CS 410: Text Information Systems

University of Illinois at Urbana Champaign

Fall 2022

Technology review

Submitted by: Unnati Hasija

uhasija2@illinois.edu

Pytorch and TensorFlow comparison for NLP

Introduction: The deep learning frameworks like PyTorch, Tensorflow, Keras, Caffe, and DeepLearning4j for making machines learn like humans with special brain-like architectures known as Neural Networks. The war of deep learning frameworks has two prominent competitors- PyTorch vs Tensorflow because the other frameworks have not yet been adopted widely. Every deep learning framework depends on the generation of computation graphs for calculating the gradient values needed for gradient descent optimization.

Static vs Dynamic computation graphs:

In a static computational graph, you just define the neural network architecture and run it through a bunch of data to train the model and make predictions while a dynamic computational graph is defined implicitly once the forward propagation is executed. One just needs to write the forward computation with all the features and the computational graph is built on-demand dynamically offering flexibility. Dynamic computational graphs are easy to debug, unlike static computational graphs.

Pytorch:

PyTorch is an open-source machine learning library. PyTorch fits perfectly in the machine learning ecosystem as it is developed to be used in Python though it has a C++ interface. Compared to other declarative deep learning frameworks, PyTorch is popular for its imperative programming style which makes it more pythonic. It is imperative, which means computations run immediately, which means the user need not wait to write the full code before checking if it works or not. We can efficiently run a part of the code and inspect it in real-time. The library is a python based built for providing flexibility as a deep learning development platform. PyTorch makes use of dynamic computational graphs.

For visualization, PyTorch's native visualization tool Visdom is flexible, customized, lightweight, and easy to use with support for PyTorch tensors.

For deployment, PyTorch users use TorchServe, a flexible model deployment tool. The model deployment tool for PyTorch is experimental stage. However, with a basic set of features like a model archiver tool, a server, metrics, logging, an API endpoint specification, and model snapshots, TorchServe attracts most of the business cases.

Encoding words into your model is probably one of the most obvious and important processes to having a fully-optimized and operational deep learning model with natural language processing.

There are many ways to have models process individual letters, but the point of creating NLP deep learning models is to, presumably, focus not on individual words and letters but the semantics and linguistic meanings of those words and phrases. Here are three basic word embedding models for NLP with PyTorch:

Simple word encoding: training the model to focus on each individual word in a sequence and letting them derive similarities and differences on their own. This is the simplest but can be difficult for the model to accurately understand or predict semantics.

N-Gram language modeling: the model is trained to learn words with respect to the words in the sequence. They can learn how words function in relationship to one another and in sentences as a whole.

Continuous bag-of-words (CBOW): an expanded version of N-Gram language modeling, the deep learning model is trained to sequence data of a set amount of words before and a set amount of words after each word in a sequence in order to deeply learn how words function with surrounding words and how they function in their sequence. This is, by far, the most popular method used by those using natural language processing with PyTorch.

Use cases for PyTorch:

- Handwritten Digit Recognition
- Text Generation
- Style Transfer
- Forecasting Time Sequences
- Image Classification

Tensorflow:

TensorFlow uses dataflow graphs to process data. Nodes in the graph represent mathematical operations, while the graph edges represent the multidimensional data arrays (tensors) that flow between them. As you build these neural networks, you can look at how the data flows through the neural network. As compared to PyTorch, TensorFlow would be a little hard to comprehend because of the low-level implementations of the neural network structure. However, its very high-level Keras API makes it easy for beginners to learn the basic concepts, and later TensorFlow can be used for a better understanding of the implementations of neural network structure. To add, debugging code in TensorFlow one needs to learn about the TF debugger and also about the variables that are requested from a TensorFlow session which is complex compared to debugging in PyTorch.

TensorFlow creates optimized static computational graphs.

For visualization, TensorFlow's Tensorboard has an awesome in-built visualization tool with a suite of web apps for understanding a deep learning model through 5 different visualizations- graphs, scalars, audio, histograms, and images.

For deployment, TensorFlow has an in-built model deployment tool TensorFlow Serving. It has been tested on more than 1000 projects and is capable of handling millions of requests per second. It is designed for industrial production environments and is a good choice where performance is a concern.

Use cases for TensorFlow:

- Video Detection
- Image Recognition
- Voice and Speech Recognition
- Text-Based Applications

Conclusion

About 38% of ML developers and data scientists work on TensorFlow and about 11% use PyTorch and 4% use both. PyTorch is easier to learn since it uses Python but doesn't fulfill all Text based use cases. TensorFlow is more accurate at lower epochs and consumes less memory as compared to PyTorch and that's why TensorFlow is widely used at the production level in Industry whereas PyTorch is more popular in the research community.

References:

https://medium.com/featurepreneur/tensorflow-vs-pytorch-which-is-better-for-your-application-development-6897d5d4dee0

https://www.projectpro.io/article/pytorch-vs-tensorflow-2021-a-head-to-head-comparison/416#toc-1

https://towardsdatascience.com/tensorflow-vs-pytorch-vs-keras-for-nlp-exxact-8e51dd13c3f5#:~:text=This%20factor%20is%20especially%20important,world%20is%20performed%20via%20tf.

https://www.knowledgehut.com/blog/data-science/pytorch-vs-tensorflow