Build a Machine Learning Workflow with Keras Tensorflow 2.0

UNDERSTANDING KERAS MODELS AND LAYERS

Janani Ravi

CO-FOUNDER, LOONYCORN

www.loonycorn.com

Overview

Supervised vs. Unsupervised Learning

Keras and TensorFlow

Sequential models and the functional API in Keras

Saving and loading models

Prerequisites and Course Outline

Prerequisites

Comfortable programming in Python using Jupyter notebooks

Understanding of basic machine learning algorithms

Basic familiarity with deep learning using neural networks

Prerequisite Courses

Understanding Machine Learning with Python

Designing a Machine Learning Model

Getting started with TensorFlow 2.0

Course Outline

Keras and TensorFlow - models and layers

Regression and classification

Image classification models

Unsupervised machine learning with autoencoders

Custom layers and models

Introducing Keras

Keras (Then)

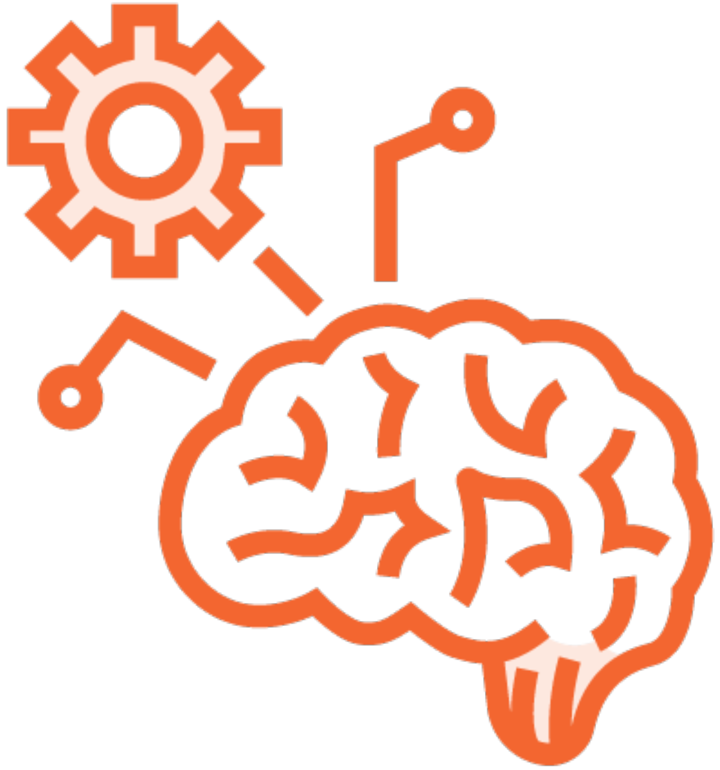
A high-level neural networks API, written in Python and capable of running on top of TensorFlow, CNTK, or Theano.

Keras (Now)

A central part of the tightly-connected TensorFlow 2.0 ecosystem, covering every part of the machine learning workflow.

https://keras.io

TensorFlow and Keras



TensorFlow 2.0 is an open-source machine learning platform

Executes computation graphs which can scale to multiple devices

Keras is an easy-to-use intuitive API for solving machine learning problems

Abstractions and building blocks for creating deep learning models

TensorFlow and Keras

TensorFlow 2.0 includes implementation of Keras API spec

High-level API contained in tf.keras

First-class support for TF-specific functionality

Estimators, pipelines, eager execution

Use tf.keras to build, train, evaluate models

Also use to save/restore models, and leverage GPUs

Supervised and Unsupervised Learning

> “What lies behind us and what lies ahead of us are tiny matters compared to what lives within us”

Henry David Thoreau

Whales: Fish or Mammals?

