Building Dynamic Computation Graphs

Janani Ravi
CO-FOUNDER, LOONYCORN
www.loonycorn.com

Overview

Static vs. dynamic computation graphs

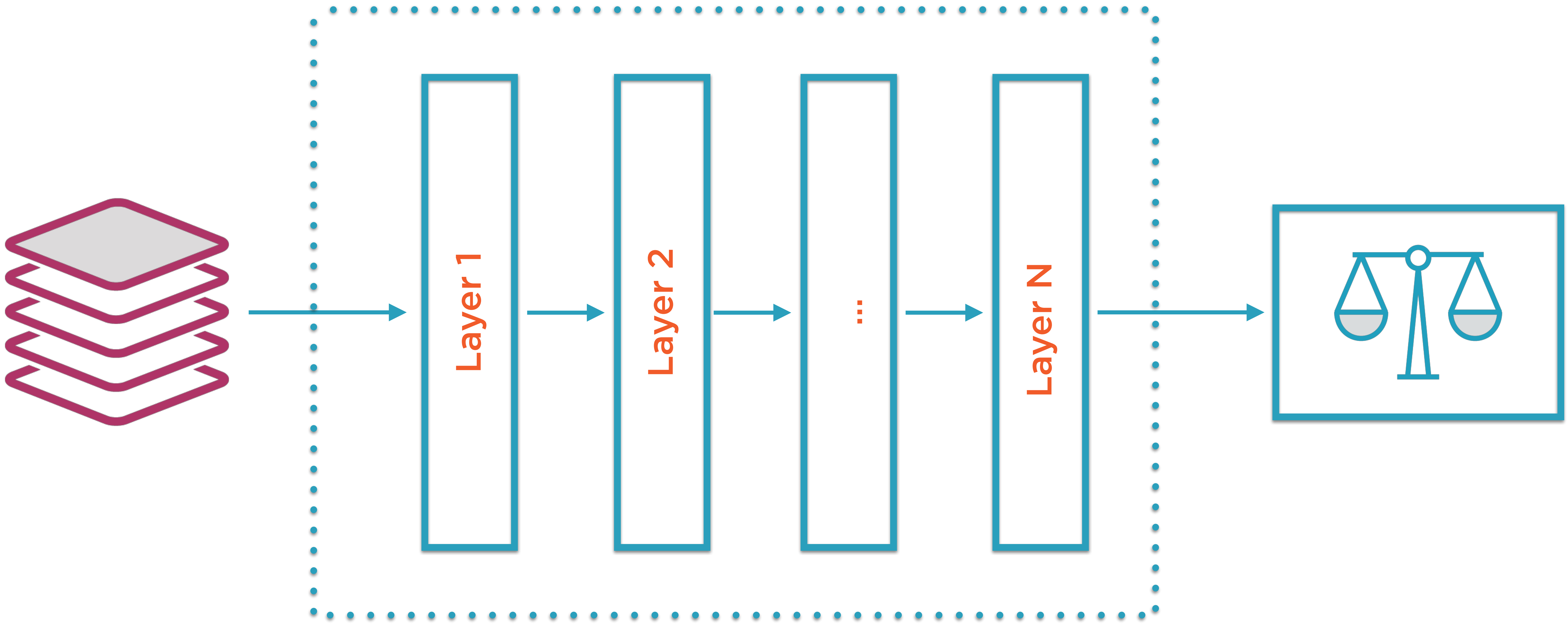
Benefits and drawbacks for dynamic graphs

