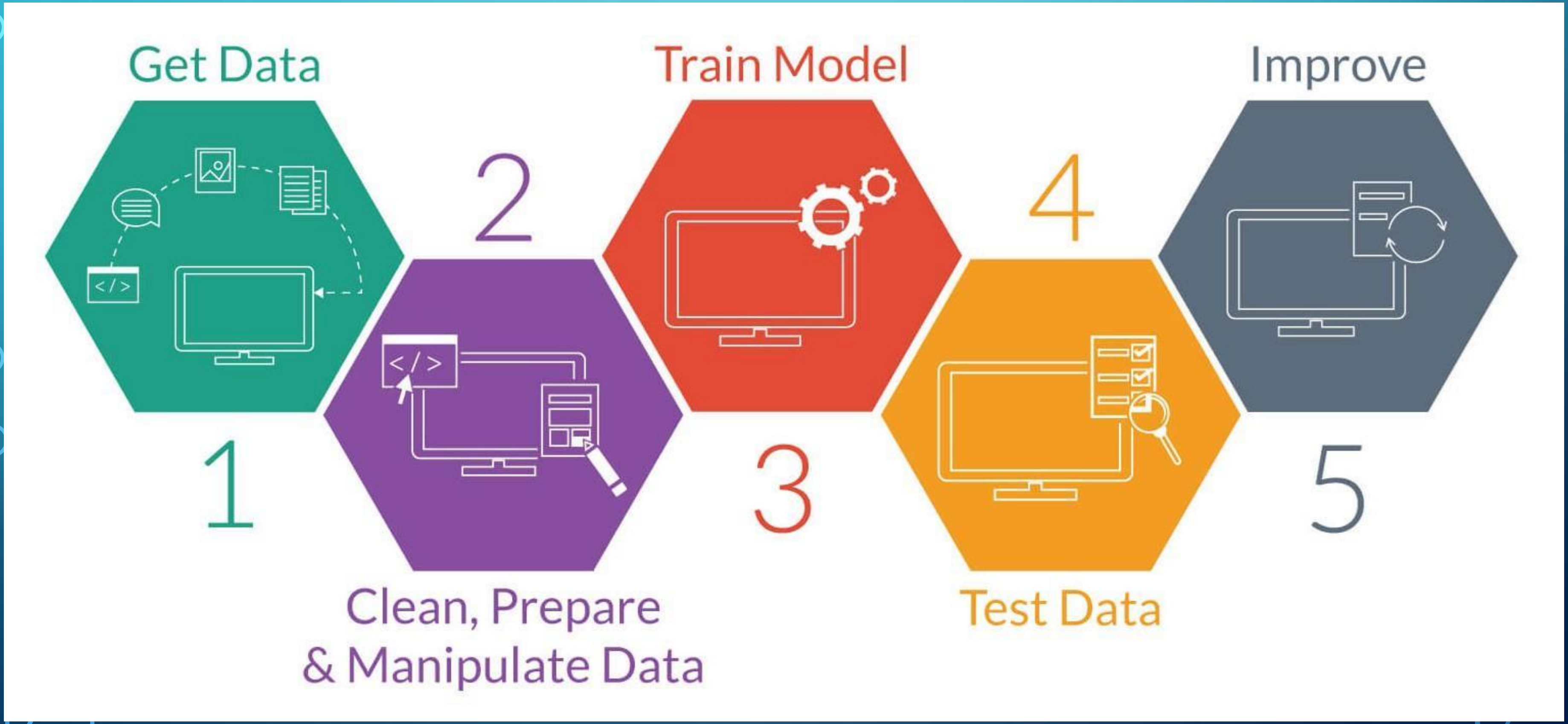
DEEP LEARNING WITH KERAS

BASIC

KERAS MECHANICS



LEARN.PY (1/6)

```
# import packages
from keras.models import Sequential
from keras.layers import Dense
import numpy as np



# global constants and hyper-parameters
MY_EPOCH = 10
MY_BATCH = 16
```

LEARN.PY (2/6)

```
    ####################
    # DATABASE SETTING #
    ####################

# create a random DB
np.set_printoptions(precision = 3)

# generates random floating point number in [0, 1]
data = np.random.random((1000, 100))

# generates random integer in [0, 5]
labels = np.random.randint(6, size = (1000, 1))
```

LEARN.PY (3/6)

```
print('\n== DATABASE SHAPE INFO ==')
print('Input shape = ', data.shape)
print('Output shape = ', labels.shape)

print("\nFirst input:")
print(data[0])
print("\nFirst output:")
print(labels[0])
```

LEARN.PY (4/6)

```
    ###############################
    # MODEL BUILDING AND TRAINING #
    ###############################



# keras sequential model
model = Sequential()

model.add(Dense(32, activation = 'sigmoid', input_dim = 100))
model.add(Dense(1, activation = 'sigmoid'))
model.summary()
model.save('before.h5')
```

KERAS ACTIVATION FUNCTION

- It transforms the summed weighted input of a neuron to an output

- Partial list (research ongoing):
  - softmax
  - elu
  - softplus
  - relu
  - tanh
  - sigmoid
  - hard_sigmoid
  - exponential
  - linear

LEARN.PY (5/6)

```python
# model compilation setting
model.compile(optimizer = 'rmsprop', loss = 'binary_crossentropy',
        metrics = ['accuracy'])


# model training and saving
model.fit(data, labels, epochs = MY_EPOCH, batch_size = MY_BATCH,
        verbose = 1)
model.save('after.h5')
```

KERAS OPTIMIZER FUNCTION

- A tool that searches for parameters that minimize our loss function

- Partial list (research ongoing):
  - SGD (stochastic gradient descent)
  - RMSprop
  - Adagrad
  - Adadelta
  - Adam
  - Adamax
  - Nadam
  - …

KERAS LOSS FUNCTION

- The penalty for a bad prediction

- Partial list (research ongoing):
  - mean_squared_error
  - mean_absolute_error
  - mean_absolute_percentage_error
  - categorical_crossentropy
  - sparse_categorical_crossentropy
  - binary_crossentropy
  - kullback_leibler_divergence
  - poisson
  - cosine_proximity

KERAS METRICS

- A function that is used to judge the performance of your model

- Partial list (research ongoing):
  - accuracy
  - binary_accuracy
  - categorical_accuracy
  - sparse_categorical_accuracy
  - …

- Loss vs. metrics
  - Loss is used during training, metric for evaluation
  - Similar concept: may use interchangeably

LEARN.PY (6/6)

```
    ####################
    # MODEL EVALUATION #
    ####################



score = model.evaluate(data, labels, verbose = 1)
print('Test loss:', score[0])
print('Test accuracy:', score[1])
```