ESE 546, FALL 2020

HOMEWORK 5

SHEIL SARDA [SHEILS@SEAS]

Solution 1 (Time spent: 2 hour)**.** (1) To Prove:

$$\mathrm{KL}(q\|p) = \beta \sum_{w\in\mathcal{W}} q(w)\Phi(w) + \sum_{w\in\mathcal{W}} q(w)\log q(w) + \log Z(\beta)$$

Starting with the definition of KL Divergence:

$$\begin{aligned}
\mathrm{KL}(q\|p) &= \sum_{w\in\mathcal{W}} q(w)\frac{q(w)}{p(w)} \\
&= \sum_{w\in\mathcal{W}} q(w)\frac{Zq(w)}{e^{-\beta\Phi(w)}} && \text{Since } p(w) = \frac{e^{\beta\Phi(w)}}{Z} \\
&= \sum_{w\in\mathcal{W}} q(w)\left[log(Z) + log(q(w)) + \beta\Phi(w)\right] \\
&= log(Z)\sum_{w\in\mathcal{W}} q(w) + \sum_{w\in\mathcal{W}} q(w)log(q(w)) + \beta\sum_{w\in\mathcal{W}} q(w)\Phi(w) \\
&= log(Z) + \sum_{w\in\mathcal{W}} q(w)log(q(w)) + \beta\sum_{w\in\mathcal{W}} q(w)\Phi(w) && \text{Since } \sum_{w\in\mathcal{W}} q(w) = 1
\end{aligned}$$

$\square$

(2) To Prove:

$$\mathop{\mathbb{E}}_{w\sim p(w)}[\Phi(w)] = -\frac{\partial \log Z(\beta)}{\partial \beta}$$

Starting from the LHS:

$$\mathop{\mathbb{E}}_{w\sim p(w)}[\Phi(w)] = \sum_{w\in\mathcal{W}} p(w)\Phi(w)$$

Substituting for $\Phi(w)$:

$$1 = \sum_{w\in\mathcal{W}} p(w) = \sum_{w\in\mathcal{W}} \frac{e^{\beta\Phi(w)}}{Z}$$

$$\implies Z = \sum_{w\in\mathcal{W}} e^{\beta\Phi(w)}$$

$$\implies \frac{\partial Z}{\partial \beta} = \sum_{w\in\mathcal{W}} e^{\beta\Phi(w)}\,(-\Phi(w))$$

By definition of $p(w)$, we also know:

$$p(w) = \frac{e^{\beta\Phi(w)}}{Z} \implies Z \times p(w) = e^{\beta\Phi(w)} \tag{1}$$

$$log(Z \times p(w)) = -\beta\Phi(w) \implies \Phi(w) = \frac{log(Z \times p(w))}{-\beta} \tag{2}$$

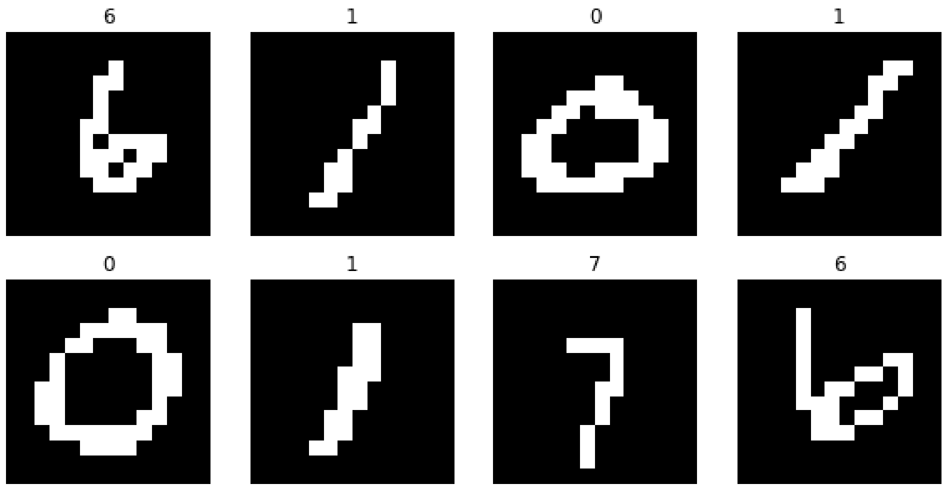
Substituting (1) + taking partial w.r.t. β on both sides:

