(Deep) probabilistic programming for Fermi LAT point sources

Christoph Weniger, Univ. of Amsterdam, 1 Oct 2018

Problem

- How to make sense of gamma-ray point sources?
  - Spatial distribution of various source classes?
  - Spectral characteristics
  - Characterizing unassociated sources
  - Accounting for unresolved sources
  - Are there DM subhalos contributing?
- Technical challenges
  - Source detection threshold
  - Association with other sources (multi-wavelength)
  - In the case of pulsars: Complicated completeness function

What is Deep Probabilistic Programming?

Probabilistic programming language

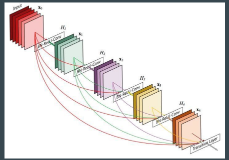
- Functions have a double role as
  - Probability distribution
  - Random number generator
- Functions can be conditioned on data
- Inference is core part of language
- Extremely expressive (sometimes factor 10x less lines of code)

```
(sample, weights) = query(program, parameters, data)
```

Anglican (Clojure)
Church (Scheme)
Stan (C++)
PyMC3 (Python)
Turing.jl (Julia)
Edward (Tensorflow)
Pyro (PyTorch)
...

Deep learning

- Multiple layers → Complex structures
- Millions of parameters



Deep probabilistic programming

- Probabilistic programming applied to deep learning problems
- Here: Probabilistic programming using deep learning machinery

General strategy (semi-supervised learning)

Parametric model (per source class)
- Energy spectra (parameterization and prior)
- Spatial distribution
- Variability

Detection probability
- For given spectrum, can be estimated with Fisher forecasting
- Differentiable

Extra information
- Distance
- Association

Model likelihood
- Hierarchical Bayesian model
- (Un-)known source types are implemented as discrete priors
- → log(L)

3FGL
- Position
- Luminosity
- Spectrum

Sub-threshold sources?
- Information could be obtained from PSC-masked gamma-ray data
- Need to incorporate this in likelihood analysis, CCN not helpful
- Mayby train network on wavelet fluctuation analysis results?

Inference
- Hamiltonian Monte Carlo and variational inference
- Source types inferred from posterior predictive distribution

Why variational inference?

Problem

- **Very** large number of parameters, required for each of the sources
  - Spectral parameters
  - Source distance
  - Luminosity
  - Source type

→ O(10000) parameters

Solution

- This could be still handled using Variational inference, where the idea is to not sample the posterior, but instead to fit it with, e.g., a normal distribution
- We can rely on auto-differentiation in order to make optimization feasible

In practice this is done by ELBO maximization

$$\begin{aligned}\lambda^* &= \arg\min_{\lambda} \mathrm{KL}(q(\mathbf{z}\,;\,\lambda) \,\|\, p(\mathbf{z} \mid \mathbf{x})) \\ &= \arg\min_{\lambda} \; \mathbb{E}_{q(\mathbf{z}\,;\,\lambda)}\big[\log q(\mathbf{z}\,;\,\lambda) - \log p(\mathbf{z} \mid \mathbf{x})\big]\end{aligned}$$

$$\begin{aligned}\log p(\mathbf{x}) = \mathrm{KL}(q(\mathbf{z}\,;\,\lambda) \,\|\, p(\mathbf{z} \mid \mathbf{x})) \\ + \; \mathbb{E}_{q(\mathbf{z}\,;\,\lambda)}\big[\log p(\mathbf{x}, \mathbf{z}) - \log q(\mathbf{z}\,;\,\lambda)\big]\end{aligned}$$

$$\mathrm{ELBO}(\lambda) = \; \mathbb{E}_{q(\mathbf{z}\,;\,\lambda)}\big[\log p(\mathbf{x}, \mathbf{z}) - \log q(\mathbf{z}\,;\,\lambda)\big]$$

Computational tools

Machine learning library

- Tensorflow
  - Biggest community
  - Great visualisation and debugging tools
  - Static graphs
- Pytorch
  - Build on torch
  - Growing community
  - Dynamic graphs

Probabilistic programming library

- Edwardlib
  - Very well developed
  - Large number of inference strategies implemented
  - Built on tensorflow
- Pyro
  - Using Pytorch
  - Relatively easy to extend
  - Rather `pythonic' and hence easy to use