Building dynamic graphs in PyTorch

Contrast with static graphs built in TensorFlow

Computation Graphs in PyTorch

Neural Networks

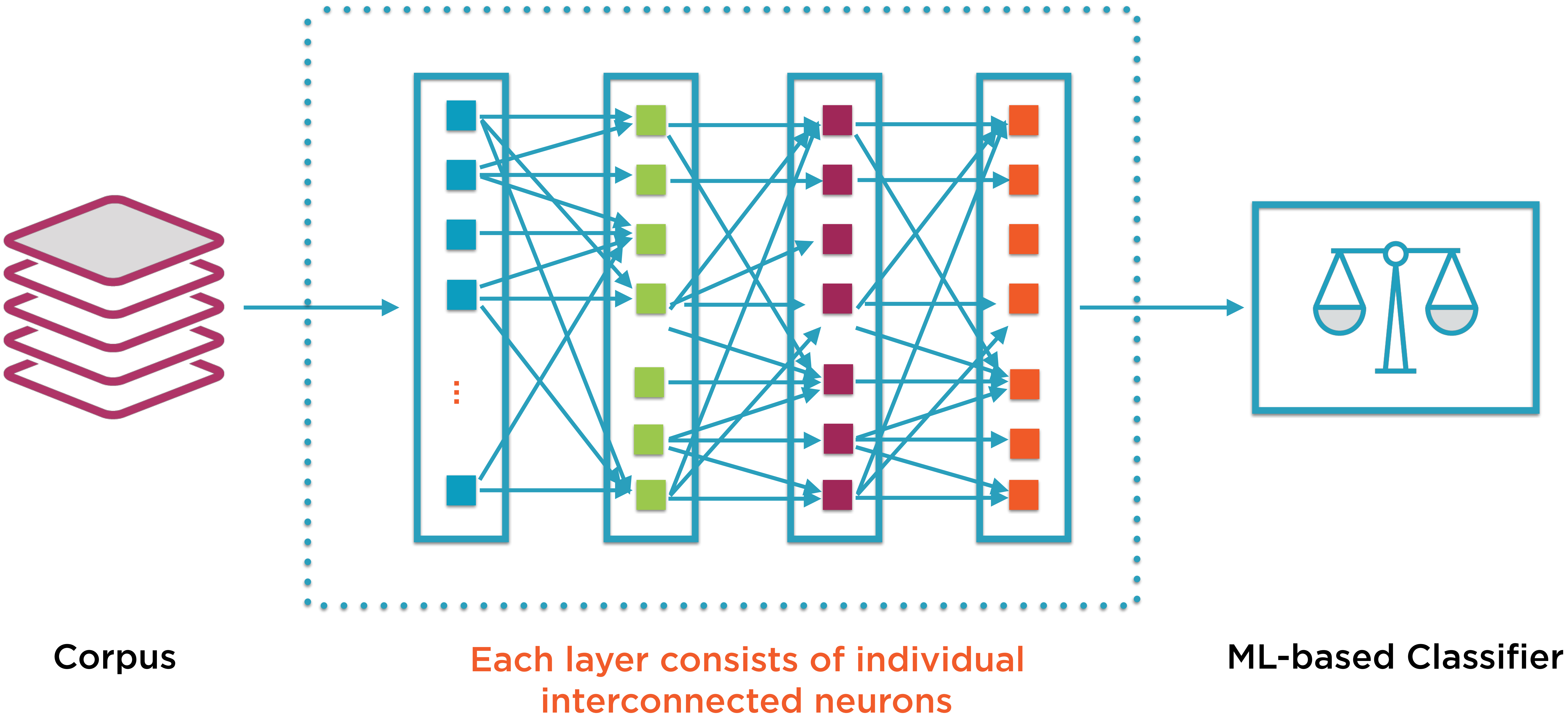