$$\frac{\partial logZ}{\partial \beta} = \sum_{w\in\mathcal{W}} Z \times p(w)\frac{log(Z \times p(w))}{\beta}$$

$$\frac{\partial logZ}{\partial \beta} = -Z\sum_{w\in\mathcal{W}} \frac{log(Z \times p(w))}{-\beta} \times p(w)$$

Solution 2 (Time spent: 10 hours)**.** Plots from the Jupyter notebook attached here for reference.

(1) Plot of Binarized and 14*14 subsampled images of MNIST



(2) Encoder and Decoder classes.

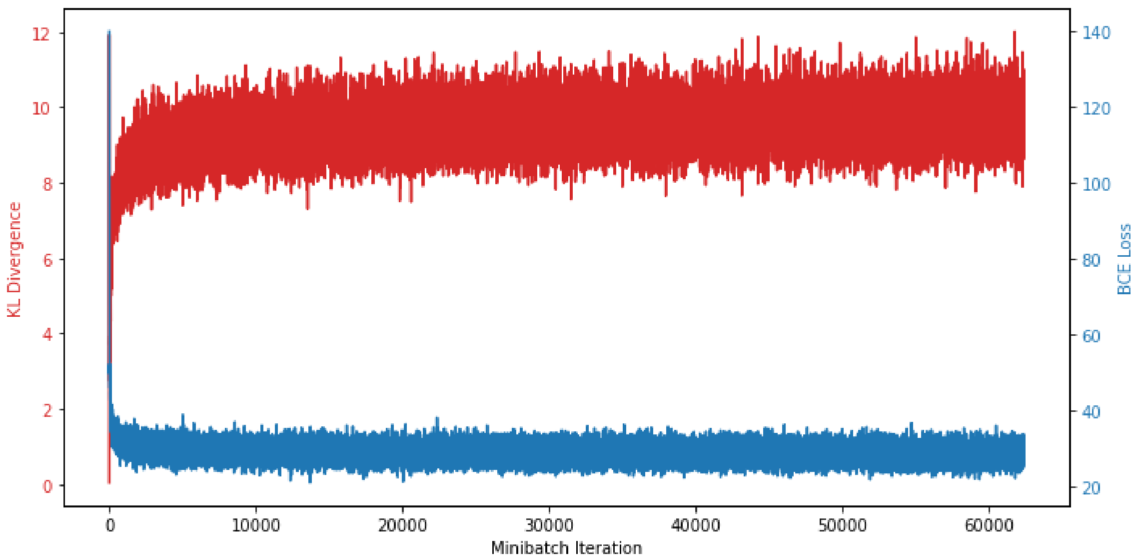
```
class Encoder(nn.Module):
    def __init__(self):
        super(Encoder, self).__init__()
        self.fc1 = nn.Linear(196, 128)
        self.fc2 = nn.Linear(128, 16)
        self.fc3 = nn.Linear(128, 16)

    def forward(self, x):
        reshaped = x.reshape(x.shape[0], -1)
        out = torch.tanh(self.fc1(reshaped))
        fc2_out = self.fc2(out)
        fc3_out = self.fc3(out)

        mu = (fc2_out[:, :8] + fc3_out[:, :8])/2
        logvar = (fc2_out[:, 8:] + fc3_out[:, 8:])/2

        std = logvar.mul(0.5).exp_()
        eps = torch.randn_like(std)
        z = eps.mul(std).add_(mu)
```

```
        return z, mu, logvar

class Decoder(nn.Module):
    def __init__(self):
        super(Decoder, self).__init__()
        self.fc1 = nn.Linear(8, 128)
        self.fc2 = nn.Linear(128, 196)

    def forward(self, x):
        out = torch.tanh(self.fc1(x))
        out = torch.sigmoid(self.fc2(out))
        return out
```

