Deployment

Machine Learning Operations

Nicki Skafte Detlefsen,

Postdoc

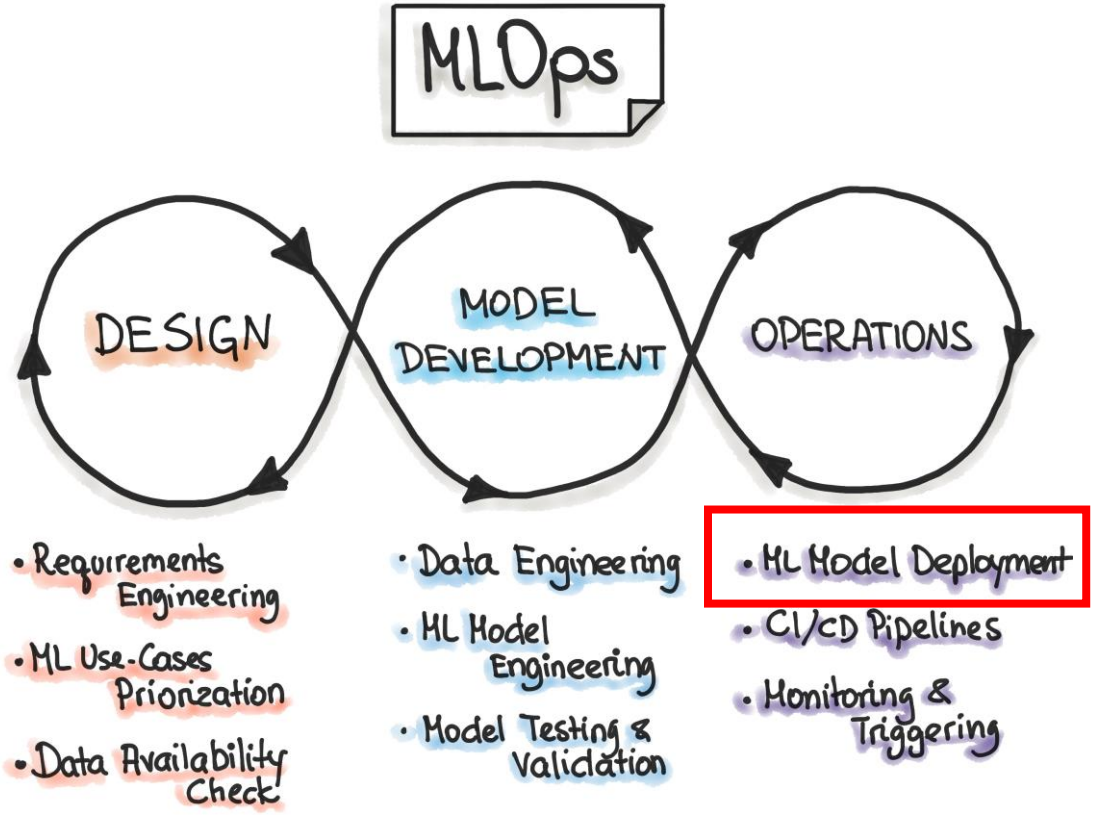
DTU Compute

Loosely based on https://www.youtube.com/watch?v=2awmrMRf0dA

Freeing the model

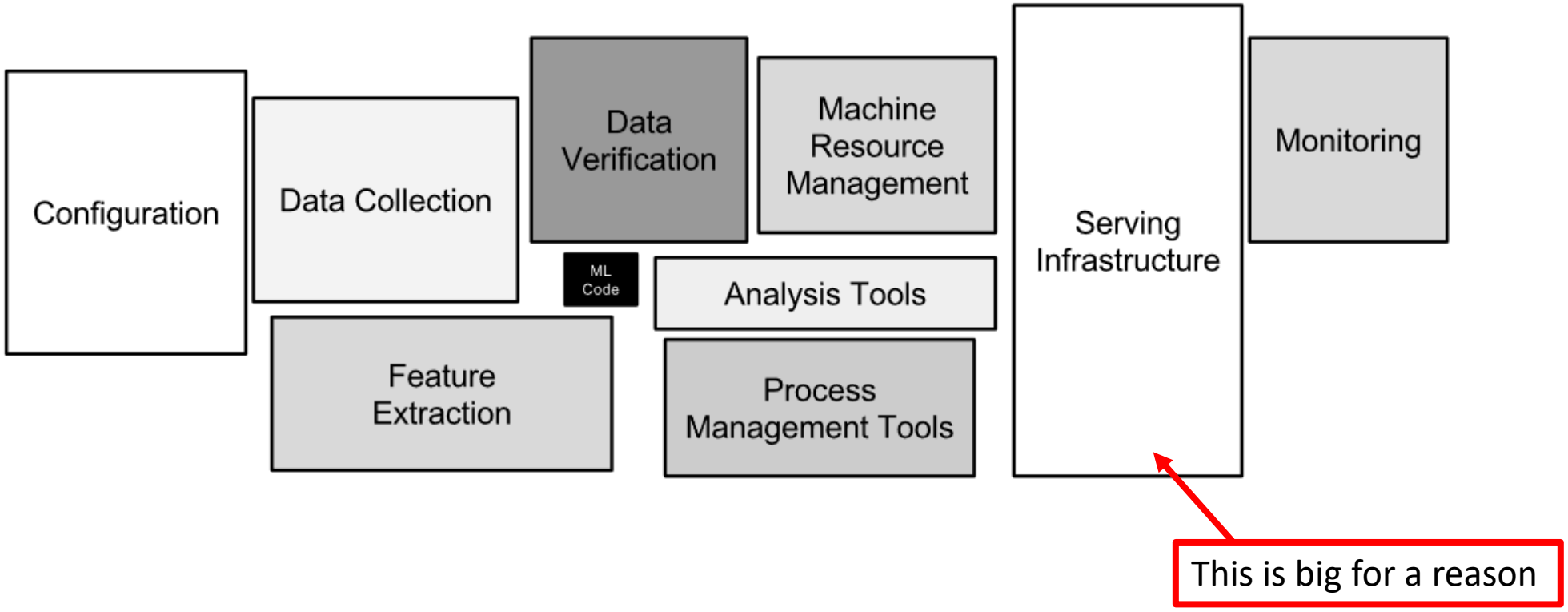
MLOps

- Model deployment is part of the operations in MLOps
- In a nutshell: make the model available to others



Remember this?