Corpus

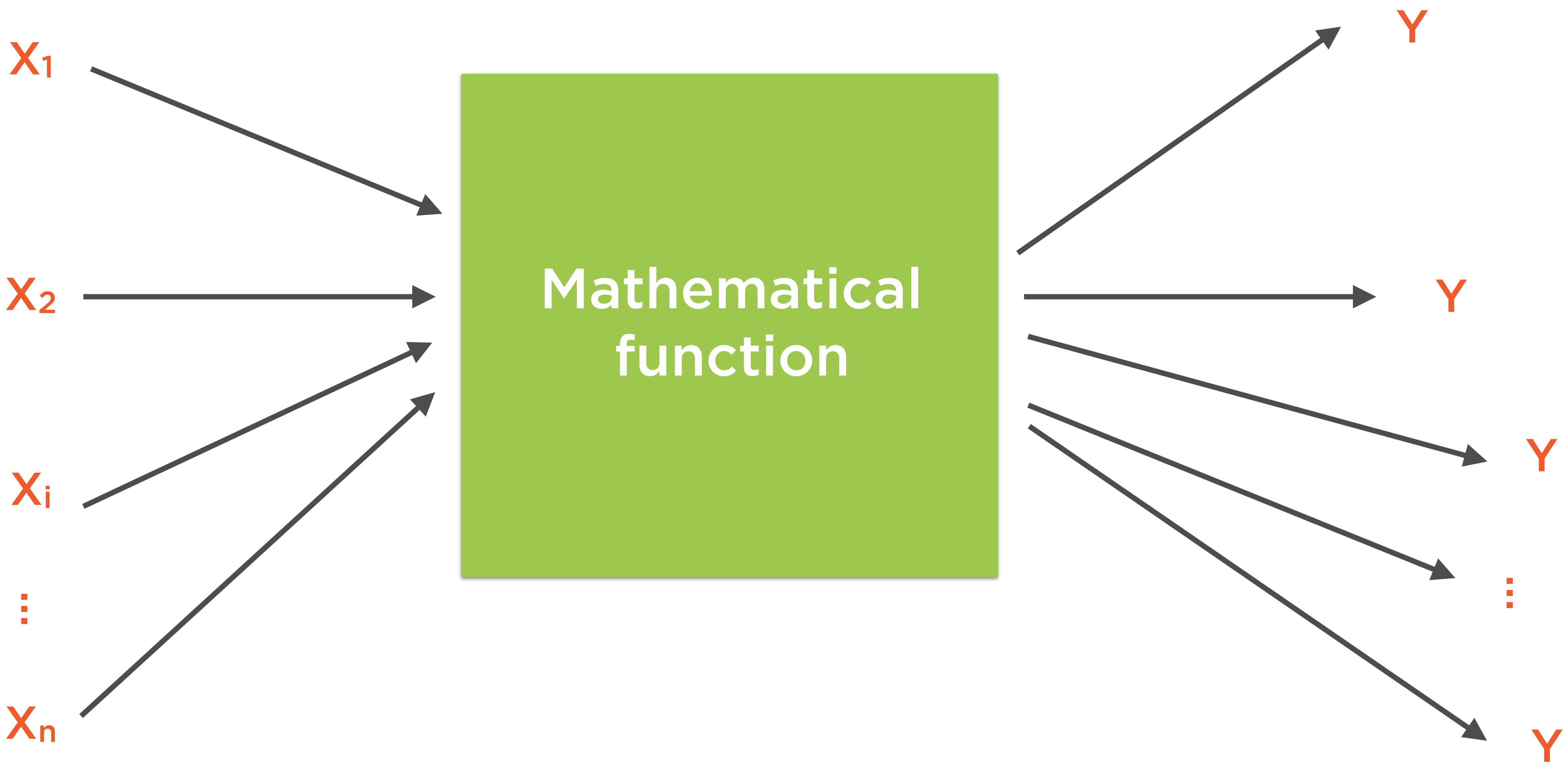
Layers in a neural network

ML-based Classifier

Neural Networks

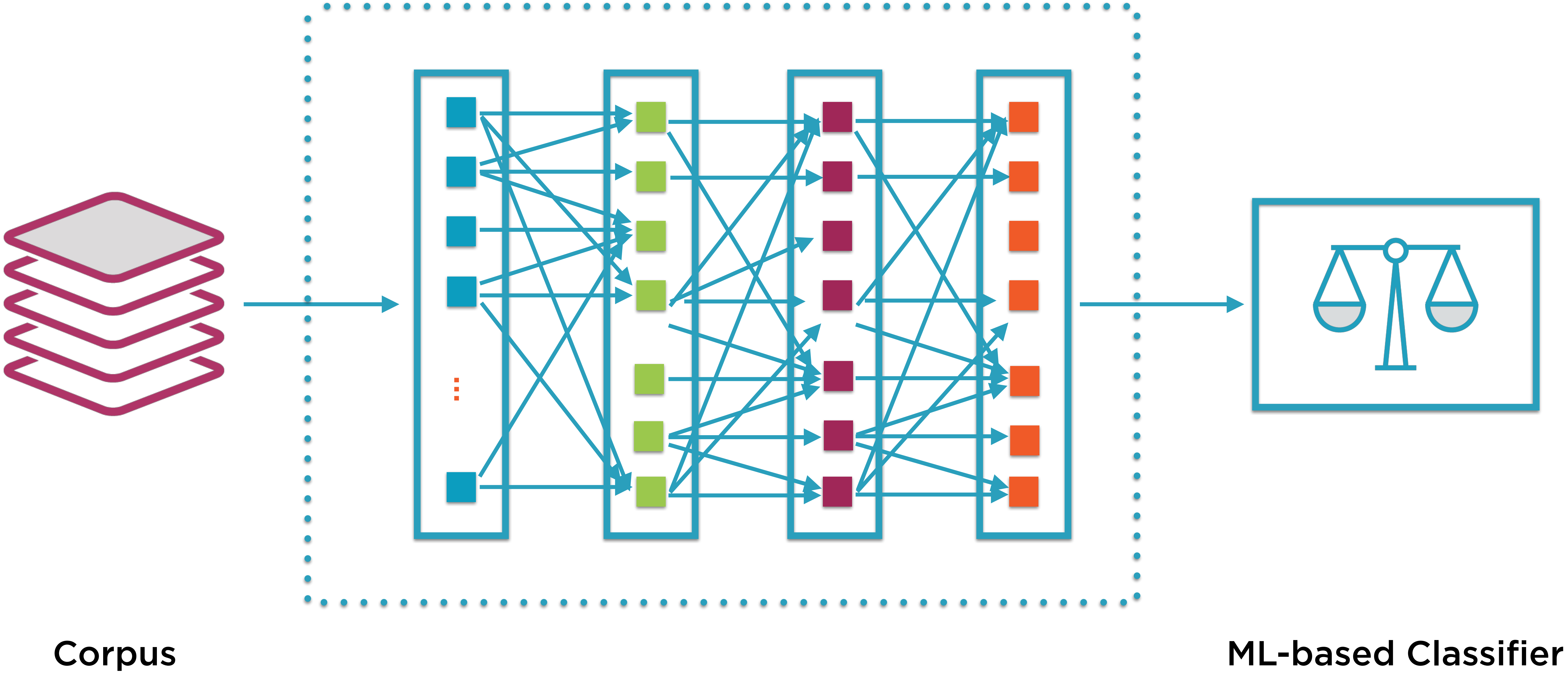