Mammals

Members of the infraorder *Cetacea*

Fish

Look like fish, swim like fish, move with fish

ML-based Classifier

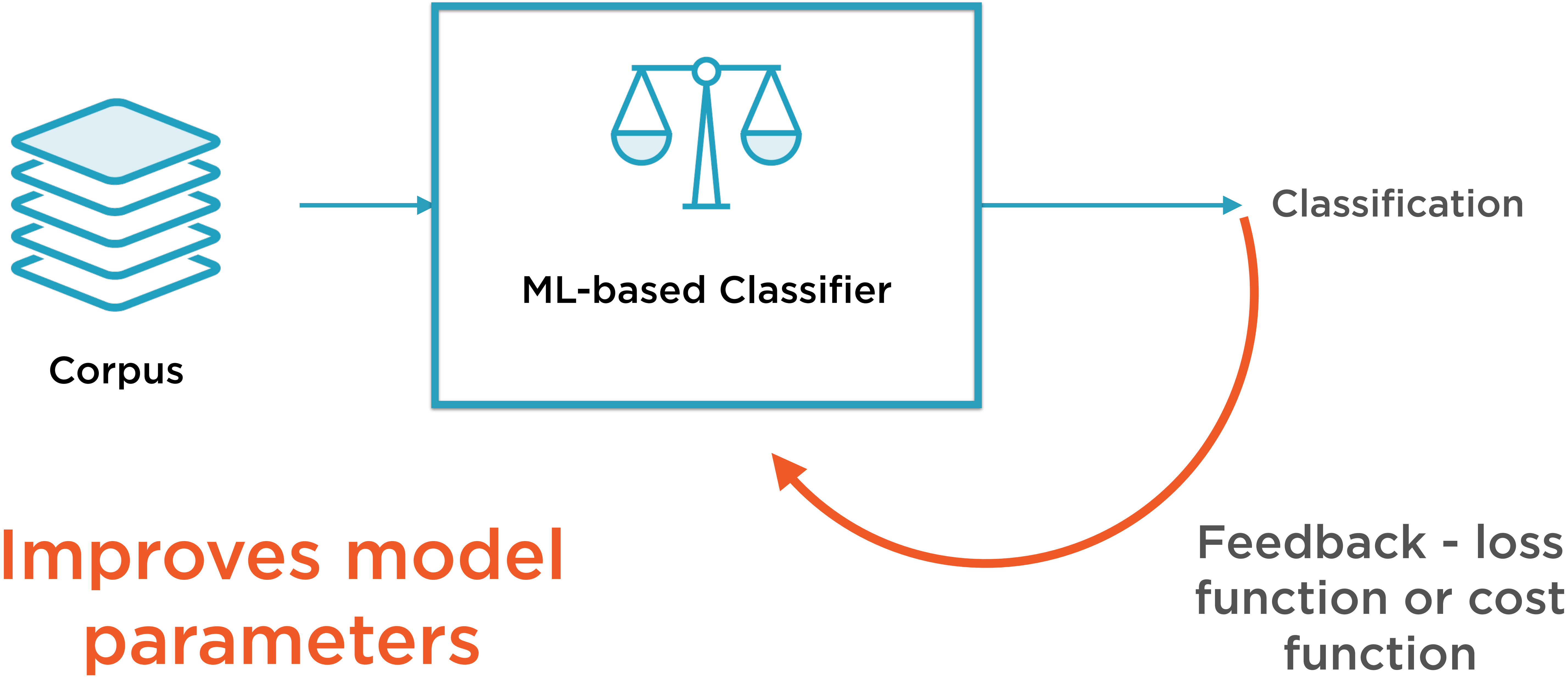
Training

Feed in a large corpus of data classified correctly

Prediction

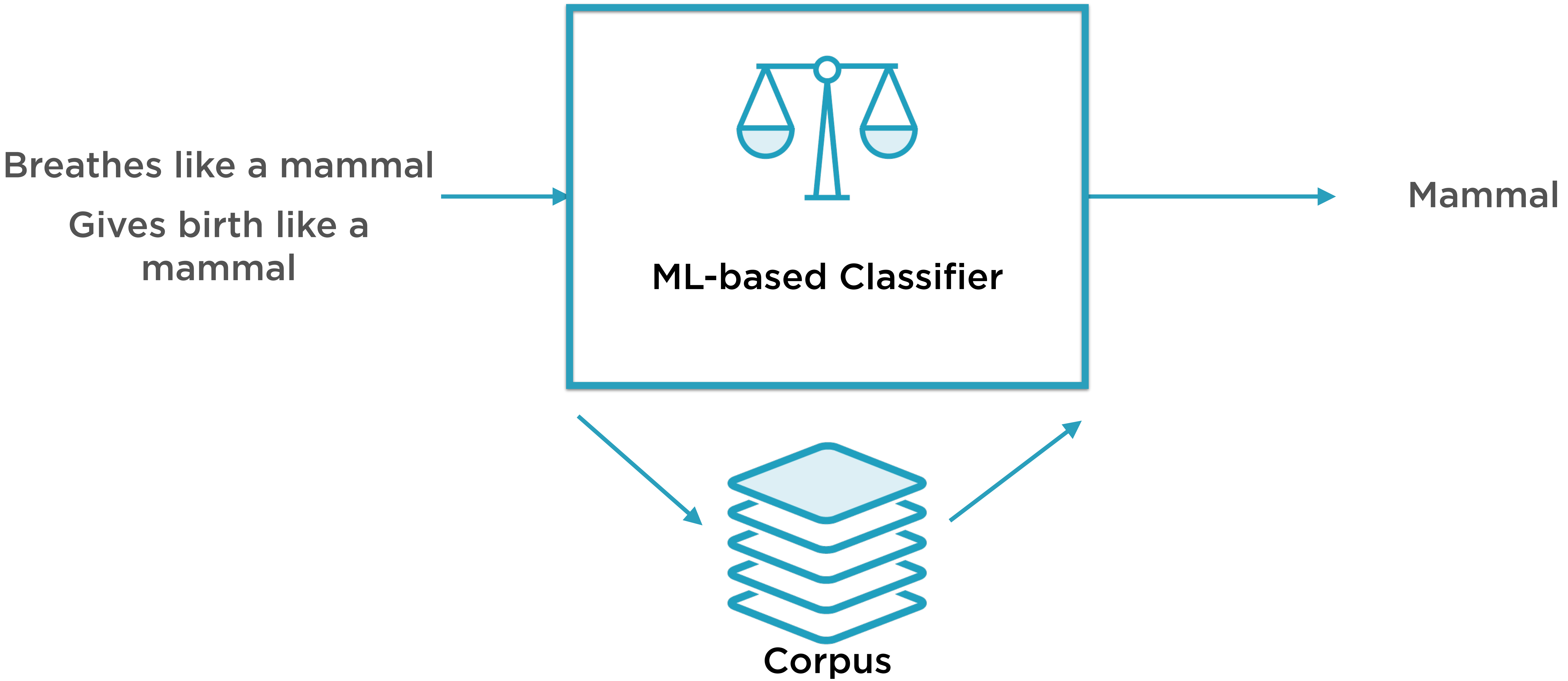
Use it to classify new instances which it has not seen before

