Bayesian learning lab

Mihaela Rosca

Variational inference

Variational inference

Inferring posterior distributions when exact inference is not possible (intractable).

Posterior distributions over what?

Posterior distributions over what?

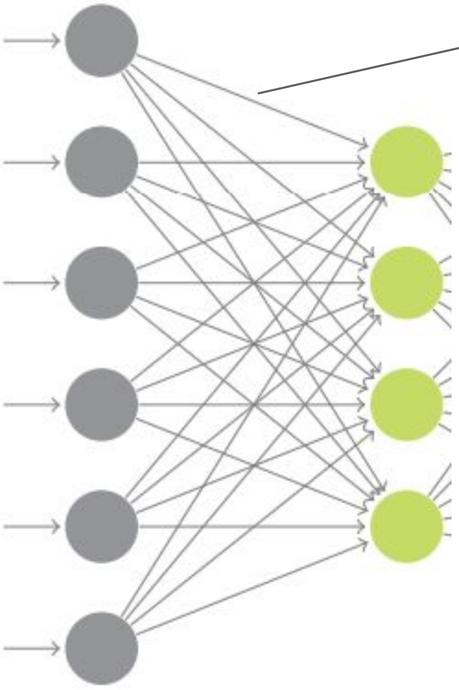
Model variables

Learn a posterior distribution over model variables.

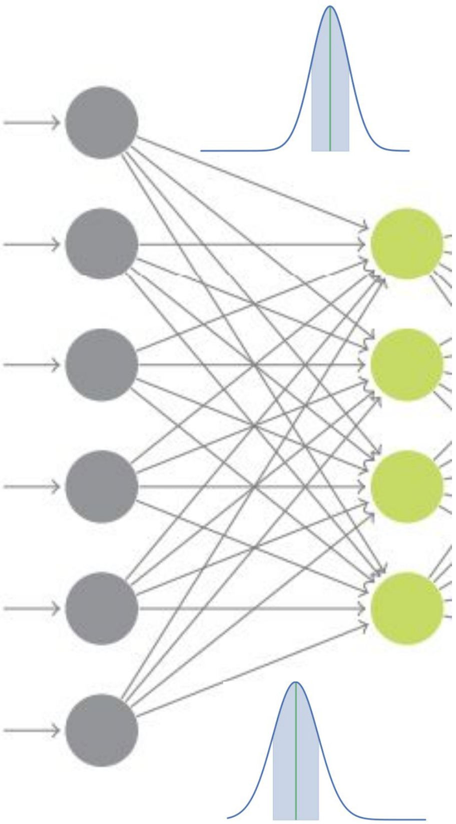
Latent variables

Learn a distribution over latent variables.

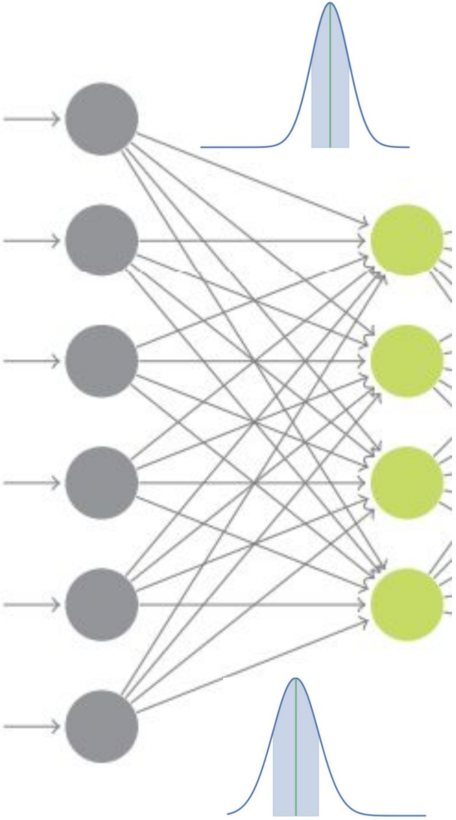
Posterior distributions over model variables



learned parameters



learned distribution over parameters.



we will model each parameter independently.

How?