Neuron is a Mathematical Function



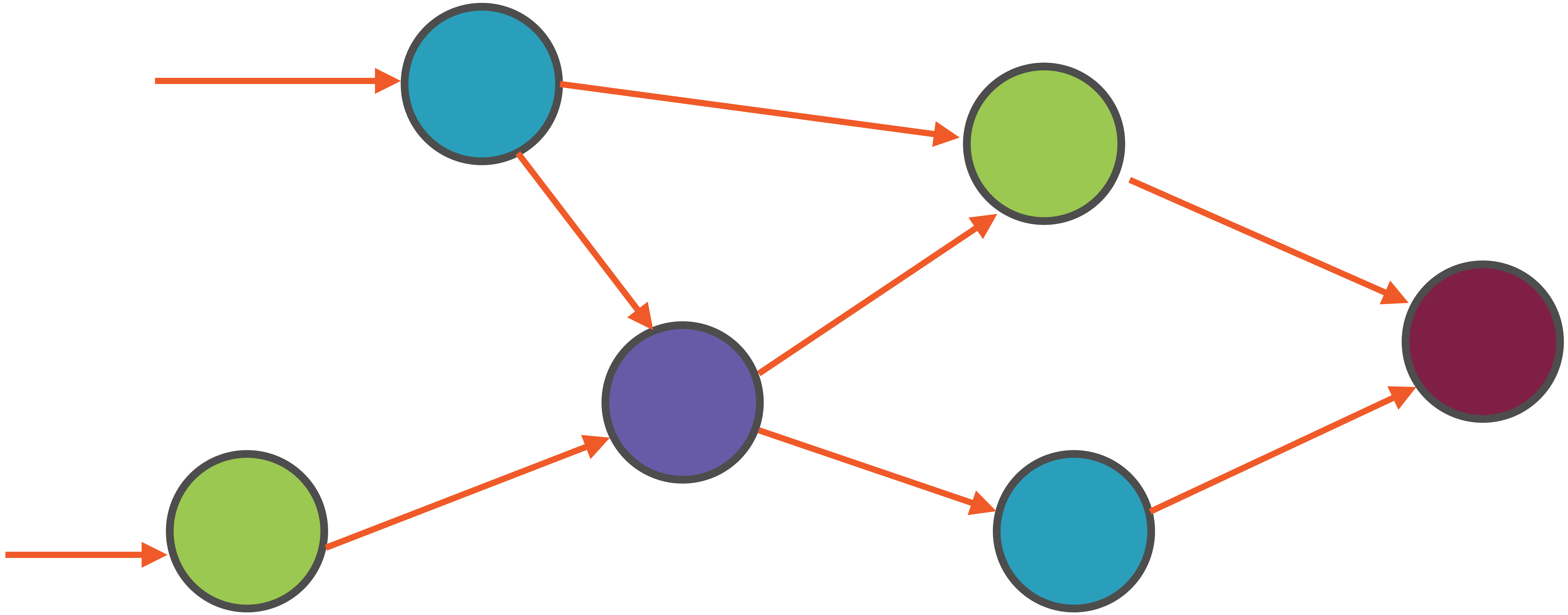
A combination of a linear function and an activation function

