Deep Gaussian processes for large datasets

James Hensman[a,b], Andreas Damianou[a,b] and Neil D. Lawrence[a,b]

Problem

We wish to develop a Deep Belief Network where the transformation between layers is probabilistic and modelled with Gaussian processes.

../diagrams/graphical_full.pdf

- The function $f_l(\cdot)$ is modelled with a GP
- Assume noise β_l at each level
- The mapping instantiations $\bar{f}_l = f_l(h_l)$ can be marginalised out analytically: $p(h_l|h_{l-1}) =$

$$= \int p(h_l|\bar{f}_l) p(\bar{f}_l|h_{l-1})\,\mathrm{d}\bar{f}_l$$
$$= \mathcal{N}(h_l|0, K(h_{l-1}, h_{l-1}) + \beta_l I)$$

- *How to learn the intermediate hidden layers?*
- *How to efficiently train the model?*

Inducing variables

../diagrams/graphical_inducing.pdf

- Also marginalise out hidden spaces: learn a posterior $q(h_l)$ for each layer.
- Use *inducing points* z_l: $u_l = f_l(z_l)$.
- Let $q(u_l) = \mathcal{N}(u_l|m_l, \Sigma_l)$.
- $\{z_l, u_l\}$ play the same role as $\{h_l, \bar{f}\}$ pairs but result in low rank representations of the covariance matrices.
- Inducing points are variational parameters allowing us to lower bound the evidence: $\mathcal{F} \leq \log p(y)$

Given the above, how can we define $p(h_l|h_{l-1}, u_l)$?

Inference strategies

We need to deal with $q(h)$ and $q(u)$. Three strategies:

1. Collapse out u *([Damianou et al., AISTATS 2013])*
   - Optimize $q(h)$
2. Maintain $q(h)$ and $q(u)$
   - EM-style optimisation for $q(u)$ and $q(h)$
3. Compress $q(h)$ into $q(u)$ using $p(h|u)$

Variational compression

Consider one layer of our infrence problem. Use the conditional distribution as the variational distribution:

$$p(y|u) = \frac{p(y|h)p(h|u)}{p(h|u,y)}$$
$$\log p(y|u) = \mathbb{E}_{p(h|u)}[\log p(y|h)] + \underbrace{\mathrm{KL}\left[p(h|u) \parallel p(h|b,y)\right]}_{\text{Small if } u \text{ explains } h \text{ very well}}$$

We can compute the marginal distribution in the variational approximation easily:

$$q(h_l) = \int \underbrace{p(h_l|u_l)}_{\sim\mathcal{N}} \underbrace{q(u_l)}_{\sim\mathcal{N}} \mathrm{d}u_l$$

../diagrams/graphical_collapsed.pdf

- Given X and a fixed $q(u_1)$, we can compute $q(h_1)$
- For a fixed $q(h_1)$, we can variationally propagate using $q(u_2)$ to get $q(h_2)$ (blue arrows)
- Continue to feed-forward to the bottom layer. The variational propagation at each layer introduces a penalty (regularizing) term which affects the bound on the marginal likelihood
- Applying the chain-rule leads to backpropagation (red arrows), but with *Gaussian* messages passed layer-to-layer

Inference for large datasets

How can we handle large datasets?

Stochastic variational inference (SVI):

SVIGP-style: *[Hensman et al., UAI 2013]*

- Represent the parameters $\{m, \Sigma\}$ of $q(u)$ in two *equivalent* ways:
  - Canonical form: $\theta = \{\Sigma^{-1}m, -\frac{1}{2}\Sigma^{-1}\}$
  - Expectation form: $\eta = \{m, mm^\top + \Sigma\}$
- Treat u as *global variables*. This allows for the factorisation of the contributions of every input/output pair $\{x_{l,i}, y_i\}$.
- Optimise the parameters using the natural gradients of $q(u)$:

$$\theta_{t+1} = \theta_t + s\frac{\mathcal{F}}{\mathrm{d}\eta},$$

where s is the learning step

Adapting the learning step s:

- Stochastic optimisation is very sensitive to s.
- [Ranganath et al. ICML 2013] dynamically adapt s to minimize the expected loss between the parameter vector after the stochastic variational update, θ_{t+1}, and the vector after a full variational update θ^*_{t+1}
- Here we consider the loss in the KL sense, considering the involved distributions:

$$\mathrm{KL}\left[q(u|\theta^*) \parallel q(u|\theta)\right]$$

This takes into account the geometry of the parameter space.

TODO

- Currently, SVI inference is implemented only for 1-layer models
- Extend SVI inference framework in deep models
- Training scheme combining optimisation of variational and kernel parameters
- Fix initialisation issues
- Explore auto-encoder architectures

Experiments

Toy problem

Fit

GP (1 layer)

../diagrams/png/H0_predict.png

2 layers

../diagrams/png/H1_predict.png

4 layers

../diagrams/png/H3_predict.png

Hidden spaces

../diagrams/step_all.pdf

Loop detection in robotics

True path

../diagrams/robot_path.pdf

Hidden layer 1

../diagrams/robot_layer1.pdf

Hidden layer 2

../diagrams/robot_layer2.pdf

- Dynamically constrained model
- Correctly detects the loop
- Learns temporal continuity and corner-like features in different layers

Data fit

../diagrams/png/robot_wireless_signal1.png

../diagrams/png/robot_wireless_signal2.png

Example data fits for 2 of the 30 output dimensions

Big Data

- 12 Subjects, 95 diverse motions, 20K datapoints
- Learns a general model of human motion
- Outperforms Bayesian GP-LVM (trained on subsets) for reconstructing part of test body parts
- We considered a 1-layer model but used SVI inference with adaptive learning step

Example frame

../diagrams/mocapExample.pdf

Adaptive learning step

../diagrams/learning_steps.pdf

Hidden space projections:

Global motion features

../diagrams/png/X0_vs

Clustered motion features

../diagrams/png/X2_vs

[a] University of Sheffield, Department of Computer Science, UK
Corresponding author: james.hensman@sheffield.ac.uk

[b] Sheffield Institute for Translational Neuroscience (SITraN), UK