1. Choose your model
2. Derive an objective for your posterior (evidence lower bound)
3. Derive gradients to update your posterior (can be tricky)
4. Learn (backprop)

Step 1

Model case study: Bayesian Logistic Regression

Problem: supervised learning via maximum likelihood.

$$\mathbb{E}_{p^*(x,y)}\left[p_\theta(y|x)\right]$$

$$p_\theta(y|x) = \int p(y|x,\theta)p(\theta)d\theta$$

Step 1

Model case study: Bayesian Logistic Regression

Problem: supervised learning via maximum likelihood.

$$\mathbb{E}_{p^*(x,y)}[p_\theta(y|x)]$$

$$p_\theta(y|x) = \int p(y|x,\theta)p(\theta)d\theta$$

Step 1

Model case study: Bayesian Logistic Regression

$$p_\theta(y|x) = \int p(y|x,\theta)p(\theta)d\theta$$

$$p(y|x,\theta) = y\sigma(w^T x + b) + (1-y)(1-\sigma(w^T x + b))$$

$$\theta = \{w, b\}$$

Step 2

Deriving the evidence lower bound

$$\mathbb{E}_{p^*(x,y)} \log p_w(y|x) =$$

$$\mathbb{E}_{p^*(x,y)} \log \int p(y|x,w)p(w)\delta w =$$

$$\mathbb{E}_{p^*(x,y)} \log \int p(y|x,w)p(w)\frac{q(w)}{q(w)}\delta w \geq$$

Jensen Ineq.

$$\mathbb{E}_{p^*(x,y)}\mathbb{E}_{q(w)} \log \left[p(y|x,w)\frac{p(w)}{q(w)}\right] =$$

$$\mathbb{E}_{q(w)}\mathbb{E}_{p^*(x,y)} \log p(y|x,w) - KL(q(w)||p(w))$$

ELBO

Step 3

Compute posterior gradients

$$\nabla_\theta \left[\mathbb{E}_{q_\theta(w)}\mathbb{E}_{p^*(x,y)} \log p(y|x,w) - KL(q_\theta(w)||p(w))\right]$$

Cannot do the usual trick of putting the gradient inside the expectation, since the expectation depends on the parameters.

In our case, we can compute it in closed form, do standard backprop.

Step 3

Compute posterior gradients - REINFORCE

$$\nabla_\theta \mathbb{E}_{q_\theta(w)} \mathbb{E}_{p^*(x,y)} \log p(y|x,w) =$$

$$\nabla_\theta \int q_\theta(w) \mathbb{E}_{p^*(x,y)} \log p(y|x,w) \delta w =$$

$$\int \nabla_\theta \left[q_\theta(w) \mathbb{E}_{p^*(x,y)} \log p(y|x,w) \right] \delta w =$$

Change gradients and integral order

$$\int \nabla_\theta \left[q_\theta(w) \right] \mathbb{E}_{p^*(x,y)} \log p(y|x,w) \delta w =$$

$$\int q_\theta(w) \nabla_\theta \left[\log q_\theta(w) \right] \mathbb{E}_{p^*(x,y)} \log p(y|x,w) \delta w =$$

Log ratio trick

$$\mathbb{E}_{q_\theta(w)} \left[\nabla_\theta \log q_\theta(w) \mathbb{E}_{p^*(x,y)} \log p(y|x,w) \right]$$

Back to expectations

Step 3

Compute posterior gradients - REINFORCE

$$\nabla_\theta \mathbb{E}_{q_\theta(w)} \mathbb{E}_{p^*(x,y)} \log p(y|x,w) =$$

$$\nabla_\theta \int q_\theta(w) \mathbb{E}_{p^*(x,y)} \log p(y|x,w) \delta w =$$

$$\int \nabla_\theta \left[q_\theta(w) \mathbb{E}_{p^*(x,y)} \log p(y|x,w) \right] \delta w =$$

Change gradients and integral order

$$\int \nabla_\theta \left[q_\theta(w) \right] \mathbb{E}_{p^*(x,y)} \log p(y|x,w) \delta w =$$

$$\int q_\theta(w) \nabla_\theta \left[\log q_\theta(w) \right] \mathbb{E}_{p^*(x,y)} \log p(y|x,w) \delta w =$$