Directed-acyclic Graphs

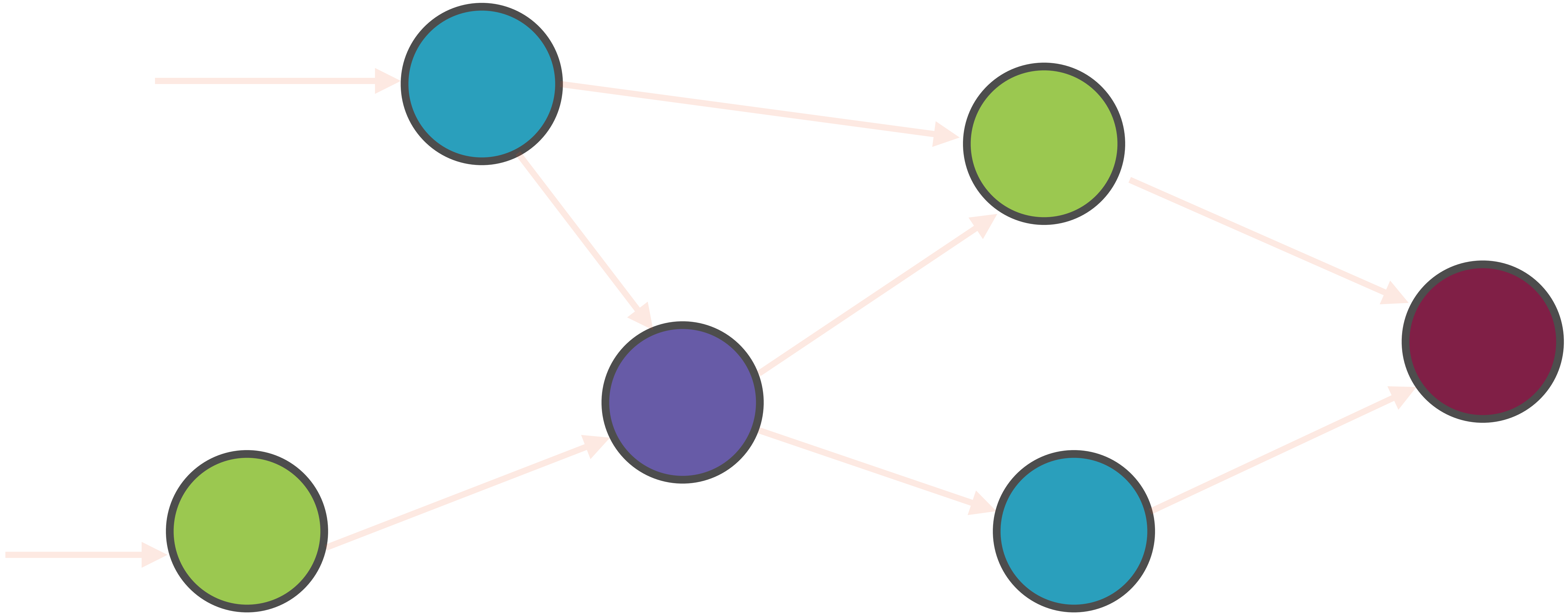


All of the computations and tensors in PyTorch together make up a directed-acyclic graph