Training the ML-based Classifier



Improves model parameters

ML-based Binary Classifier



Breathes like a mammal

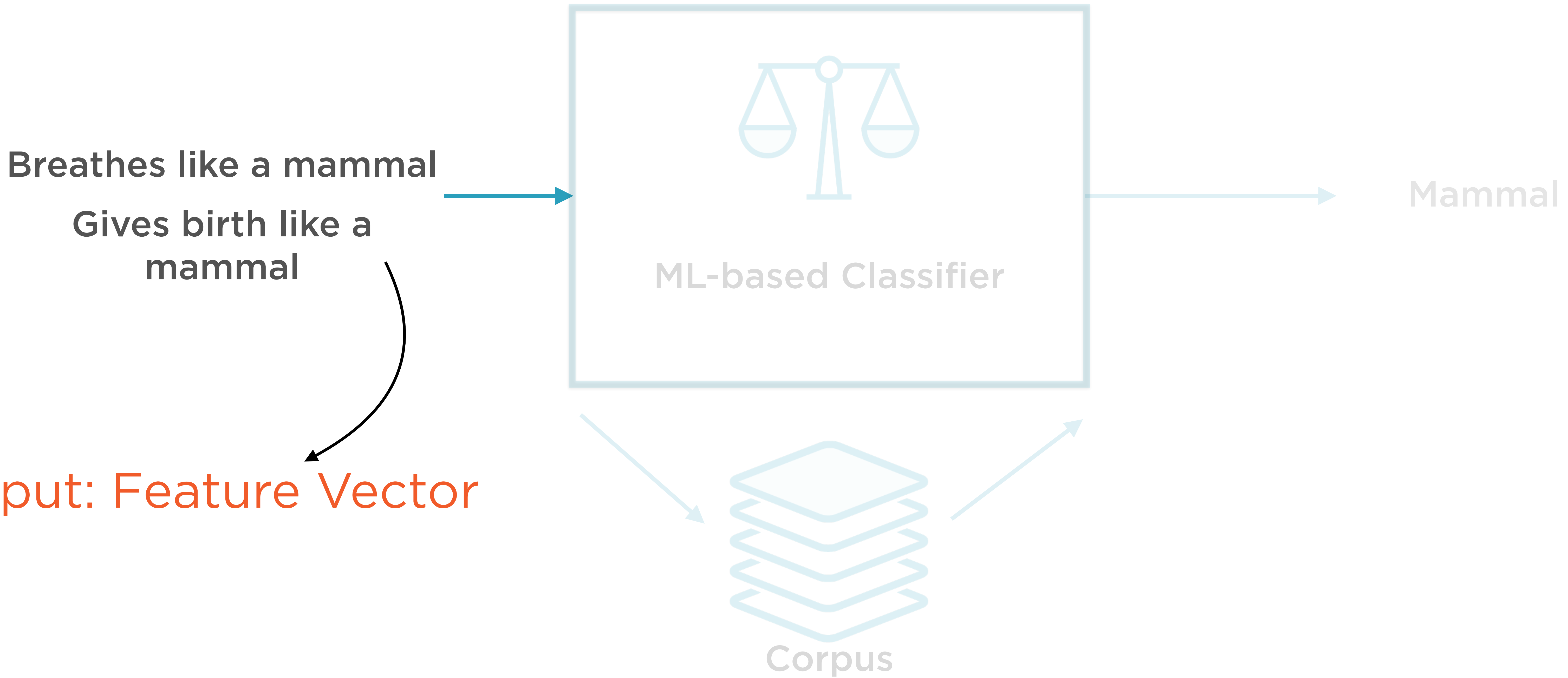
Gives birth like a mammal

ML-based Classifier

Mammal

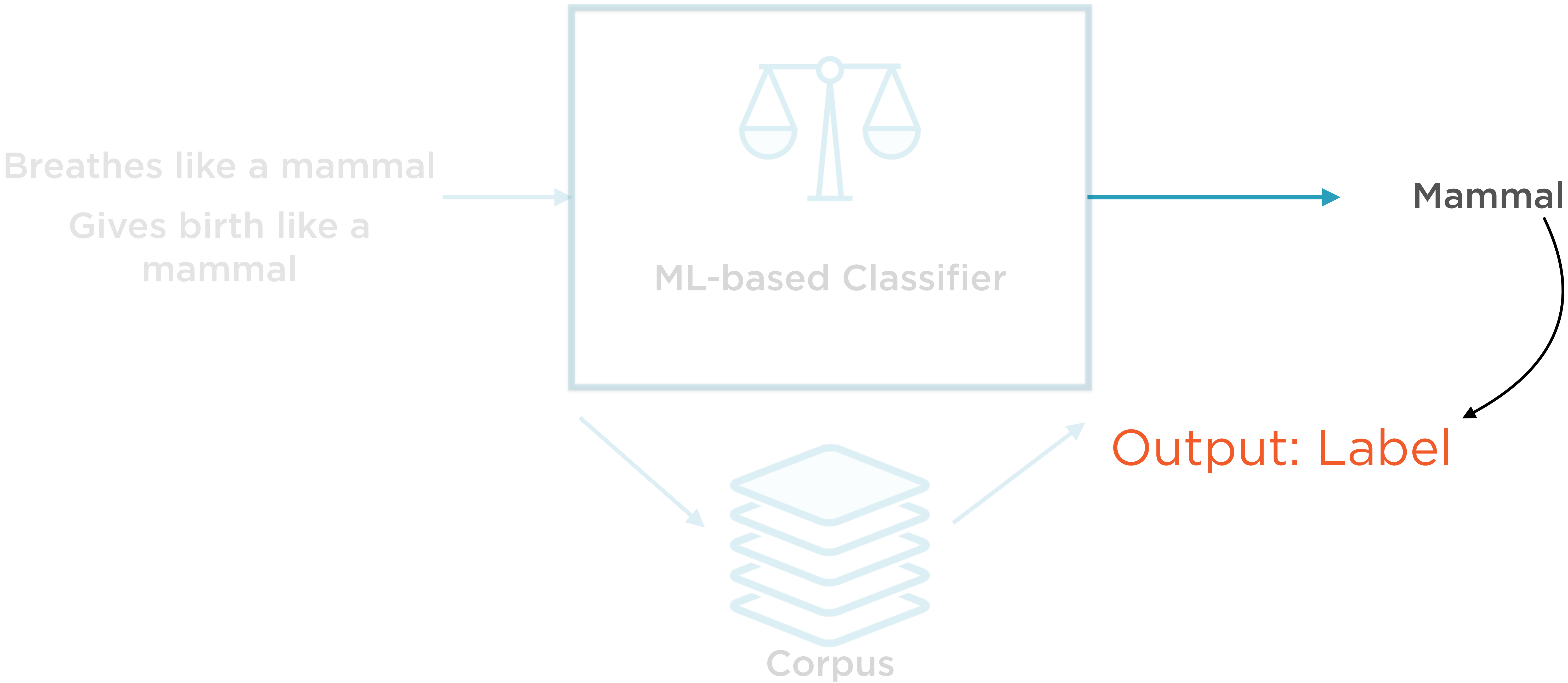
Corpus

ML-based Binary Classifier

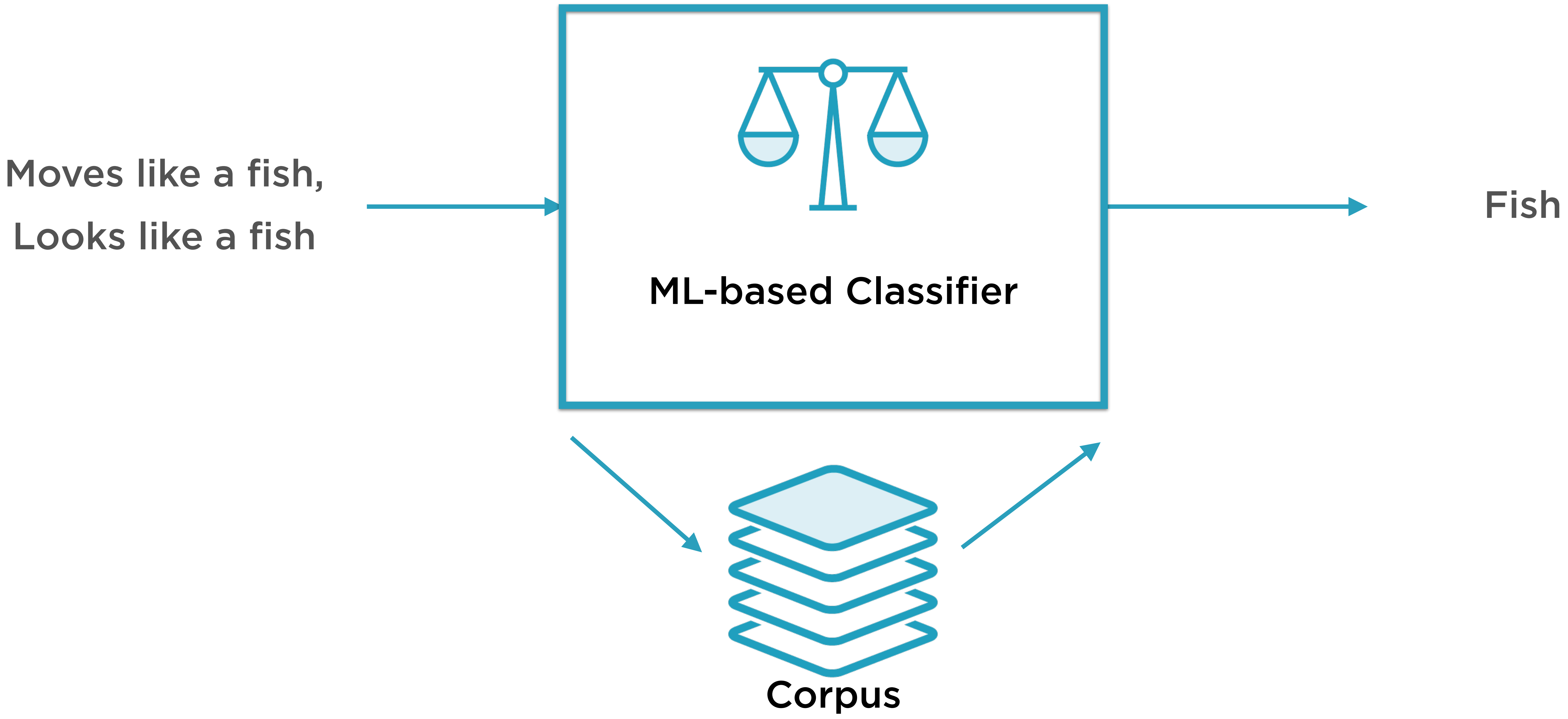


Input: Feature Vector

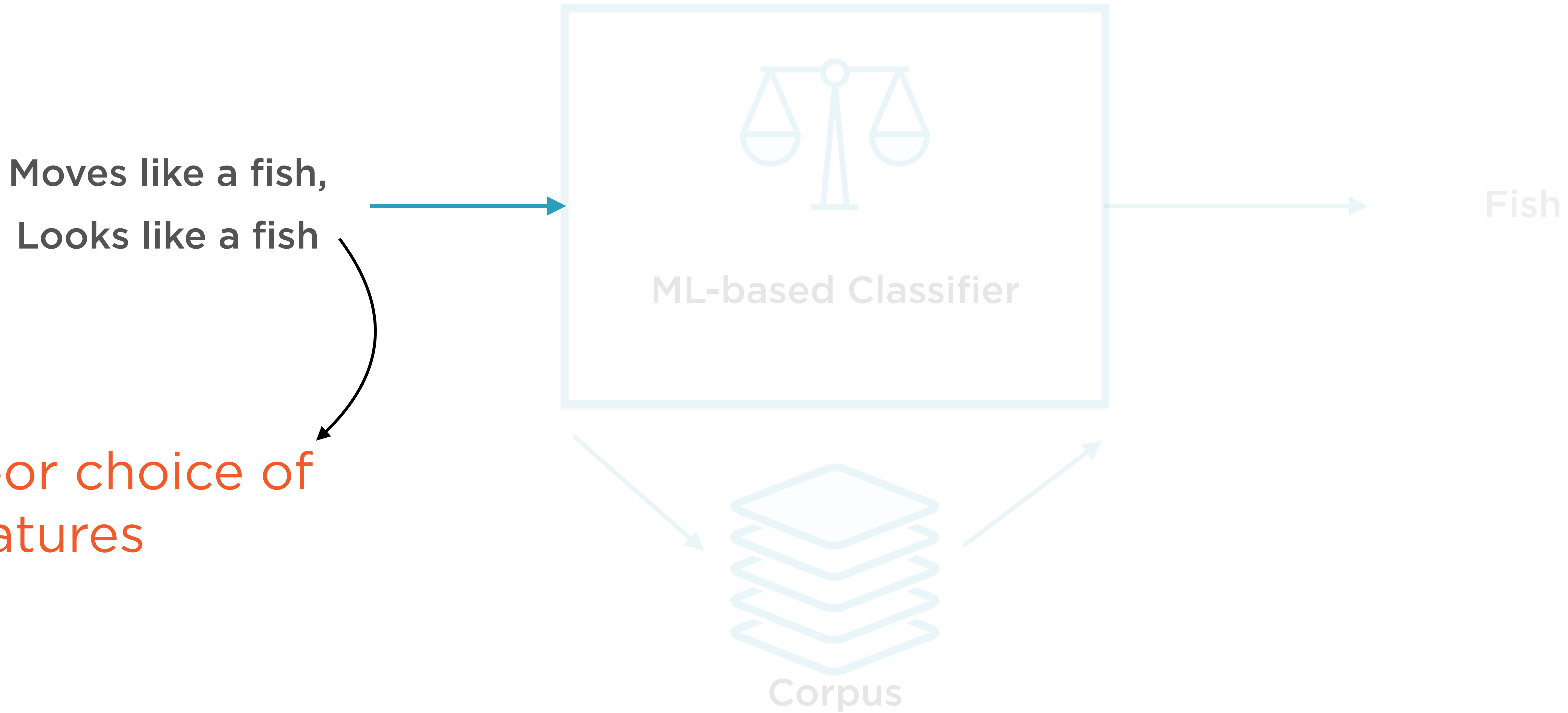
ML-based Binary Classifier



ML-based Binary Classifier



ML-based Binary Classifier

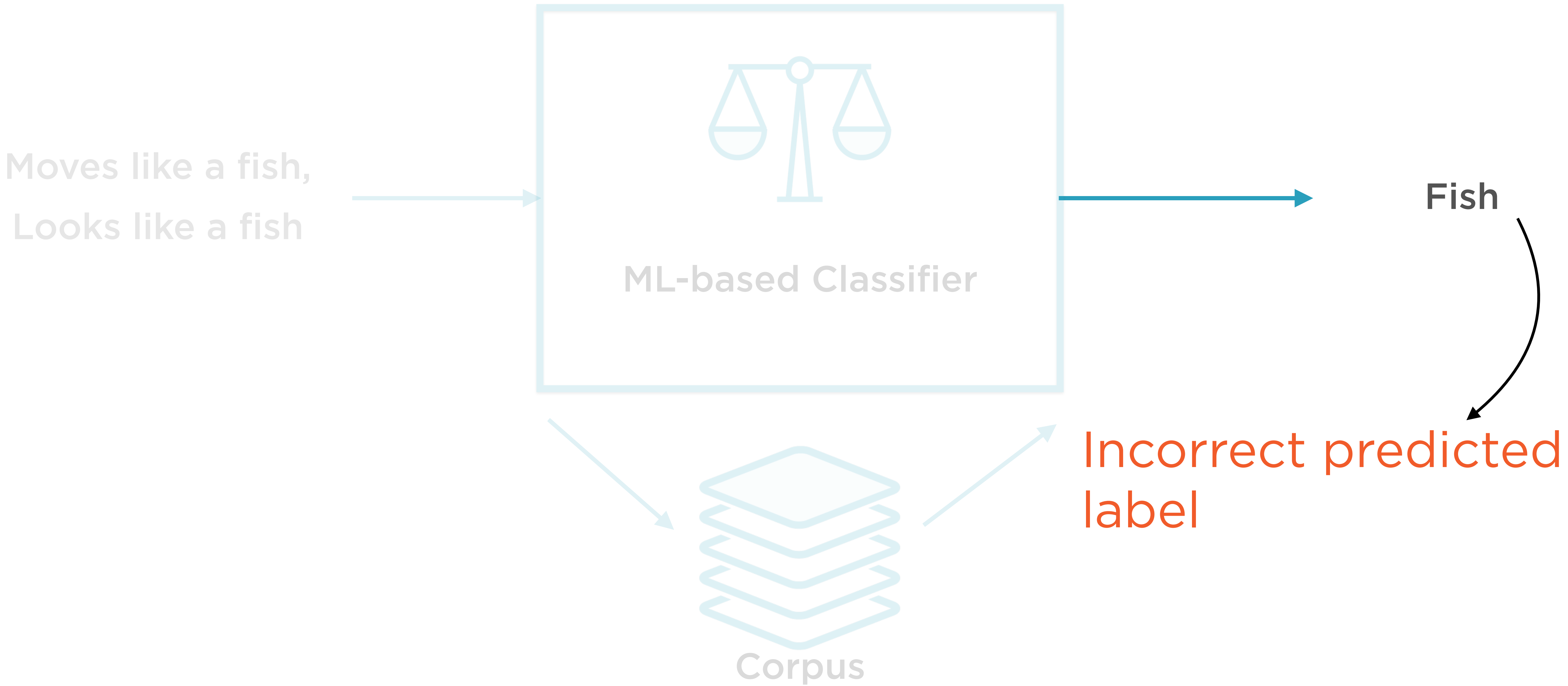


Moves like a fish,

Looks like a fish

Poor choice of features

ML-based Binary Classifier



Incorrect predicted label

x Variables

The attributes that the ML algorithm focuses on are called features

Each data point is a list - or vector - of such features

Thus, the input into an ML algorithm is a feature vector

Feature vectors are usually called the x variables

y Variables

The attributes that the ML algorithm tries to predict are called labels

Types of labels

- categorical (classification)
- continuous (regression)