Log ratio trick

$$\mathbb{E}_{q_\theta(w)} \left[\nabla_\theta \log q_\theta(w) \mathbb{E}_{p^*(x,y)} \log p(y|x,w) \right]$$

Back to expectations

Estimate integral using Monte Carlo estimation (by sampling from the posterior distribution)

Step 3

Compute posterior gradients - REINFORCE

$$\nabla_\theta \mathbb{E}_{q_\theta(w)} \mathbb{E}_{p^*(x,y)} \ldots g\, p(y|x, w) =$$

$$\nabla_\theta \int \ldots (w) \mathbb{E}_p \ldots (y| \ldots w)\delta w =$$

$$\int \nabla \ldots)] \delta w =$$

Change gradients and integral order

$$\int \nabla \ldots w)\delta w =$$

$$\int q_\theta(w) \nabla_\theta [\ldots \log p \ldots y|x, w)\delta w =$$

Log ratio trick

$$\mathbb{E}_{q_\theta(w)} \left[\nabla_\theta \ldots g\, q_\theta(w) \mathbb{E}_{p^*(x,y)} \log p(y|x, w)\right]$$

Back to expectations

Same as in the RL colab!

Estimate integral using Monte Carlo estimation (by sampling from the posterior distribution)

Step 3

Derive posterior gradients - Reparametrization

$$z \sim N(\mu, \sigma), z = \mu + \epsilon\sigma, \text{with } \epsilon \sim N(0, 1)$$

$$\nabla_\theta \mathbb{E}_{q_\theta(w)} \mathbb{E}_{p^*(x,y)} \log p(y|x, w) =$$
$$\nabla_\theta \mathbb{E}_{p(\epsilon)} \mathbb{E}_{p^*(x,y)} \log p(y|x, \mu + \epsilon\sigma) =$$
$$\mathbb{E}_{p(\epsilon)} \nabla_\theta \mathbb{E}_{p^*(x,y)} \log p(y|x, \mu + \epsilon\sigma)$$

Step 3

Derive posterior gradients - Reparametrization

$$z \sim N(\mu, \sigma), z = \mu + \epsilon\sigma, \text{with } \epsilon \sim N(0,1)$$

$$\nabla_\theta \mathbb{E}_{q_\theta(w)} \mathbb{E}_{p^*(x,y)} \log p(y|x,w) =$$
$$\nabla_\theta \mathbb{E}_{p(\epsilon)} \mathbb{E}_{p^*(x,y)} \log p(y|x,\mu + \epsilon\sigma) =$$
$$\mathbb{E}_{p(\epsilon)} \nabla_\theta \mathbb{E}_{p^*(x,y)} \log p(y|x,\mu + \epsilon\sigma)$$

Estimate integral using Monte Carlo estimation (by sampling from p(ε)).

Reinforce

- no assumptions about the cost function
- posterior log density needs to be differentiable
- unbiased estimator
  - estimates the true gradient
- consistent estimator
  - the more samples we use, the better the estimator is

Reparametrization

- posterior = Gaussian
- cost function needs to be differentiable
- unbiased estimator
  - estimates the true gradient
- consistent estimator
  - the more samples we use, the better the estimator is

More here: https://arxiv.org/abs/1906.10652

Step 4

Learning - putting it all together

- Define your model and define your posterior
- Write your objective (ELBO)
- Choose your gradient estimator
    - Might have to change your loss to rely on automatic differentiation in TF
        - surrogate loss for Reinforce
        - reparametrization is default in TF
- Use stochastic gradient descent to learn the model

Step 4

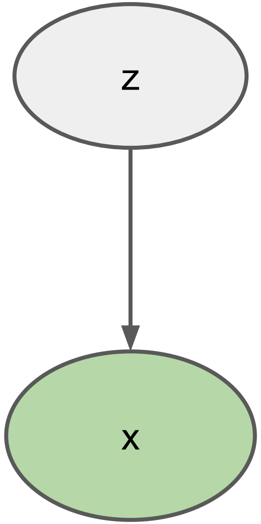
Learning - putting it all together

- Define your model and define your posterior
- Write your objective (ELBO)
- Choose your gradient estimator
  - Might have to change your loss to rely on automatic differentiation in TF
    - surrogate loss for Reinforce
    - reparametrization is default in TF
- Use stochastic gradient descent to learn the model

