**PyTorch**, created by Facebook’s AI Research lab, has gained recognition for its simplicity and user-friendliness. Pytorch can efficiently handle dynamic computational graphs.

A **computation graph** is a way to represent mathematical operations and data flow in a model — nodes represent operations, and edges represent the flow of data (tensors).

Computation graphs are a visual representation of mathematical operations and their relationships, which help show how data flows through the model, effectively assisting the computer in organizing and executing calculations efficiently when training neural networks.

PyTorch is simpler and works in a “Pythonic” way, making it a great choice for both beginners and researchers. Its dynamic computation graph makes things easier to change on the fly, which fits its use for experimentation.

PyTorch is generally easier to use than TensorFlow, making it a preferred tool for people want a more straightforward approach in their project.

**TensorFlow**, Google’s brainchild, has robust production capabilities and support for distributed training. TensorFlow excels in scenarios where you need large-scale machine learning models in real-world applications.

TensorFlow uses a technique called distributed training, which is applied to large complex models, essentially spreading the training process across multiple devices. TensorFlow is the preferred choice for companies that require scalability and reliability in their models.

TensorFlow is more structured in its approach. It utilizes static computation graphs, which forces the developer to plan ahead. It comes with a steep learning curve, which might ultimately lead to more optimized and high-performance models.

TensorFlow 2.0 has attempted to bridge the gap in simplicity between it and PyTorch through its Eager Execution feature, which gives you the ability to run operations immediately, using dynamic computation graphs but it still falls short of PyTorch’s simplicity and ease of learning.

**PyTorch vs TensorFlow: What to use?**

For those who need ease of use and flexibility, PyTorch would be more convenient, if scalability is more of a priority, however, TensorFlow is the way to go.