Pyro
for Deep Generative Models

Presented By

Pranav Talluri

Significance

- PPLs aim to reduce the difficulty of developing probabilistic models
- All PPLs offer some trade-off between key properties
  - Expressivity
  - Scalability
  - Flexibility
  - Minimality
- Pyro aims to offer an optimal trade-off

System	Expressivity: Dynamic control	Scalability: Subsampling, AD	Flexibility: Flexible inference	Minimality: Host language
Stan	Static control flow	Some, CPU	Automated	None
Church	Yes	No, None	Automated	Scheme
Venture	Yes	No, None	Yes	None
webPPL	Yes	No, CPU	Some	JavaScript
Edward	Static control flow	Yes, CPU/GPU	Yes	TensorFlow
Pyro	Yes	Yes, CPU/GPU	Yes	Python, PyTorch

Context: Deep Generative Modelling

- Exciting field, with rapid progress
- Fundamentally rooted in probabilistic ideas and inference
- Examples
  - GANs
  - VAEs
  - Normalising Flows
  - Diffusion
  - LLMs

Goals

- Investigate whether an optimal trade-off is achieved by Pyro
  - Specifically in the context of deep generative models
- Study the (under-)utilisation of Pyro for deep generative models
  - Experiment with my own code
  - Review existing literature that leverages Pyro
- Provide a comprehensive review of Pyro for deep generative models as a guide for researchers
- Compare Pyro to alternative PPLs, and other options, for deep generative modelling

Progress

- Completed:
  - Literature review of Pyro
  - Implementation of basic Bayesian inference in Pyro
  - Implementation of VAEs in Pyro
- In progress:
  - Learning how to use Pyro
- To do:
  - Implementations of other generative models in Pyro
  - Exploration of wider literature
  - Exploration of alternatives to Pyro
  - Writeup

Any questions?