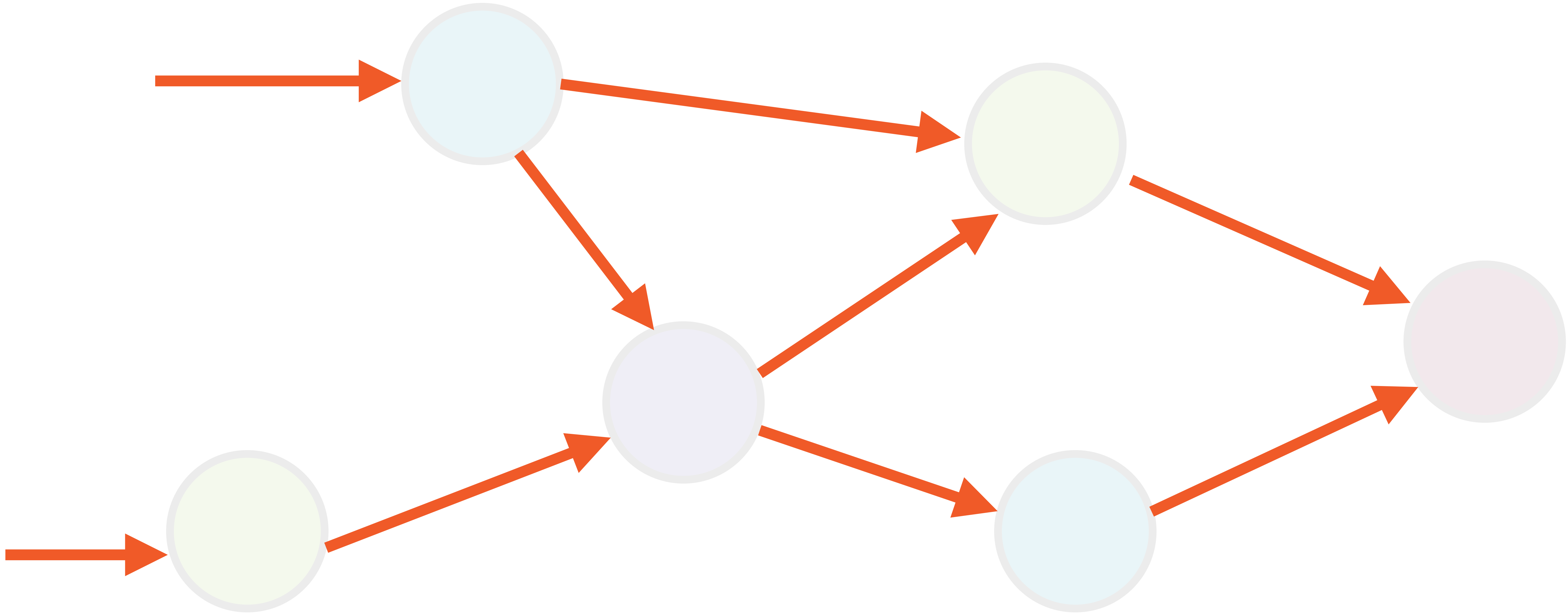
Everything Is a Graph



Tensors



Functions Which Mutate Tensors



PyTorch computation graphs are **dynamic**

The graph is defined as it is executed

Two Approaches to Computation Graphs

Static

TensorFlow - Symbolic programming of NNs

Dynamic

PyTorch - Imperative programming of NNs

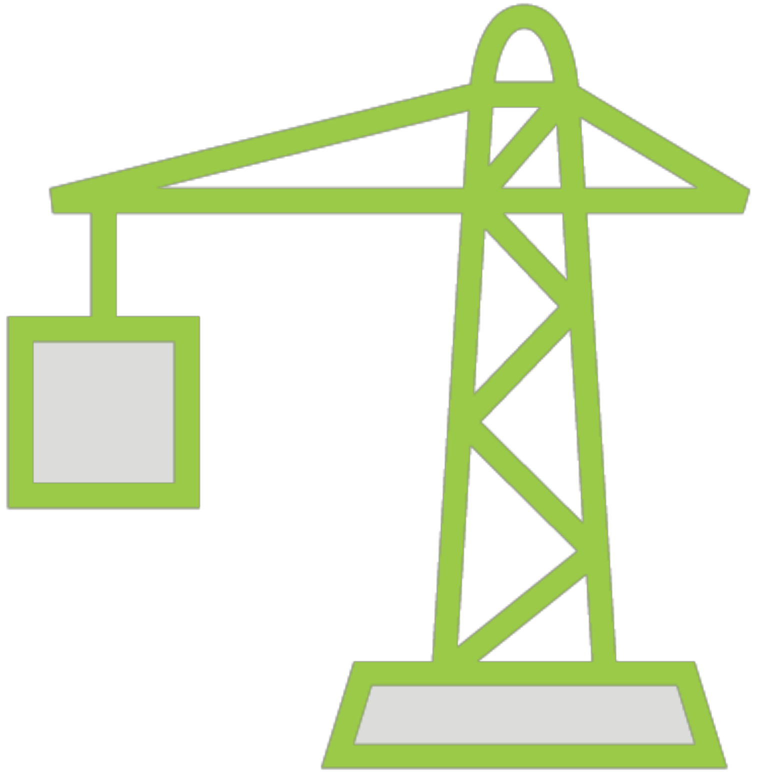
Two Approaches to Programming

Symbolic	Imperative
First define operations, then execute	Execution performed as operations defined
Define functions abstractly, no actual computation takes place	Code actually executed as the function is defined
Computation explicitly compiled before evaluation	No explicit compilation step before evaluation
e.g. Java, C++	e.g. Python

Two Approaches to Building NNs

Symbolic	Imperative
First define computation, then run	Computations run as they are defined
Computation first defined using placeholders	Computation directly performed on real operands
Computation explicitly compiled before evaluation	No explicit compilation step before evaluation
Results in static computation graph	Results in dynamic computation graph

TensorFlow: “Define, Then Run”



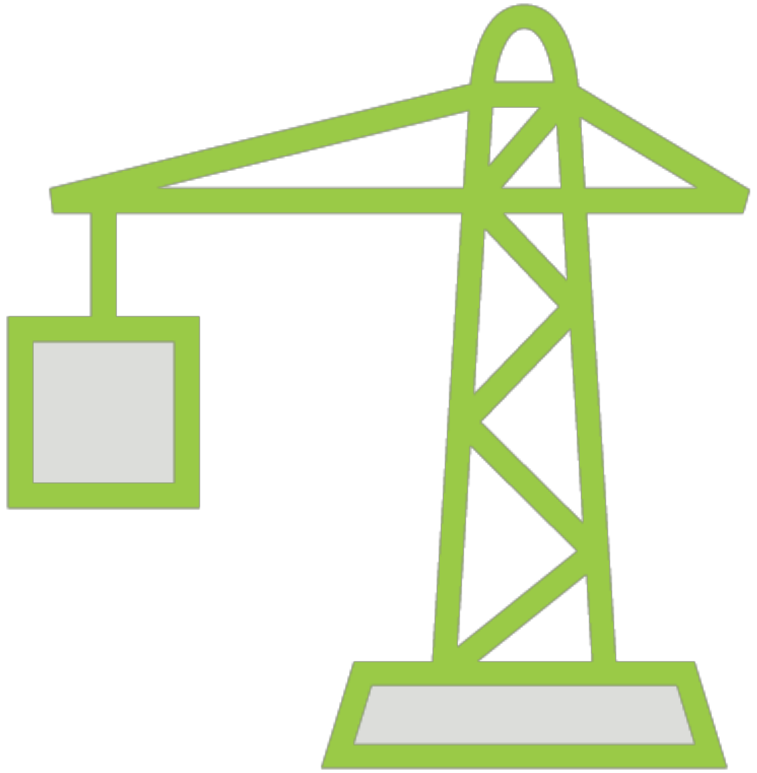
Building a Graph

Specify the operations and the data

Running a Graph

Execute the graph to get the final result

PyTorch: “Define by Run”



