Autoregressive Models[1]

[1] Based on the lecture notes developed at Stanford by Stefano Ermon and Aditya Grover.

Sequence-Based Model I

- Let us look at constructing models from data D consisting of a sequence of vectors $\langle \mathbf{x_1}, \mathbf{x_2}, \ldots \mathbf{x_N} \rangle$. For simplicity, we will assume each of the the data instances to consists of values for n Boolean-valued r.v's $X_1, X_2, \ldots X_n$.
- By the chain rule, for any data point $\mathbf{x}$:

$$Pr(\mathbf{X}) = \prod_{i=1}^{d} P(X_i|X_1, X_2, \ldots, X_{i-1}) = \prod_{i=1}^{n} Pr(X_i|\mathbf{X}_{<i})$$

  where $\mathbf{X}_{<i}$ is short for the values of r.v's $X_1, X_2, \ldots X_{i-1}$
- Can you draw the factorisation of the Chain Rule as a Bayesian Network?

Sequence-Based Model II

- This kind of BN makes no conditional independence assumptions, and is said to be *autoregressive*
- Once we pick an ordering of the X_i, and we have the conditional probability tables, then we can represent any joint distribution over the X_i
  - How much space will be needed to represent the joint distribution in this way?
  - QUESTION: The number of parameters to be estimated is $O(\ldots)$
  - Can we make this more compact, using a functional representation of the CPT, rather than a table?

Sequence-Based Model III

- In an *autoregressive generative model* the conditionals are specified as functions with a fixed number of parameters

$$Pr(X_i = 1|\mathbf{X}_{<i}) \;=\; Bern(f_i(X_1, X_2, \ldots, X_{i-1}))$$

  where f_i has θ_i parameters. If $X \sim Bern(p)$, then $Pr(X = 1) = p$ and $E[X] = p$. The function f_i is therefore sometimes called the *mean function* for X_i

$$f_i : \{0, 1\}^{i-1} \mapsto [0, 1]$$

- So, the conditional probability for X_i is now restricted to being represented by a function f_i with θ_i parameters
  - QUESTION: Can all possible functions be represented in this way?

Sequence-Based Model IV

- A simple model: f_i is a logistic function. More generally:

$$f_i(X_1, \ldots X_{i-1}) \;=\; \sigma(\sum_{k=0}^{i-1} \alpha_k^{(i)} X_k$$

(where $X_0 = 1$), and σ is a sigmoid (S-shaped) function.
  - The logistic function is the sigmoid $\frac{1}{1+e^{-x}}$
  - The parameters θ_i are $\alpha_0^{(i)}, \ldots \alpha_{i-1}^{(i)}$
  - QUESTION: The parameters needed in this autorregressive model is $O(\ldots)$
- Issue with NADE: the model depends on the ordering of the X_i. What can be done to mitigate this?

Sequence-Based Model V

- Think about what needs to be done if the X_i are real-valued r.v's
  - HINT: Consider how you could model $Pr(X_i|\mathbf{X}_{<i})$ when $\mathbf{X}_{<i}$ is a real valued vector.
- What (additional) parameters will need to be estimated now?

More Powerful Autoregression I

- A neural network can be used for a more powerful representation of the mean function f_i
  - NADE: Neural Autoregressive Density Estimation
- Example, use a NN for each $Pr(x_i|\cdot)$ with one hidden layer

$$\mathbf{h}_i \;=\; \sigma(W_i \mathbf{X}_{<i} + \mathbf{c}_i)$$

and

$$f_i(X_1, X_2, \ldots, X_i) \;=\; \sigma(\alpha^{((i)}\mathbf{h}_i + b_i)$$

- The parameters are now the weights W_i, $\mathbf{c}_i$, $\alpha^{(i)}$and b_i
- Assuming d hidden units in each layer:
  - QUESTION: $W_i \in \Re^{\cdots}$
  - QUESTION: $\mathbf{c}_i \in \Re^{\cdots}$
  - QUESTION: $\alpha^{(i)} \in \Re^{\cdots}$
  - QUESTION: $b_i \in \Re^{\cdots}$

More Powerful Autoregression II

- QUESTION: The total parameters to be estimated is now $O(\ldots)$
- NADE is a refinement of this approach, with some reduction in parameters to be estimated. The reduction is achieved by sharing parameters:

$$\mathbf{h}_i = \sigma(W\mathbf{X}_{<i} + \mathbf{c})$$

- QUESTION: The number of parameters estimated by NADE is $O(\ldots)$

Estimating parameters in a NADE model I

- Use the usual optimisation approach: (a) objective function to maximise is the likelihood of the data using the model; and (b) use SGD to find a (local) maximum
  - When constructing an approximation to the observed distribution P using a model Q, using maximising the likelihood of the data using Q is equivalent to minimising the quantity known as the (Forward) *Kullback-Leibler Divergence*

$$KLD(P, Q) = E_{X \sim P} \log \left[\frac{P(x)}{Q(x)} \right]$$

  - Intuitively, the Forward KLD measures how close Q(x) is to P(x) when the x are values of an r.v distributed according to P

Estimating parameters in a NADE model II

- In our case, Q is obtained from a model with parameters θ. That is, $Q(x)$ is really $Q(x;\theta)$. Suppose we want find the value of θ that minimises $KLD(P,Q)$:

$$\arg\min_{\theta} KLD(P, Q(x;\theta)) = \arg\min_{\theta} E_{X\sim P}\ (\log[P(x)] - \log[Q(x;\theta)])$$

Simplifying:

$$\arg\min_{\theta} KLD(P, Q(x;\theta)) = \arg\min_{\theta} -E_{X\sim P}\ \log Q(x;\theta)$$

or

$$\arg\min_{\theta} KLD(P, Q(x;\theta)) = \arg\max_{\theta} E_{X\sim P}\ \log Q(x;\theta)$$

Now, we are left with having to work out the expectation in the rhs. We can do this using a Monte Carlo estimate,

Estimating parameters in a NADE model III

assuming the data are a sample from P. The optimisation problem is then

$$\arg\max_{\theta} \frac{I}{N} \sum_{i=1}^{N} \log Q(x_i; \theta)$$

which is maximising the (log) likelihood of the data using the model

- Finding a good value for the θ is can then be done using the usual gradient descent update rule:

$$\theta^{(k+1)} = \theta^{(k)} + \eta_{(k)}$$

This is usually computed using a sample of instances from D, or a mini-batch. Iterative update stops when there is no change in the value of the (log) likelihood of a "validation" set.