Colab 1

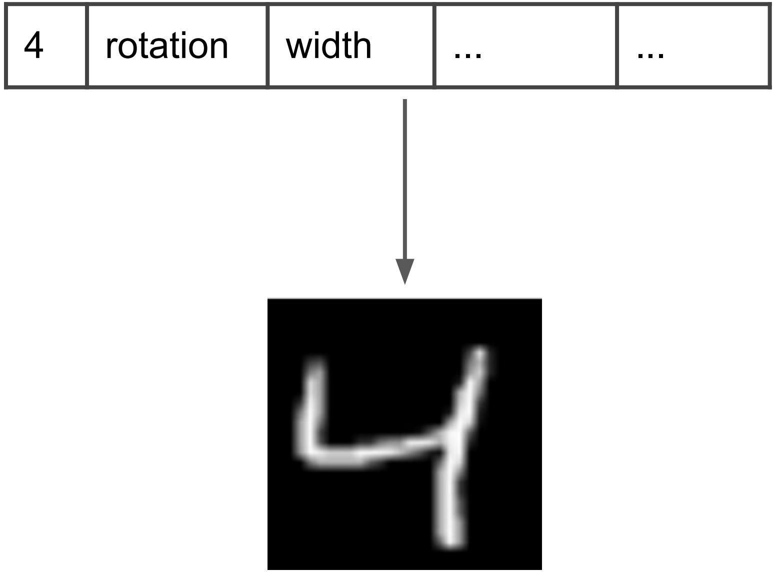
Posterior distributions over latent variables

Leverage underlying data structure in generative process.

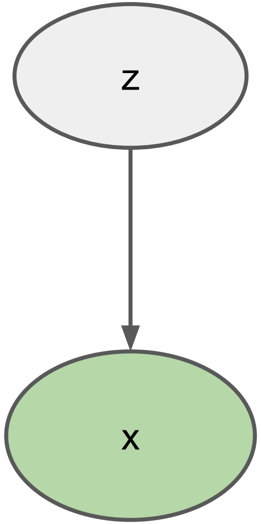


z = latents
x = observed

z = latents



x = observation



z = latents
x = observed

$$p_{\boldsymbol{\theta}}(\mathbf{x}) = \int p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})p(\mathbf{z})d\mathbf{z}$$

prior

True posterior distribution: p(z|x)

Learned posterior distribution: $q_\eta(z|x)$

$$\log p_{\boldsymbol{\theta}}(\mathbf{x}) = \log \int p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})p(\mathbf{z})d\mathbf{z} \geq \underbrace{\mathbb{E}_{q_\eta(\mathbf{z}|\mathbf{x})}[\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})]}_{\text{Encode information about x - make sampling efficient}} - \underbrace{\mathrm{KL}[q_\eta(\mathbf{z}|\mathbf{x})||p(\mathbf{z})]}_{\text{Stay close to the prior}}$$

Equality when $q_\eta(z|x) = p(z|x)$

Learned posterior distribution: $q_\eta(z|x)$

$$\log p_{\boldsymbol{\theta}}(\mathbf{x}) = \log \int p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})p(\mathbf{z})d\mathbf{z} \geq \underbrace{\mathbb{E}_{q_\eta(\mathbf{z}|\mathbf{x})}[\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})]}_{\text{reconstruction loss}} - \underbrace{\mathrm{KL}[q_\eta(\mathbf{z}|\mathbf{x})||p(\mathbf{z})]}_{\text{KL loss}}$$

How?