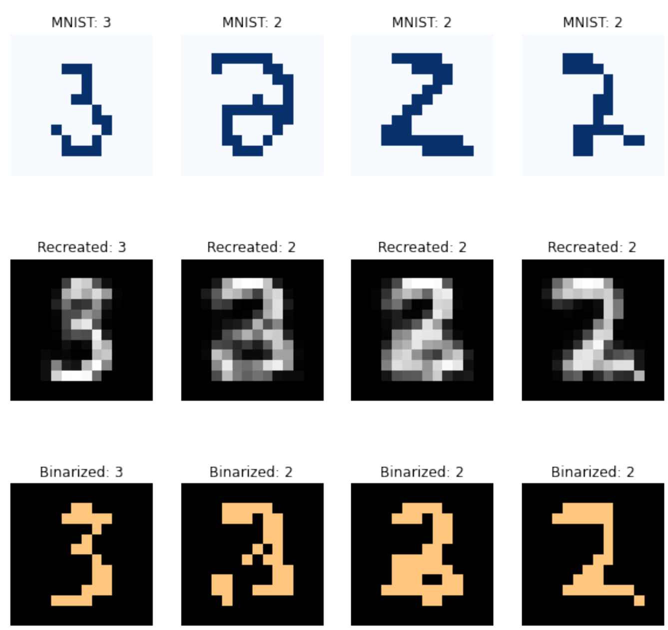
Loss functions.

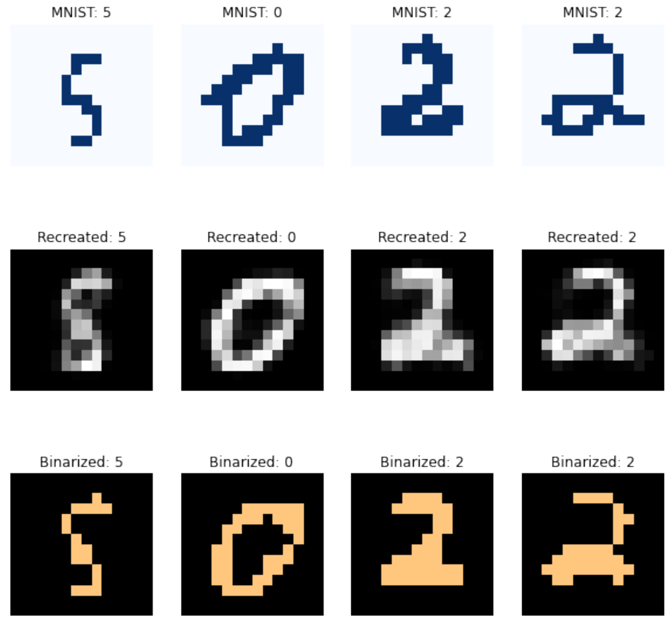
```
def KL(mu, sigma):
    contribution = 1 + sigma - mu**2 - torch.exp(sigma)
    return (torch.sum(-contribution/2))

def TotalLoss(x, mu, sigma, decoding):
    kl_loss = KL(mu, sigma)

    bce =  F.binary_cross_entropy(decoding,
                                  x.view(-1, 196),
                                  reduction='sum')
    return (kl_loss + bce, kl_loss, bce)
```

(3) Plot of first and second term of ELBO as a function of the number of weight updates.



(4) Reconstruction of MNIST images using the Autoencoder.





(5) Images created by sampling from the generative model and running the decoder.