_l2_reg = model.fit(
    train_data, train_labels,
    epochs=20, batch_size=512, validation_split=0.4)
```

```
Epoch 1/20
30/30 [==============================] - 2s 35ms/step - loss: 0.6350 - accuracy: 0.7
Epoch 2/20
30/30 [==============================] - 1s 27ms/step - loss: 0.4504 - accuracy: 0.8
Epoch 3/20
30/30 [==============================] - 1s 27ms/step - loss: 0.3713 - accuracy: 0.9
Epoch 4/20
30/30 [==============================] - 1s 27ms/step - loss: 0.3321 - accuracy: 0.9
Epoch 5/20
30/30 [==============================] - 1s 27ms/step - loss: 0.3107 - accuracy: 0.9
Epoch 6/20
30/30 [==============================] - 1s 27ms/step - loss: 0.2906 - accuracy: 0.9
Epoch 7/20
```

```
30/30 [==============================] - 1s 27ms/step - loss: 0.2796 - accuracy: 0.9
Epoch 8/20
30/30 [==============================] - 1s 27ms/step - loss: 0.2683 - accuracy: 0.9
Epoch 9/20
30/30 [==============================] - 1s 27ms/step - loss: 0.2598 - accuracy: 0.9
Epoch 10/20
30/30 [==============================] - 1s 28ms/step - loss: 0.2537 - accuracy: 0.9
Epoch 11/20
30/30 [==============================] - 1s 28ms/step - loss: 0.2465 - accuracy: 0.9
Epoch 12/20
30/30 [==============================] - 1s 26ms/step - loss: 0.2443 - accuracy: 0.9
Epoch 13/20
30/30 [==============================] - 1s 27ms/step - loss: 0.2375 - accuracy: 0.9
Epoch 14/20
30/30 [==============================] - 1s 28ms/step - loss: 0.2298 - accuracy: 0.9
Epoch 15/20
30/30 [==============================] - 1s 28ms/step - loss: 0.2323 - accuracy: 0.9
Epoch 16/20
30/30 [==============================] - 1s 28ms/step - loss: 0.2175 - accuracy: 0.9
Epoch 17/20
30/30 [==============================] - 1s 27ms/step - loss: 0.2280 - accuracy: 0.9
Epoch 18/20
30/30 [==============================] - 1s 28ms/step - loss: 0.2159 - accuracy: 0.9
Epoch 19/20
30/30 [==============================] - 1s 28ms/step - loss: 0.2175 - accuracy: 0.9
Epoch 20/20
30/30 [==============================] - 1s 27ms/step - loss: 0.2130 - accuracy: 0.9
```

Different weight regularizers available in Keras

```
from tensorflow.keras import regularizers
regularizers.l1(0.001)
regularizers.l1_l2(l1=0.001, l2=0.001)
```

```
<keras.regularizers.L1L2 at 0x7f696568ead0>
```

Adding dropout

Adding dropout to the IMDB model

```
model = keras.Sequential([
    layers.Dense(16, activation="relu"),
    layers.Dropout(0.5),
    layers.Dense(16, activation="relu"),
    layers.Dropout(0.5),
    layers.Dense(1, activation="sigmoid")
])
model.compile(optimizer="rmsprop",
              loss="binary_crossentropy",
              metrics=["accuracy"])
history_dropout = model.fit(
    train_data, train_labels,
    epochs=20, batch_size=512, validation_split=0.4)
```

```
Epoch 1/20
30/30 [==============================] - 2s 39ms/step - loss: 0.6155 - accuracy: 0.6
Epoch 2/20
30/30 [==============================] - 1s 28ms/step - loss: 0.4993 - accuracy: 0.7
Epoch 3/20
30/30 [==============================] - 1s 27ms/step - loss: 0.4252 - accuracy: 0.8
Epoch 4/20
30/30 [==============================] - 1s 27ms/step - loss: 0.3649 - accuracy: 0.8
Epoch 5/20
30/30 [==============================] - 1s 28ms/step - loss: 0.3194 - accuracy: 0.8
Epoch 6/20
30/30 [==============================] - 1s 26ms/step - loss: 0.2820 - accuracy: 0.9
Epoch 7/20
30/30 [==============================] - 1s 27ms/step - loss: 0.2446 - accuracy: 0.9
Epoch 8/20
30/30 [==============================] - 1s 27ms/step - loss: 0.2210 - accuracy: 0.9
Epoch 9/20
30/30 [==============================] - 1s 27ms/step - loss: 0.1990 - accuracy: 0.9
Epoch 10/20
30/30 [==============================] - 1s 27ms/step - loss: 0.1743 - accuracy: 0.9
Epoch 11/20
30/30 [==============================] - 1s 27ms/step - loss: 0.1580 - accuracy: 0.9
Epoch 12/20
30/30 [==============================] - 1s 27ms/step - loss: 0.1438 - accuracy: 0.9
Epoch 13/20
30/30 [==============================] - 1s 28ms/step - loss: 0.1271 - accuracy: 0.9
Epoch 14/20
30/30 [==============================] - 1s 27ms/step - loss: 0.1193 - accuracy: 0.9
Epoch 15/20
30/30 [==============================] - 1s 27ms/step - loss: 0.1086 - accuracy: 0.9
Epoch 16/20
30/30 [==============================] - 1s 27ms/step - loss: 0.1021 - accuracy: 0.9
Epoch 17/20
30/30 [==============================] - 1s 27ms/step - loss: 0.0944 - accuracy: 0.9
Epoch 18/20
30/30 [==============================] - 1s 27ms/step - loss: 0.0925 - accuracy: 0.9
Epoch 19/20
30/30 [==============================] - 1s 27ms/step - loss: 0.0821 - accuracy: 0.9
Epoch 20/20
30/30 [==============================] - 1s 28ms/step - loss: 0.0795 - accuracy: 0.9
```

● ✕