Composing graphical models with neural networks

(most slides stolen from Matthew James Johnson)
Outline

● Motivation and Examples
● Stochastic Variational Inference
● Structured Variational Auto-Encoders
Example: Generative Clustering

\[ \pi \sim \text{Dir}(\alpha), \quad (\mu_k, \Sigma_k) \overset{\text{iid}}{\sim} \text{NIW}(\lambda) \]

\[ z_n \overset{\text{iid}}{\sim} \pi \quad y_n \mid z_n, \{(\mu_k, \Sigma_k)\}_{k=1}^K \overset{\text{iid}}{\sim} \mathcal{N}(\mu_{z_n}, \Sigma_{z_n}) \]
Example: Generative Clustering

\[ \gamma \sim p(\gamma) \quad x_n \overset{iid}{\sim} \mathcal{N}(0, I) \]

\[ y_n \mid x_n, \gamma \overset{iid}{\sim} \mathcal{N}(\mu(x_n; \gamma), \Sigma(x_n; \gamma)) \]

\[ \pi \sim \text{Dir}(\alpha) \quad (\mu_k, \Sigma_k) \overset{iid}{\sim} \text{NIW}(\lambda) \]

\[ z_n \mid \pi \overset{iid}{\sim} \pi \quad x_n \overset{iid}{\sim} \mathcal{N}(\mu(z_n), \Sigma(z_n)) \]

\[ y_n \mid x_n, \gamma \overset{iid}{\sim} \mathcal{N}(\mu(x_n; \gamma), \Sigma(x_n; \gamma)) \]
Probabilistic graphical models

+ structured representations
+ priors and uncertainty
- rigid assumptions may not fit
- feature engineering
+ arbitrary inference queries
+ data and computational efficiency within rigid model classes
- more flexible models can require slow top-down inference

Deep learning

- neural net “goo”
- difficult parameterization
+ flexible, high capacity
+ feature learning
- limited inference queries
- data- and compute-hungry
+ recognition networks learn how to do inference
Application: Parse mouse behaviour

Aim: Generative model for different mouse behaviour
Application: Parse mouse behaviour

**Input:** Videos

**Aim:** Come up with a generative model for the behaviour and the video frames
Example: Switching Linear Dynamical System

\[ x_{t+1} = A^{(z_t)} x_t + B^{(z_t)} u_t \quad \text{iid} \sim \mathcal{N}(0, I) \]
Example: Switching Linear Dynamical System
Special Cases

GMM

LDS
conjugate prior on global variables
exponential family on local variables
any prior on observation parameters
neural network observation model
Stochastic Variational Inference
\[ q(\theta)q(x) \approx p(\theta, x \mid y) \]

**Objective function**

\[ \mathcal{L}(\eta_{\theta}, \eta_x) \triangleq \mathbb{E}_{q(\theta)q(x)} \left[ \log \frac{p(\theta, x, y)}{q(\theta)q(x)} \right] \]

\[ \eta_x^*(\eta_\theta) \triangleq \arg \max_{\eta_x} \mathcal{L}(\eta_\theta, \eta_x) \]

\[ \mathcal{L}_{SVI}(\eta_\theta) \triangleq \mathcal{L}(\eta_\theta, \eta_x^*(\eta_\theta)) \]

**Equivalent to the Kalman Filter**

- \( p(x \mid \theta) \) is a linear dynamical system
- \( p(y \mid x, \theta) \) is a linear-Gaussian observation
- \( p(\theta) \) is a conjugate prior
Algorithm

Step 1: compute evidence potentials

Step 2: run fast message passing

Step 3: compute natural gradient
Structured Variational Auto Encoder
Mean Field Approximation

\[ q(\theta)q(\gamma)q(x) \approx p(\theta, \gamma, x \mid y) \]

Objective function

\[
\mathcal{L}(\eta_\theta, \eta_\gamma, \eta_x) \triangleq \mathbb{E}_{q(\theta)q(\gamma)q(x)} \left[ \log \frac{p(\theta, \gamma, x) p(y \mid x, \gamma)}{q(\theta)q(\gamma)q(x)} \right]
\]

\[ p(x \mid \theta) \text{ is a linear dynamical system} \]
\[ p(y \mid x, \gamma) \text{ is a neural network decoder} \]
\[ p(\theta) \text{ is a conjugate prior, } p(\gamma) \text{ is generic} \]

\[ \eta_x^*(\eta_\theta, \eta_\gamma) \triangleq \arg \max_{\eta_x} \mathcal{L}(\eta_\theta, \eta_\gamma, \eta_x) \]

\[ \mathcal{L}_{\text{SVI}}(\eta_\theta, \eta_\gamma) \triangleq \mathcal{L}(\eta_\theta, \eta_\gamma, \eta_x^*(\eta_\theta, \eta_\gamma)) \]
Variational Auto-Encoder

\[ q^*(x) \triangleq \mathcal{N}(x \mid \mu(y; \phi), \Sigma(y; \phi)) \]
\[ \hat{\mathcal{L}}(\eta_\theta, \eta_x, \phi) \triangleq \mathbb{E}_{q(\theta)q(\gamma)q(x)} \left[ \log \frac{p(\theta, \gamma, x) \exp\{\psi(x; y, \phi)\}}{q(\theta)q(\gamma)q(x)} \right] \]

\[ \eta^*_x(\eta_\theta, \phi) \triangleq \arg \max_{\eta_x} \hat{\mathcal{L}}(\eta_\theta, \eta_x, \phi) \]

\[ \mathcal{L}_{SVAE}(\eta_\theta, \eta_\gamma, \phi) \triangleq \mathcal{L}(\eta_\theta, \eta_\gamma, \eta^*_x(\eta_\theta, \phi)) \]
Algorithm

Step 1: apply recognition network

Step 2: run fast PGM algorithms

Step 3: sample, compute flat grads

Step 4: compute natural gradient
Figure 6: Predictions from an LDS SVAE fit to depth video. In each panel, the top is a sampled prediction and the bottom is real data. The model is conditioned on observations to the left of the line.

Figure 7: Examples of behavior states inferred from depth video. Each frame sequence is padded on both sides, with a square in the lower-right of a frame depicting when the state is the most probable.
\[ q^*(x) \triangleq \arg \max_{q(x)} \mathcal{L}[q(\theta)q(x)] \]

\[ q^*(x) \triangleq \mathcal{N}(x | \mu(y; \phi), \Sigma(y; \phi)) \]

- expensive for general obs.
- optimal local factor
- exploits conj. graph structure
+ arbitrary inference queries

+ fast for general obs.
- suboptimal local inference
- \( \phi \) does all local inference
- limited inference queries
Conclusion

Possible applications

Reinforcement Learning and Bandits
Semi-supervised Learning