Variational Autoencoders: A Brief Survey

Mayank Mittal*
Roll No. 14376

Harkirat Behl*
Roll No. 13286

1 Introduction

After the whooping success of deep neural networks in machine learning problems, deep generative modeling has come into limelight. Generative modeling is the task of learning the underlying complex distribution which generated a given set of data. One of the popular approach for generative modeling is Variational Autoencoder (VAE) [8] and has received a lot of attention in the past few years reigning over the success of neural networks. Variational Autoencoders are a class of deep generative models based on variational method [3]. In the work, we aim to develop a through understanding of the variational Autoencoders, look at some of the recent advances in VAEs and highlight the drawbacks of VAEs particularly in text generation.

2 Background

2.1 Problem Description

Consider a typical Bayesian learning setup for generative modeling in which there are some observable data $\mathbf{x} = \{\mathbf{x_1}, \mathbf{x_2}, ..., \mathbf{x_n}\}$ that need to be model through some latent variables $\mathbf{z} = \{\mathbf{z_1}, \mathbf{z_2}, ..., \mathbf{z_m}\}$. Mathematically speaking the objective is to to maximize the probability of each $\mathbf{x}$ in the observed data under the generative process, that is:

$$p(\mathbf{x}) = \int p(\mathbf{x}|\mathbf{z})p(\mathbf{z})\partial\mathbf{z}$$

The key idea behind VAEs is that in higher dimensions most $\mathbf{z}$ would give $p(\mathbf{x}|\mathbf{z})$ as zero. Hence, instead of trying to compute the intractable distribution $p(\mathbf{z}|\mathbf{x})$, a simpler function $q(\mathbf{z}|\phi)$ is computed that can provide a distribution over the latent variables that are more likely to produce the observations $\mathbf{x}$. This is done by minimizing the KullbackLeibler (KL) divergence between the two distributions as shown below:

$$q^*(\mathbf{z}) = \underset{q(\mathbf{z})\in\mathcal{Q}}{\operatorname{argmin}}\, \mathcal{D}_{KL}[q(\mathbf{z})||p(\mathbf{z}|\mathbf{x})] \tag{1}$$

where $\mathcal{Q}$ is family of distributions over latent variables $\mathbf{z}$. However, due to the dependency of the KL divergence on the evidence $p(\mathbf{x})$, the evidence lower bound $(ELBO)$ is optimized instead:

$$\begin{aligned} ELBO(q) &= \mathbb{E}_q[\log p(\mathbf{z})] + \mathbb{E}_q[\log p(\mathbf{x}|\mathbf{z})] - \mathbb{E}_q[\log q(\mathbf{z})] \\ &= \mathbb{E}_q[\log p(\mathbf{x}|\mathbf{z})] - \mathcal{D}_{KL}[q(\mathbf{z})||p(\mathbf{z})] \end{aligned}$$

2.2 Amortized Variational Inference

Traditional approaches in variational inferences (VI) like mean-field VI or hierarchal VI iterate over the observed data and update the variational parameters through closed-form coordinate ascent updates. These methods suffer from the disadvantages for being difficult to scale for large datasets and extending to non-conjugate models.

On the other hand, amortized VI utilizes the inferences from the past computations to support future computations. It assumes that the latent variables can be predicted by a global parametrized

function of the data. Once this function has been estimated, passing new data points through the function would provide the latent variables. Deep neural networks or *inference* networks can be used to estimate this function and helps combine the representational power of deep learning with probabilistic modeling.

VAEs is one of the popular class of models for amortized VI and was initially proposed for inference in deep latent Gaussian models (DLGM) by two separate groups in [8] and [12]. In variational autoencoders (VAEs) two sets of neural networks are used:

- **top-down generative model:** mapping from the latent variables $\mathbf{z}$ to the data $\mathbf{x}$
- **bottom-up inference model:** approximates posterior $p(\mathbf{z}|\mathbf{x})$