Labels are usually called the y variables

```
y = f(x)
```

Supervised Machine Learning

Most machine learning algorithms seek to “learn” the function f that links the features and the labels

```
def doSomethingReallyComplicated(x1,x2…):
  …
  …
  …
  return complicatedResult
```

f(x) = doSomethingReallyComplicated(x)

ML algorithms such as neural network can “learn” (reverse-engineer) pretty much anything given the right training data

Everything so far discussed really applied only to **Supervised Learning**

Unsupervised Learning does not have:

- y variables
- a labeled corpus

Types of ML Algorithms

Supervised

Labels associated with the training data is used to correct the algorithm

Unsupervised

The model has to be set up right to learn structure in the data

Supervised Learning

Input variable x and output variable y

Learn the mapping function $y = f(x)$

Approximate the mapping function so for new values of x we can predict y

Use existing dataset to correct our mapping function approximation

Unsupervised Learning

Only have input data x – no output data

Model the underlying structure to learn more about data

Algorithms self discover the patterns and structure in the data

Why Look Within

In Life	In Machine Learning
To be emotionally self-sufficient	To make unlabelled data self-sufficient
To learn what values matter to you	Latent factor analysis
Identify others who share them…	Clustering
..and those who don’t	Anomaly detection
Eliminate what does not matter	Quantization
In general, to train yourself to navigate the outside world	Pre-training for supervised learning problems (classification, regression)

Unsupervised Learning Use-cases

ML Technique	Use-case
To make unlabelled data self-sufficient	Identify photos of a specific individual
Latent factor analysis	Find common drivers of 200 stocks
Clustering	Find relevant document in a corpus
Anomaly detection	Flag fraudulent credit card transactions
Quantization	Compress true color (24 bit) to 8 bit
Pre-training for supervised learning problems (classification, regression)	All of the above!

Unsupervised Learning Use-cases

What	How
To make unlabelled data self-sufficient	Autoencoder
Latent factor analysis	Autoencoder
Clustering	Clustering
Anomaly detection	Autoencoder
Quantization	Clustering
Pre-training for supervised learning problems (classification, regression)	All of the above!

Unsupervised ML Algorithms

Clustering

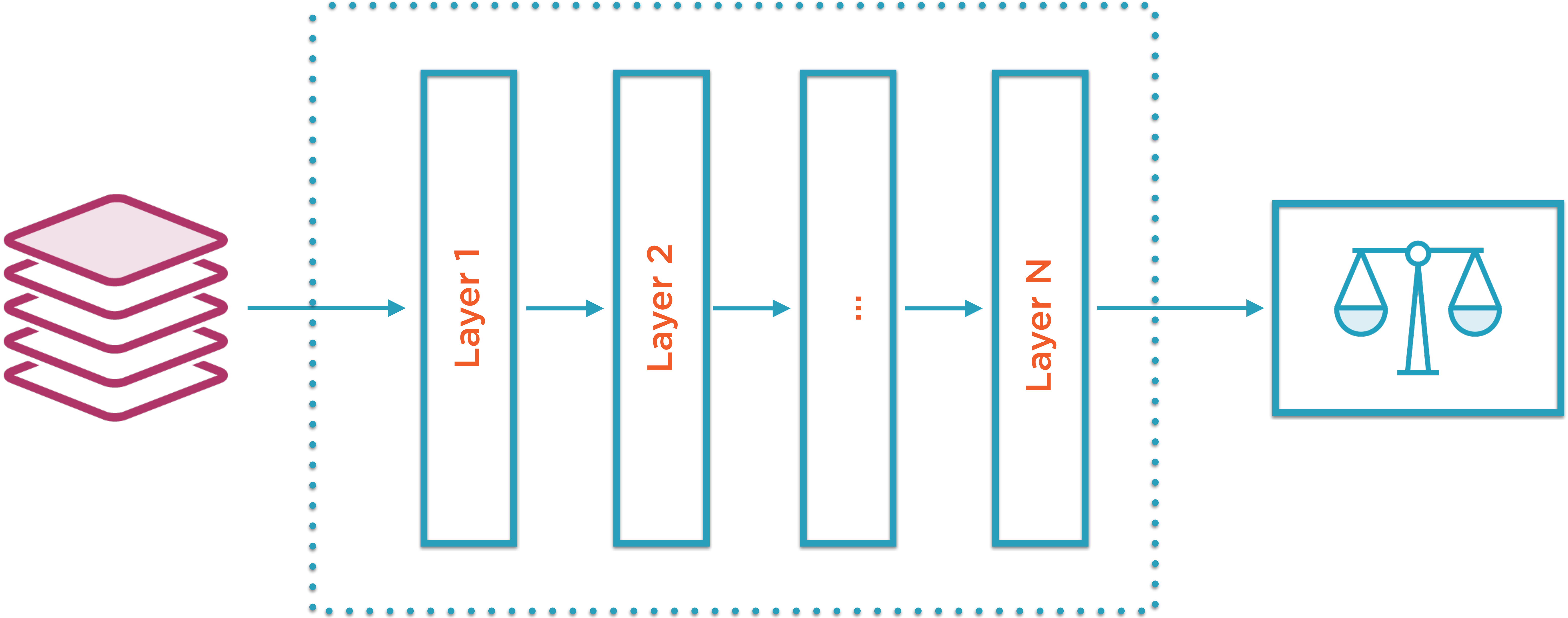
Identify patterns in data items e.g. K-means clustering