Building a Graph

Specify the operations and the data

Running a Graph

Execute the graph to get the final result

Two Approaches to Computation Graphs

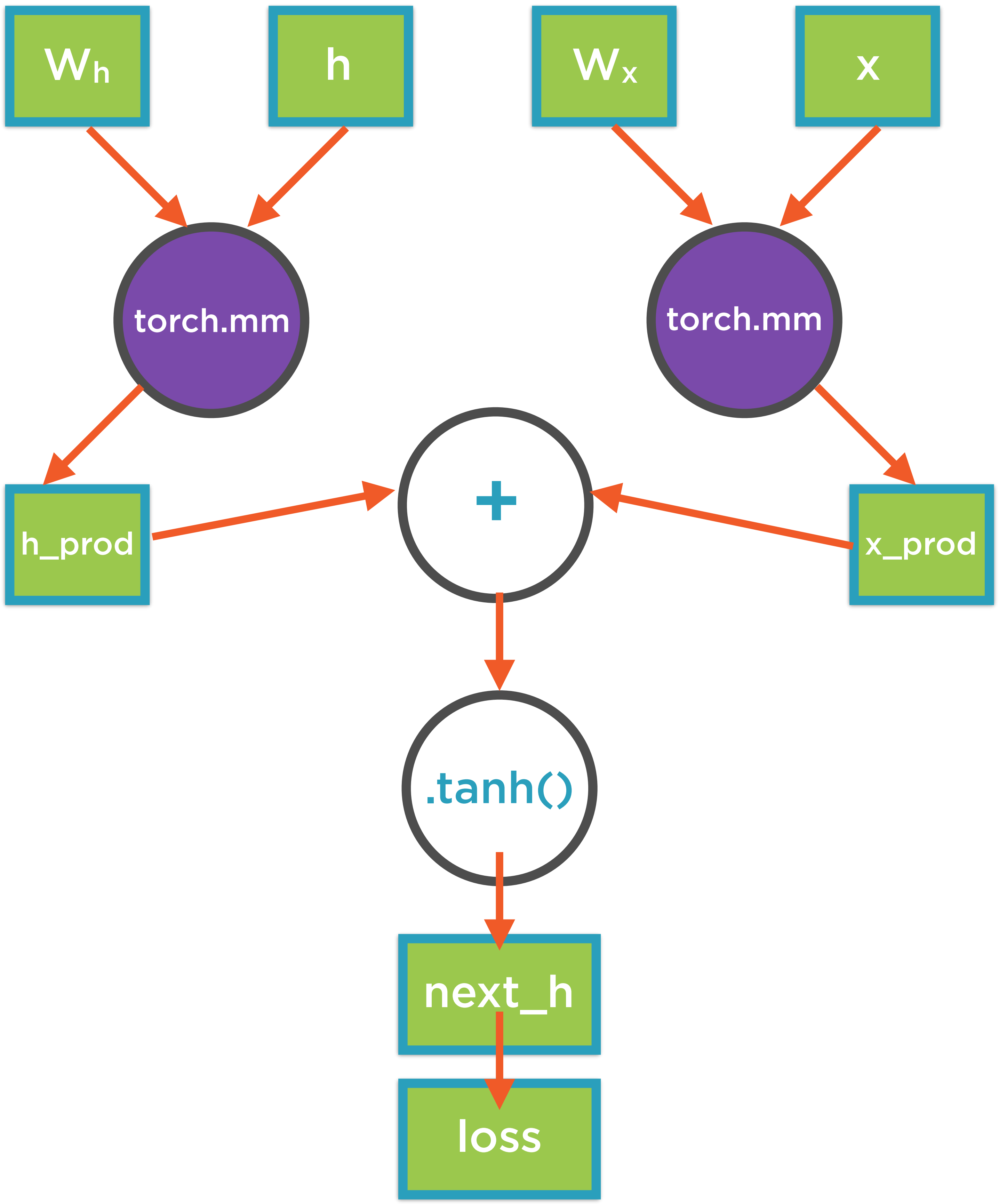
Static	Dynamic
TensorFlow	PyTorch
"Define, then run"	"Define by run"
Explicit compile step	No explicit compile step
Compilation converts the graph into executable format	Graph already in executable format