MLOps



Many levels of deployment (within machine learning)

MLOps

1. Github reposatory + link to model weights
   - Easy to ”deploy”
   - Pain in the *** to use
2. Deploy on local computer/cluster
   - Fairly easy getting up and running, just requires people can access from outside
   - Can be fairly easy to use
   - Does not scale at all
3. Deploy to cloud service
   - Can be a pain to setup
   - Easy to use and scales to ∞ (and beyond!)

Production requirements

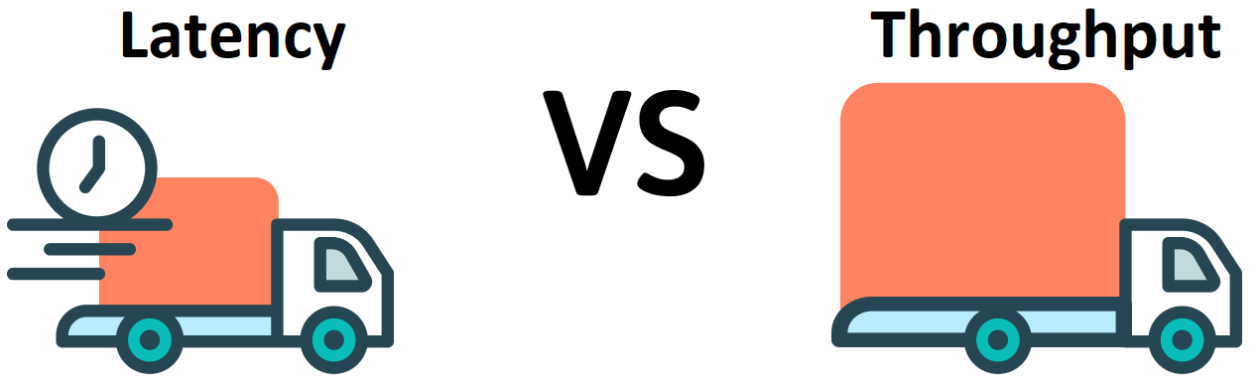
MLOps

1. Portability

   Models should be exportable to wide variety of enviroments, from C++ servers to mobile

2. Performance

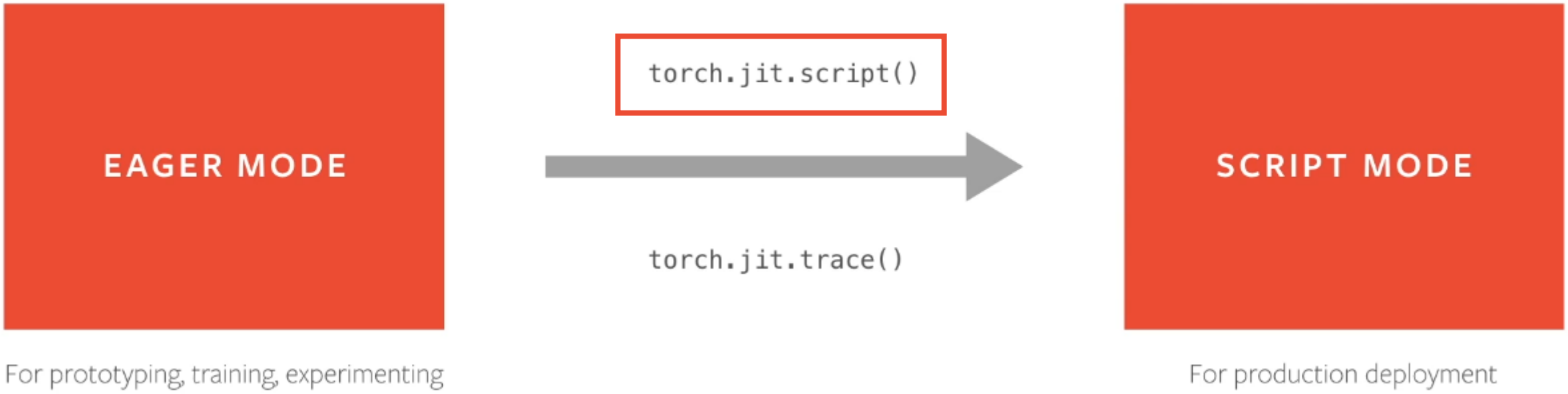
   We want to optimize common patterns in neural networks to improve inference latency and throughput



What are the challenges with Pytorch in production

- Pytorch is a dynamic framework (uses a dynamic graph)
  - This is not great in production as we need to know sizes etc. for compilation and optimization
- Why not use a static framework (Tensorflow 1.x, Caffe2 etc.)?
  - Do you really want to port all your work?
- What can we do to solve this?

Convert to script mode!



Serilization

MLOps

- torch.jit.script **serialize the model, but what does it mean?**
- Serilization essentially encodes all modules methods, submodules, parameters, and attributes into a byte stream
- This makes the encoded model independent of python!
- This is basically just ”pickling” and “unpickling”.