Autoencoding

Identify latent factors that drive data e.g. PCA

Keras Building Blocks

Neural Networks

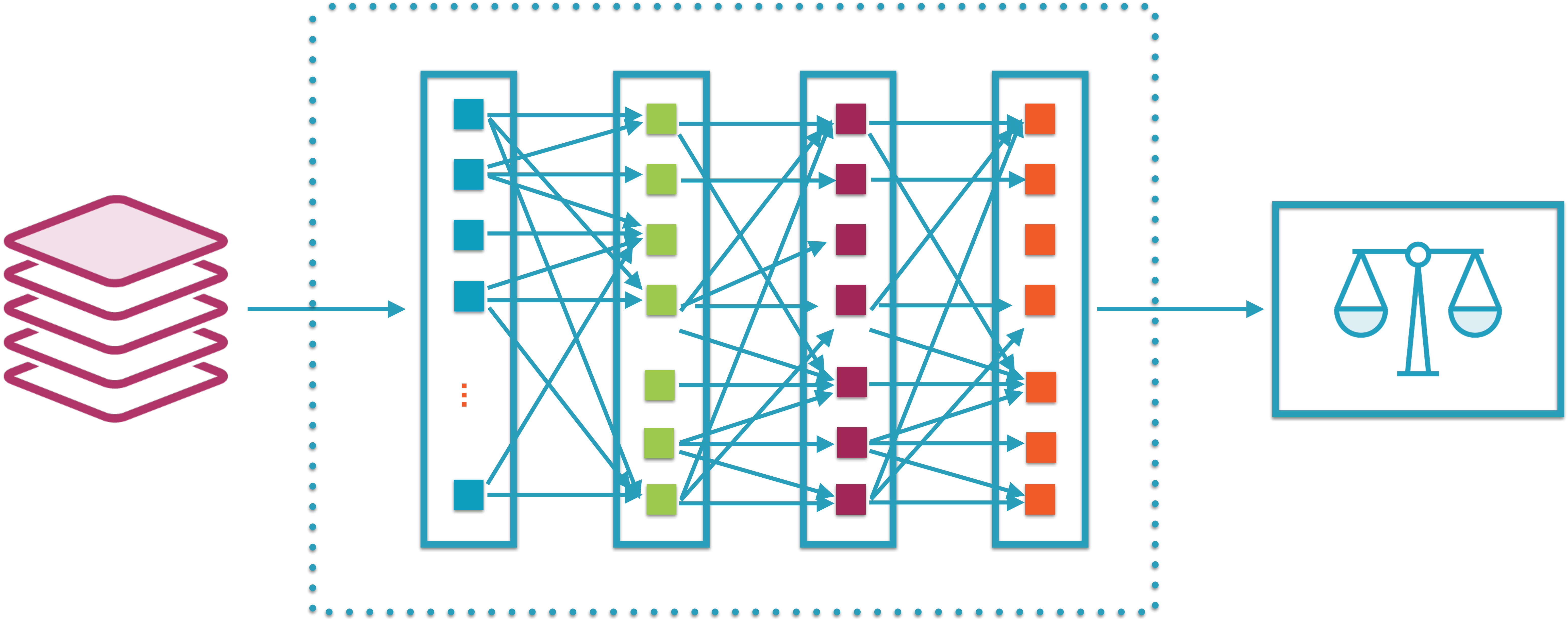


Corpus

Neural networks are deep learning models which are made up of layers

ML-based Classifier

Neural Networks



Corpus

ML-based Classifier

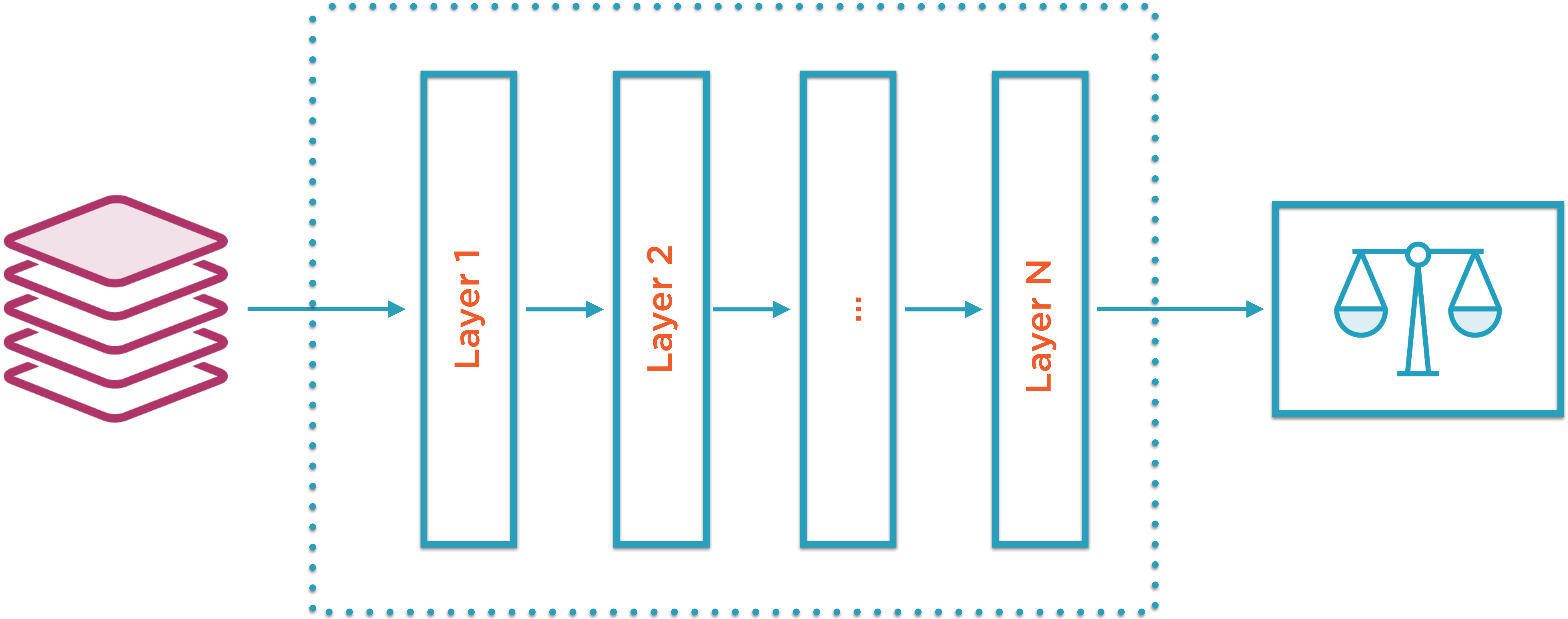
Each layer consists of individual interconnected neurons