1. Choose your model (with latent variables)
2. Derive an objective for your posterior (evidence lower bound)
3. Derive gradients to update your posterior (can be tricky)
4. Learn (backprop)

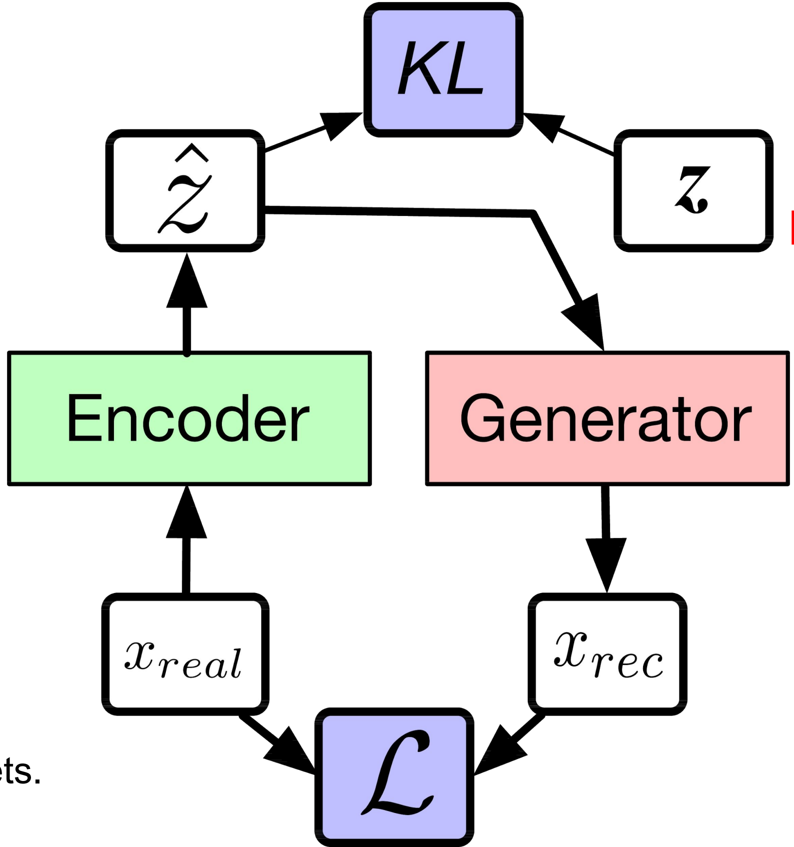
How?

1. Choose your model (with latent variables)
2. Derive an objective for your posterior (evidence lower bound)
3. Derive gradients to update your posterior (can be tricky)
4. Learn (backprop)

Same as learning posterior over model variables.

Case study: Variational autoencoders

Learned posterior distribution: $q_\eta(z|x)$



Learned model: $p_\theta(z|x)$

Encoder and generator are neural nets.

Step 2

Learning the posterior

Optimize the evidence lower bound:

$$\mathbb{E}_{p^*(x)}\left[\mathbb{E}_{q_\eta(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] - KL[q_\eta(\mathbf{z}|\mathbf{x})]||[p(\mathbf{z})]\right]$$

ELBO

Step 3

Gradient estimation

$$\mathbb{E}_{p^*(x)}\left[\mathbb{E}_{q_\eta(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] - KL[q_\eta(\mathbf{z}|\mathbf{x})]||[p(\mathbf{z})]\right]$$

An expectation with respect to the posterior. As before, we need to find a suitable gradient estimator.

For VAEs, we will use a Gaussian posterior and use the reparametrization gradient estimator.

Step 4

Learning - putting it all together

- Define your model and define your posterior
  - posterior = Gaussian, the output of a NN encoder
  - model distribution = Bernoulli, the output of a NN decoder
- Write your objective (ELBO)
- Use reparametrization for gradient estimation (defalt in TF)
- Use stochastic gradient descent to learn the model

Step 4

Learning - putting it all together

- Define your model and define your posterior
  - posterior = Gaussian, the output of a NN encoder
  - model distribution = Bernoulli, the output of a NN decoder
- Write your objective (ELBO)
- Use reparametrization for gradient estimation (defalt in TF)
- Use stochastic gradient descent to learn the model

Colab 2

Additional task: use constrained optimization.

