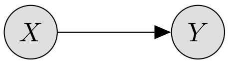
Master Project: Notes

Tristan Guigue

June 20, 2017

1 Mutual Information and Maximum Likelihood in Supervised Setting

We consider the simplest graphical model in a supervised learning setting:



We would like to check that maximising the mutual information between X and Y is equivalent to maximum likelihood:

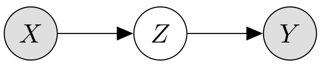
$$
\begin{aligned}
E_{X,Y}(\log p(y|x)) &= \int p(x,y)\, p(y|x)\, dx\, dy & (1)\\
&= -H(Y|X) & (2)
\end{aligned}
$$

$$
I(X,Y) = H(Y) - H(Y|X) \tag{3}
$$

So if we consider the entropy of the labels fixed, maximising the mutual information is indeed equivalent to maximum likelihood.

2 Mutual Information Maximisation in Stochastic Feed-forward Network

We consider a graphical model in a supervised learning setting:



Where X is our observed data and Y our labels. Z is a stochastic representation of our data.

2.1 Maximum Likelihood

We try to maximise the likelihood $\log p(y|x)$

2.1.1 Naive approach

The simplest way to do this would be to sample directly from the likelihood:

$$
\begin{aligned}
p(y|x) &= \int p(y,z|x)dz && (4)\\
&= \int p(y|z)p(z|x)dz && (5)\\
&= E_{Z|X}[p(y|z)] && (6)
\end{aligned}
$$

For example:

$$p(y|z) = S(y|Wz + b)$$

were S is the softmax function. And:

$$p(z|x) \sim \mathcal{N}(z|f^{\mu}(x), f^{\Sigma}(x))$$

where f^{μ} and f^{Σ} are multi-layer perceptrons.

We can approximate

$$p(y|x) \approx \frac{1}{M}\sum_{m=1}^{M} p(y|z^{(m)}) \tag{7}$$

Where $z^{(m)} \sim p(z|x)$

However this might lead to a high variance because we can't be sure that our draw from $p(z|x)$ will contribute significantly to our estimate $p(y|x)$.

2.1.2 Stochastic Gradient Variational Bayes

To prevent this, we use a variational lower bound on the log likelihood:

$$
\begin{aligned}
\log p(y|x) &= \log(\int p(y,z|x)dz) && (8)\\
&= \log(\int q(z)\frac{p(y,z|x)}{q(z)}) && (9)\\
&\geq \int q(z)\log\frac{p(y,z|x)}{q(z)} && (10)
\end{aligned}
$$

Using Jensen's inequality.

$$
\begin{aligned}
\log p(y|x) &\geq \int q(z)\log\frac{p(z|x)p(y|z)}{q(z)} && (11)\\
&= \int q(z)\log\frac{p(z|x)}{q(z)} + \int q(z)\log p(y|z) && (12)\\
&= E_q[p(y|z)] - KL[q(z)||p(z|x)] && (13)\\
&\approx \frac{1}{M}\sum_{m=1}^{M} p(y|z^{(m)}) - KL[q(z)||p(z|x)] && (14)
\end{aligned}
$$

Where $z^{(m)} \sim q(z)$ and assuming the KL divergence can be computed analytically.

So we try to maximise the expected value of decoder given z sampled from a distribution parametrised by the encoder while minimising the KL divergence between $q(z)$ and $p(z|x)$.

2.2 Maximum Mutual Information

We would like to compare this result to the maximisation of the mutual information between Z and Y:

$$\begin{aligned}
I(Z,Y) &= \int p(y,z) \log \frac{p(z,y)}{p(z)p(y)} dy\, dz && (15)\\
&= \int p(y,z) \log \frac{p(y|z)}{p(y)} dy\, dz && (16)\\
&= \int p(y,z) \log p(y|z) dy\, dz - H(y) && (17)
\end{aligned}$$

We ignore the entropy of the labels as it can't be maximised. We get:

$$\begin{aligned}
I(Z,Y) &= \int p(x,y,z) \log p(y|z) dx\, dy\, dz && (18)\\
&= \int p(z|x,y)p(x,y) \log p(y|z) dx\, dy\, dz && (19)
\end{aligned}$$

We can approximate the joint distribution of the data and the labels by:

$$p(x,y) \approx \frac{1}{N} \sum_n \delta_{x_n}(x)\delta_{y_n}(y)$$

So we get:

$$I(Z,Y) \approx \frac{1}{N} \int p(z|x^{(n)}, y^{(n)}) \log p(y^{(n)}|z)$$

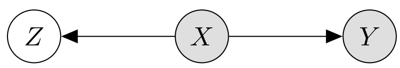
As we can see this is not equivalent to the maximum likelihood. In particular we have no way to easily get the posterior over z.

3 Information Bottleneck in Recurrent Neural Networks

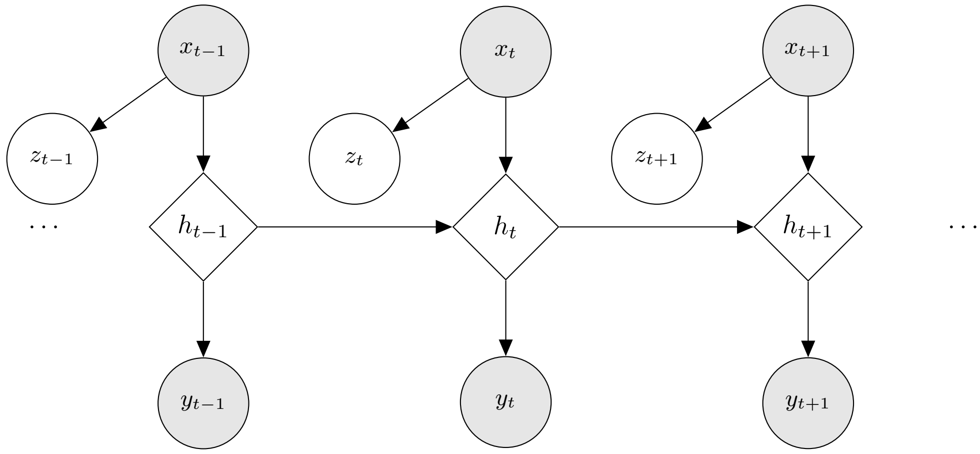
For a stochastic feed-forward network we try to be maximally compressive on the input while being maximally informative on the output. In mutual information this translates to the objective funtion:

$$J = I(Z,Y) - \beta I(X,Z)$$

under the graphical model



Applying this idea to a recurrent network we use the following graphical model:



There are several mutual information objectives possible: For example we can take

$$J = \sum_t I(Z_t, Y_t) - \beta I(X_{1:t}, Z_t)$$

but we could also take

$$J = \sum_t I(Z_t, Y_t) + I(Z_t, Y_{t+1:N}) - \beta I(X_{1:t}, Z_t)$$

Using the results from Alemi et al. [1] for the stochastic feed forward network, the objective function becomes:

$$J = \frac{1}{TN} \sum_{t=1}^{T} \sum_{n=1}^{N} \mathbb{E}_{\epsilon \sim p(\epsilon)}[-\log q(y_{n,t}|f(x_{n,t}, \epsilon))] + \beta KL[p(Z|x_{n,t}), r(Z)]$$

Note that we use the reparametrisation trick: $z = \mu + \sigma\epsilon$ where $\epsilon \sim \mathcal{N}(0, I)$ to be able to propagate the gradient through the encoder.

As previously we have:

$$p(z_t|x_t) \sim \mathcal{N}(z_t|f^{\mu}(x_t), f^{\Sigma}(x_t))$$

where f^{μ} and f^{Σ} are multi-layer perceptrons. The output of the perceptrons is divided in two, the first half gives the mean, we apply a softplus to the second half which gives the diagonal standard deviation terms.

3.1 Results

We compared the behavior of networks run with different values of beta for both feedforward and recurrent networks.

3.1.1 Feed Forward Network

Using the same parameters as in the Deep Variational Information Bottleneck: bottleneck layer of dimension K = 256 and multilayer perceptron with two hidden layer of dimensions 1024.

β	0	10^{-3}
error	2.38	1.43

3.1.2 Recurrent Network

β	0	10^{-3}
error	3.83	3.95

4 Normalising Flow and Gradient Estimator without score function

The variational lower bound can be writen as:

$$\mathcal{L} = \mathbb{E}_{z \sim q}[\log p(x, z) - \log q_\phi(z|x)]$$

Taking a single Monte Carlo sample this becomes:

$$\mathcal{L}_{mc} = \log p(x, z) - \log q_\phi(z|x)$$

The gradient with respect to the variational parameter is then:

$$\Delta_{TD}(\epsilon, \phi) = \Delta_z[\log p(z|x) - \log q_\phi(z|x)]\Delta_\phi t(\epsilon, \phi) - \Delta \log q_\phi(z|x) \tag{20}$$

Removing the score function we get another unbiased estimator that has lower variance when q is closed to the posterior:

$$\Delta_{PD}(\epsilon, \phi) = \Delta_z[\log p(z|x) - \log q_\phi(z|x)]\Delta_\phi t(\epsilon, \phi)$$

A normalising flow is a powerful representation of the posterior though a succession of invertible transformation.

$$z_K = f_K \circ ... \circ f_1(z_0)$$

And their normalised distribution:

$$q_K(z_K) = q_0(z_0) \prod_{k=1}^{K} |\frac{\partial f_k}{\partial z_{k-1}}|^{-1}$$

$$\log q_K(z_K) = \log q_0(z_0) - \sum_{k=1}^{K} \log|\frac{\partial f_k}{\partial z_{k-1}}|$$

With

$$q_0(z_0) \sim \mathcal{N}(q_0|f^\mu(x), f^\Sigma(x))$$

The free energy lower bound is now:

$$\begin{aligned}\mathcal{L} &= \mathbb{E}_{z\sim q}[\log p(x,z) - \log q_\phi(z|x)] && (21)\\ &= \mathbb{E}_{z\sim q_0}[\log p(x,z_K) - \log q_K(z_K)] && (22)\end{aligned}$$

Sampling from q_0 we get:

$$\mathcal{L}_{mc} = \log p(x,z_K) - \log q_K(z_K)$$

And the gradient:

$$\begin{aligned}\Delta_\phi \mathcal{L}_{mc} &= \frac{\partial}{\partial\phi}\log p(x,z_K) - \frac{\partial}{\partial\phi}\log q_K(z_K) && (23)\\ &= \Delta_{z_K}[\log p(x,z_K) - \log q_K(z_K)]\Delta_\phi z_K - \cancel{\Delta_\phi \log q_K(z_K)} && (24)\end{aligned}$$

We'd like to use a neural network with leaky ReLu activation for f since they have constant gradient. However the change of variable

$$q(z') = q(f^{-1}(z'))|\frac{\partial f}{\partial z}|^{-1}$$

is only valid if f is differentiable which is not the case here.

References

[1] Alexander A. Alemi, Ian Fischer, Joshua V. Dillon, Kevin Murphy. *Deep Variational Information Bottleneck*. ICLR, 2017.