Core Data Structures

Layers

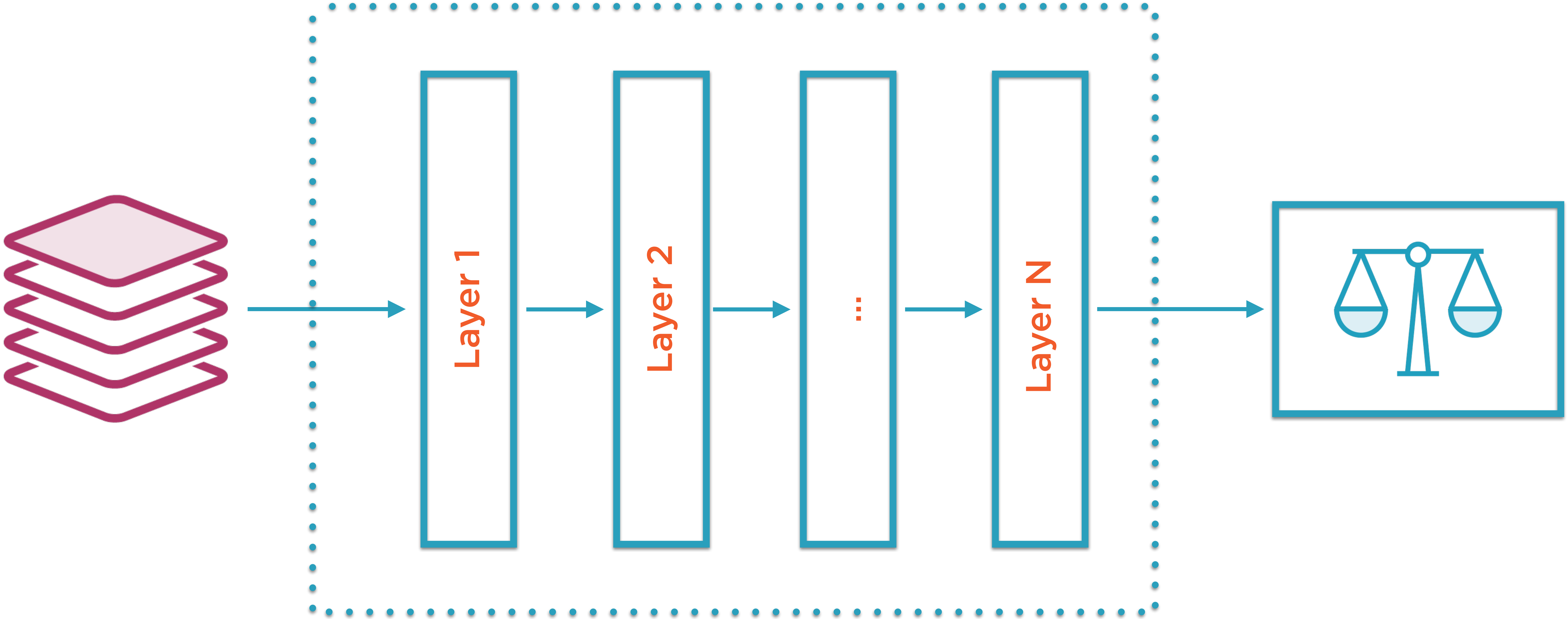
Models

Neural Networks



Layers in a neural network apply transformations on the input data

Neural Networks



Layers come together to create models which are trained and used for prediction

Keras Building Blocks

Sequential Models

Functional APIs

Model Subclassing

Custom Layers

Keras Building Blocks

Sequential Models

Functional APIs

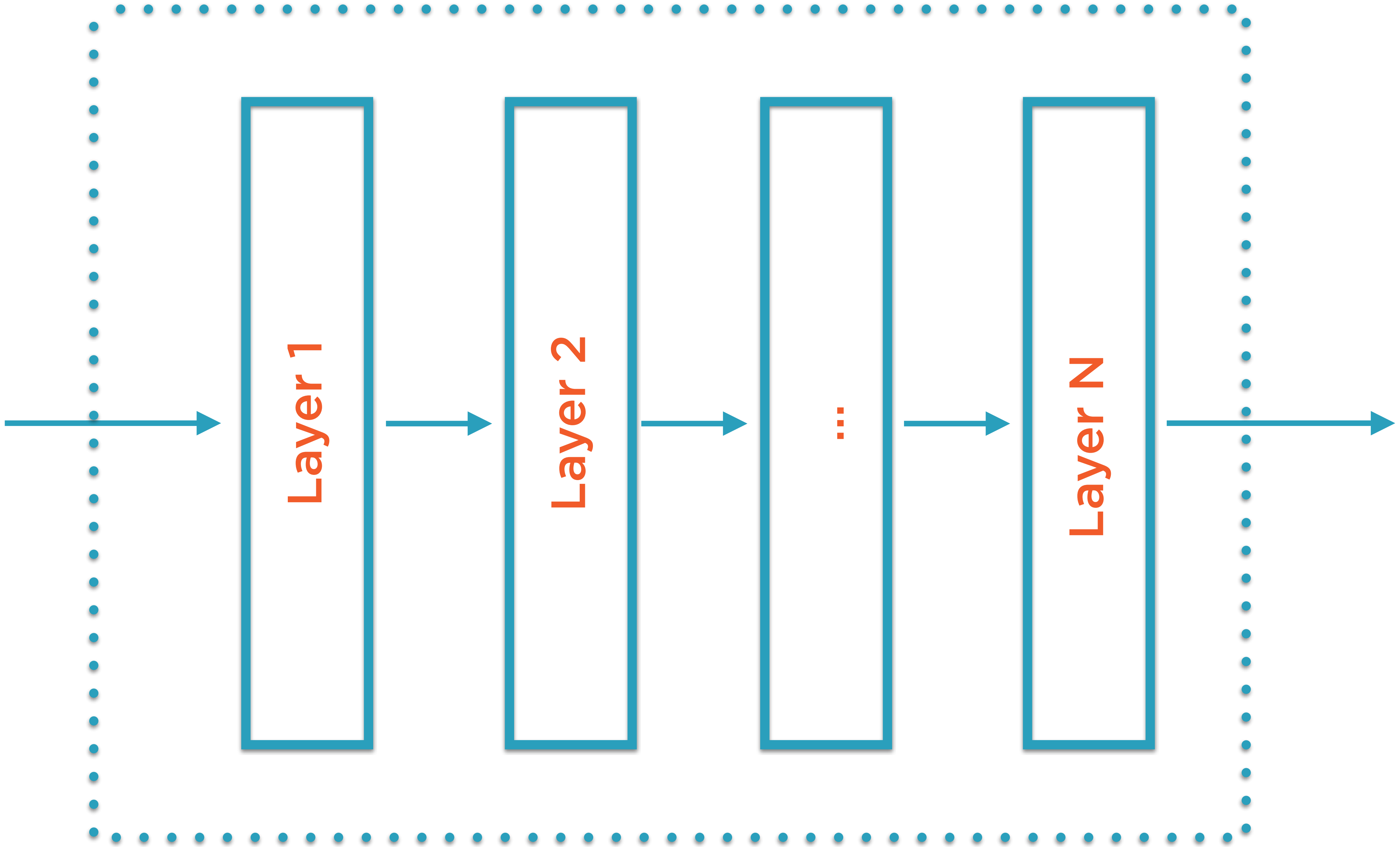
Model Subclassing

Custom Layers

Sequential Models

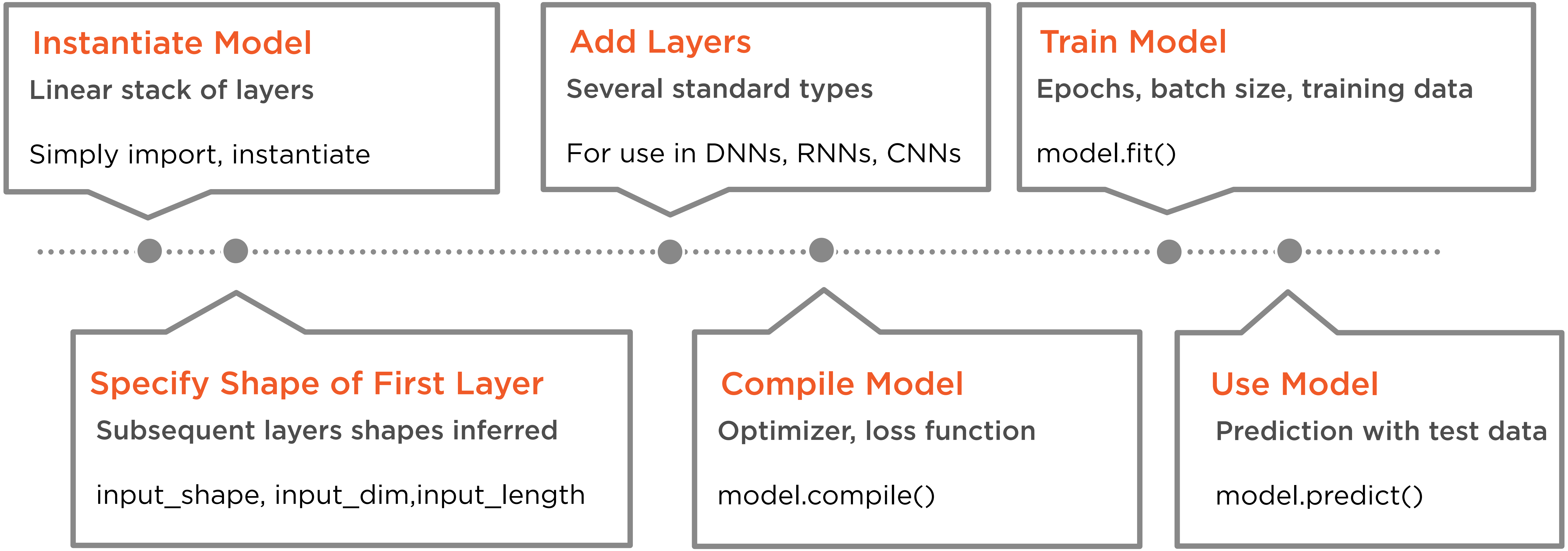
Consist of a simple stack of layers, and so cannot be used to build complex model topologies. APIs contained in tf.keras.Sequential.

Sequential Model



Simply a linear stack of layers

Using Sequential Models in Keras



Instantiate Model

Linear stack of layers

Simply import, instantiate

Specify Shape of First Layer

Subsequent layers shapes inferred

input_shape, input_dim,input_length

Add Layers

Several standard types

For use in DNNs, RNNs, CNNs

Compile Model

Optimizer, loss function

model.compile()

Train Model

Epochs, batch size, training data

model.fit()

Use Model

Prediction with test data

model.predict()

Layers

All layers have common interface

- `layer.get_weights()`
- `layer.set_weights()`
- `layer.get_config()`

Two types of layers

- Single node
- Shared

Layers

Non-shared Layer

- Single input
- All layers in Sequential models

Shared layer

- Multiple inputs
- May occur in Functional API models

Layers

Core

Convolutional

Pooling

Recurrent

Embedding

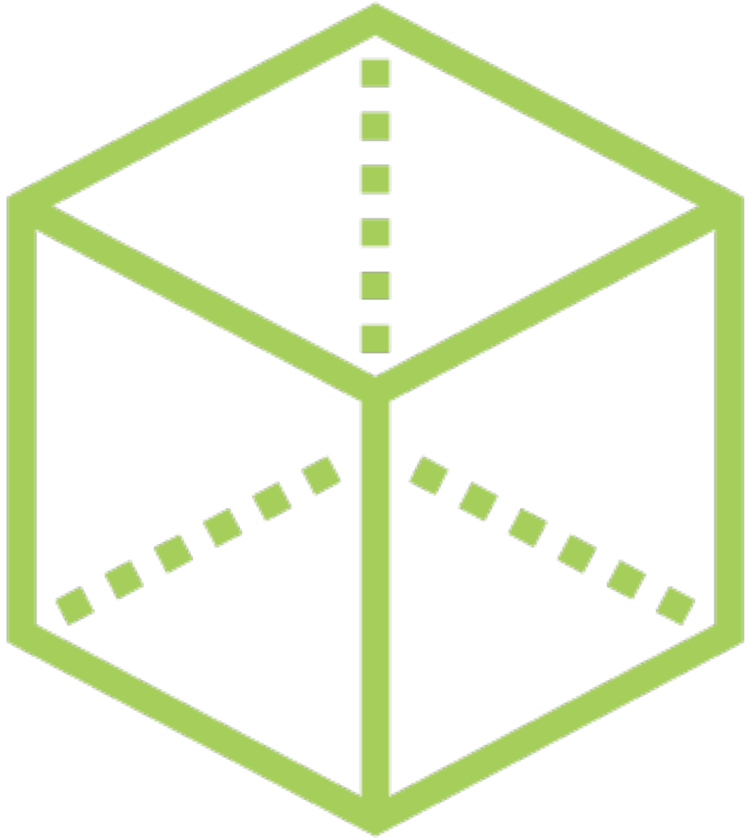
Advanced Activation

Locally connected

...

(Each type has many object types)

Model Compilation



`model.compile()`

Ties model to TF backend

Must specify optimizer and loss function

Several other optional arguments too

Keras Building Blocks

Sequential Models	Functional APIs
Model Subclassing	Custom Layers

Keras Functional API

Used to build complex model topologies that cannot be constructed using the Sequential APIs.

Functional API

Use Functional API for

- Multi-input models
- Multi-output models
- Models with shared layers
- Models with non-sequential data flows

Functional API

The Sequential API is inherently object-oriented