Two Approaches to Computation Graphs

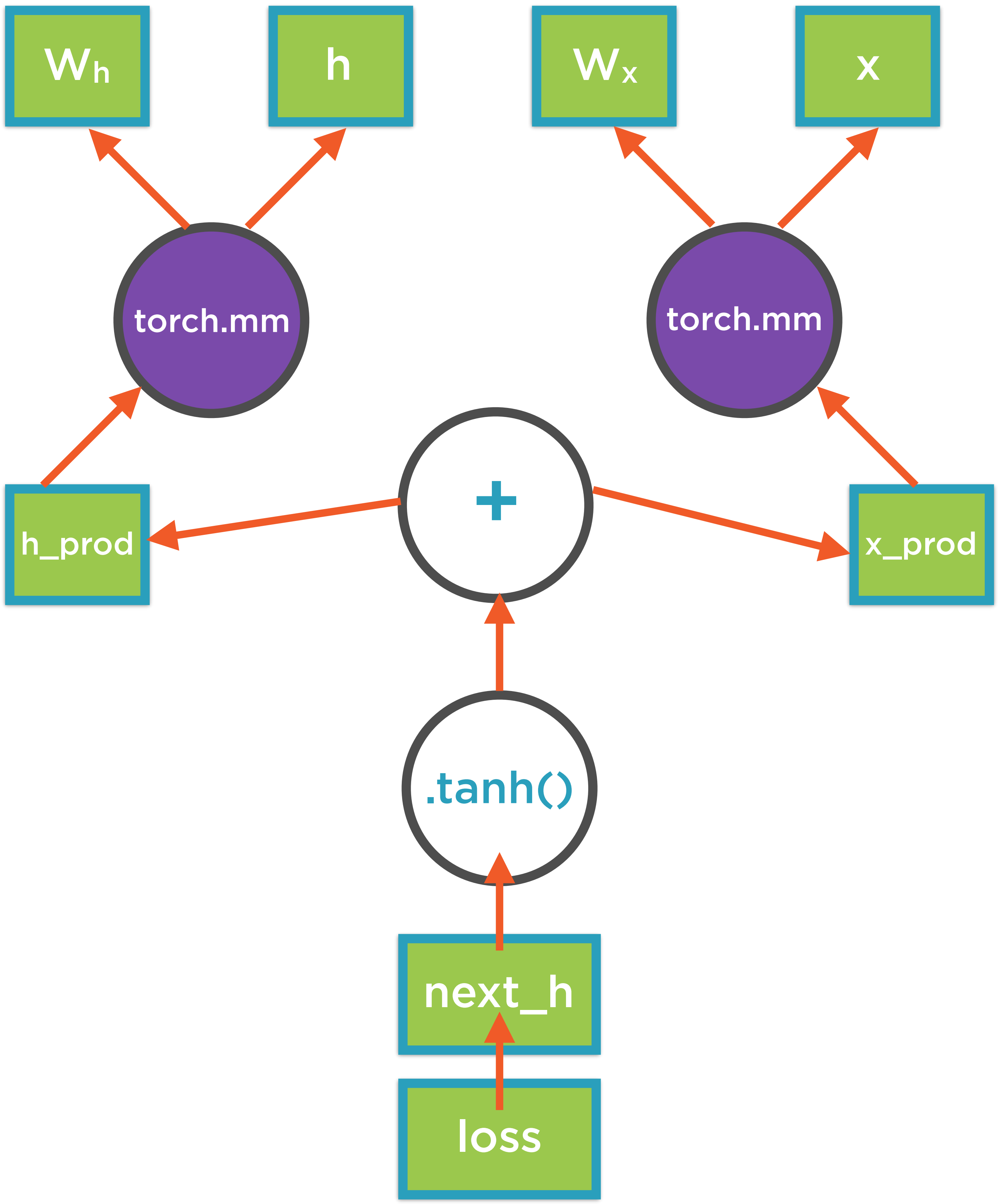
Static	Dynamic
Harder to program and debug	Writing and debugging easier
Less flexible - harder to experiment	More flexible - easier to experiment
More restricted, computation graph only shows final results	Less restricted, intermediate results visible to users
More efficient - easier to optimize	Less efficient - harder to optimize

Build and execute the graph in one go - execute as you build

```
x = torch.randn(1,10)

h = torch.randn(1,20)

W_h = torch.randn(20,20)

W_x = torch.randn(20,10)


h_prod = torch.mm(W_h,h.t())

x_prod = torch.mm(W_x,x.t())

next_h = (h_prod + x_prod).tanh()

loss = next_h.sum()
```



```
x = torch.randn(1,10)
h = torch.randn(1,20)
W_h = torch.randn(20,20)
W_x = torch.randn(20,10)

h_prod = torch.mm(W_h,h.t())
x_prod = torch.mm(W_x,x.t())
next_h = (h_prod + x_prod).tanh()
loss = next_h.sum()
loss.backward()
```



Demo

Installing TensorFlow

Demo

PyTorch for dynamic computation graphs

Demo

TensorFlow for static computation graphs

Demo

Eager execution in TensorFlow for dynamic computation graphs

Debugging in PyTorch

Debugging PyTorch is just like debugging Python - can use pdb and breakpoints (unlike in TensorFlow)

Debugging PyTorch is Easy

pdb is a standard Python debugger

Instrument code with pdb.set_trace()

Can step through forward as well as backward passes in training

Debugging TensorFlow is Hard

Fetch tensors using Session.run()

Print tensors using tf.Print()

Use tf.Assert

Interpose Python code using tf.py_func()

Use tfdbg to debug graph execution

Overview

Static vs. dynamic computation graphs

Benefits and drawbacks for dynamic graphs

Building dynamic graphs in PyTorch

Contrast with static graphs built in TensorFlow

Related Courses

Using PyTorch in the Cloud: PyTorch Playbook

Understanding the Foundations of TensorFlow

Building Deep Learning Models Using Apache MXNet