Annealing coefficients

$$\mathbb{E}_{p^*(x)}\left[\mathbb{E}_{q_\eta(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] - KL[q_\eta(\mathbf{z}|\mathbf{x})]||[p(\mathbf{z})]\right]$$

$$\mathbb{E}_{p^*(x)}\left[\mathbb{E}_{q_\eta(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] - \beta KL[q_\eta(\mathbf{z}|\mathbf{x})]||[p(\mathbf{z})]\right]$$

Annealing coefficients

$$\mathbb{E}_{p^*(x)}\left[\mathbb{E}_{q_\eta(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] - KL[q_\eta(\mathbf{z}|\mathbf{x})]||[p(\mathbf{z})]\right]$$

$$\mathbb{E}_{p^*(x)}\left[\mathbb{E}_{q_\eta(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] - \beta KL[q_\eta(\mathbf{z}|\mathbf{x})]||[p(\mathbf{z})]\right]$$

Learning coefficients

$$\mathbb{E}_{p^*(x)}\left[\mathbb{E}_{q_\eta(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] - KL[q_\eta(\mathbf{z}|\mathbf{x})]||[p(\mathbf{z})]\right]$$

$$\min_{\theta,\eta} \mathbb{E}_{p^*(x)} KL[q_\eta(\mathbf{z}|\mathbf{x})]||[p(\mathbf{z})]$$

$$st. \mathbb{E}_{p^*(x)}\mathbb{E}_{q_\eta(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] > \alpha$$

Learning coefficients

$$\mathbb{E}_{p^*(x)}\left[\mathbb{E}_{q_\eta(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] - KL[q_\eta(\mathbf{z}|\mathbf{x})]||[p(\mathbf{z})]\right]$$

$$\min_{\theta,\eta} \mathbb{E}_{p^*(x)} KL[q_\eta(\mathbf{z}|\mathbf{x})]||[p(\mathbf{z})] + \lambda\left(\alpha - \mathbb{E}_{p^*(x)}\mathbb{E}_{q_\eta(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})]\right)$$

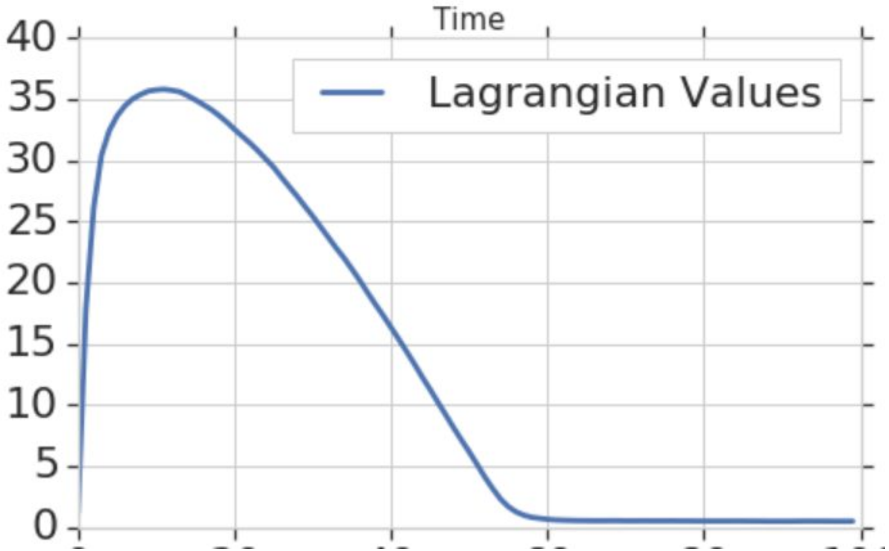
Learning coefficients

$$\min_{\theta,\eta} \mathbb{E}_{p^*(x)} KL[q_\eta(\mathbf{z}|\mathbf{x})]||[p(\mathbf{z})] + \lambda \left(\alpha - \mathbb{E}_{p^*(x)} \mathbb{E}_{q_\eta(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})]\right)$$

$$\max_{\lambda} \lambda \left(\alpha - \mathbb{E}_{p^*(x)} \mathbb{E}_{q_\eta(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})]\right)$$

Learning coefficients

$$\max_{\lambda} \lambda \left(\alpha - \mathbb{E}_{p^*(x)} \mathbb{E}_{q_\eta(\mathbf{z}|\mathbf{x})} [\log p_\theta(\mathbf{x}|\mathbf{z})] \right)$$



Further reading

- Auto-Encoding Variational Bayes
- Monte Carlo Gradient Estimation in Machine Learning
- Variational Inference: A Review for Statisticians
- Taming VAEs