The Functional API is more functional

- Built around models that can be called (like functions)

Functional API: Keras models can be "**called**" on any tensor, just like layers

Functional API

Keras models created using Functional APIs are callable

- Hence the name Functional API

Define tf.keras.Model instance

- Train just like Sequential model

Invoke on input tensors

- To get output tensor

Keras Building Blocks

Sequential Models

Functional APIs

Model Subclassing

Custom Layers

Covered in a later module

Saving and Loading Keras Models

Components of Keras Models

Architecture

Weights

Optimizer

Losses and metrics

Save/Load Operations

Keras allows these components to be saved/loaded

- All at once
- Selectively

Save/Load Operations

Whole Model

Architecture/ Configuration

Weights Only

Whole Model Save/Load

APIs

- model.save
- tf.keras.models.save_model
- tf.keras.models.load_model

Whole Model Save/Load

Architecture

Weight values (learnt during training)

Compilation information if any

Optimizer and state to resume training

Whole Model Save/Load

Two possible formats

- TF SavedModel format (recommended)
- Keras H5 format
    - External losses and metrics not saved
    - Computation graph of custom objects not saved

Architecture/Configuration Save/Load

APIs

- get_config / from_config
- tf.keras.models.model_to_json
- tf.keras.models.model_from_json

Architecture/Configuration Save/Load

Specifies layers and connections

Model needs to be freshly initialized with new weights

Custom objects/layers must override get_config and from_config

Custom functions need not override get_config

Weights-only Save/Load

Formats

- TensorFlow Checkpoint
- HDF5

APIs

- tf.keras.layers.Layer.get_weights
- tf.keras.layers.Layer.set_weights

Weights-only Save/Load

- **Pre-trained models**
  - Pre-trained, so no more training
  - Optimizer state and compilation info are no longer needed
- **Transfer learning**
  - Train new model by reusing state of old model

Demo

Install and set up TensorFlow libraries

Summary

Supervised vs. Unsupervised Learning

Keras and TensorFlow

Sequential models and the functional API in Keras

Saving and loading models

Up Next:

Building Regression and Classification Models