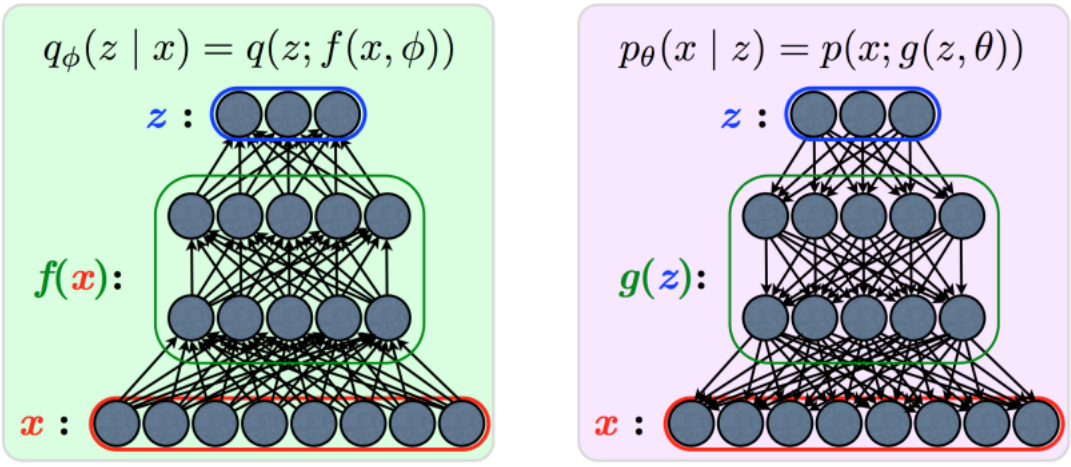

Figure 1: **Right Image:** Encoder/Recognition Network, **Left Image:** Decoder/Generative Network. *Image courtesy:* `http://videolectures.net/deeplearning2015_courville_autoencoder_extension/`

Referring to the graphical model for a variational autoencoder in Figure 2, VAEs employ an amortized variational distribution to approximate the posterior:

$$q_\phi(\mathbf{z}|\mathbf{x}) = \prod_{i=1}^{N} q_\phi(z_i|x_i) \tag{2}$$

This distribution does not depend on the local parameters and is typically chosen as $q_\phi(z_i|x_i) = \mathcal{N}(z_i|\boldsymbol{\mu}(x_i), \sigma^2(x_i)\mathbf{I})$ where where $\boldsymbol{\mu}(x_i)$ and $\sigma^2(x_i)$ are non-linear mapping of data obtained from the neural network.

During optimization, the inference and generative networks are trained together by making use of the reparameterization trick and using stochastic gradients for the model's ELBO. Given multiple datapoints $\mathbf{X}^M = \{\mathbf{x}^{(i)}\}_{i=1}^M$ drawn randomly from a dataset $\mathbf{X}$ with N datapoints, the estimated ELBO $\mathcal{L}$ is estimated using Monte-Carlo approximation:

$$\mathcal{L}(\theta, \phi, \mathbf{X}) \approx \hat{\mathcal{L}}^M(\theta, \phi, \mathbf{X}) = \frac{N}{M} \sum_{i=1}^{M} \hat{\mathcal{L}}^M(\theta, \phi, \mathbf{x}^{(i)}) \tag{3}$$

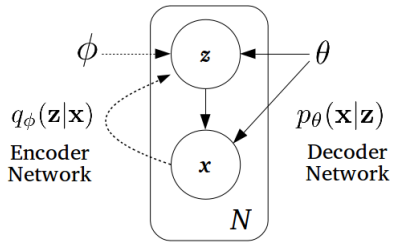

Figure 2: Graphical Model for Variational Autoencoder

3 Recent Advances in VAEs

A lot of developments [15] have been proposed over the standard VAE model that was initially introduced in [8]. In this section, we very briefly highlight the advantages of some of these advancements.

3.1 More Expressive Likelihoods

Standard VAE assumes that the likelihood factorizes over dimensions. This may result in poor approximation for tasks involving images. The following approaches have been proposed in order to overcome this:

Deep Recurrent Attentive Writer (DRAW) [4] is a neural network architecture for generating images. It uses a sequential auto-encoding framework to gradually construct the observations while automatically focusing on RoI.

PixelVAE [5], on the other hand, models the dependencies between pixels within an image as $p_\theta(x_i \mid z_i) = \prod_j p_\theta(x_i^j \mid x_i^1, x_i^2, \ldots, x_i^{j-1})$ where x_i^j denotes the j^{th} dimension of the observation i. By doing so, the dimensions are generated in a sequential fashion and accounts for the local dependencies present.

3.2 More Expressive Posteriors

Mean field approach used in standard VAEs lacks expressiveness for modeling a complex posterior. This problem is often resolved by easing out the modeling assumptions in the inference network and tightening the variational bound. A few methods which can be used to do so are:

Importance-weighted VAEs (IWAE) [1] provides increased flexibility to model complex posteriors which do not fit the VAE modeling assumptions. It requires L weighted samples from the approximate posterior with weights $w_l \propto \frac{p_\theta(\mathbf{x_i}, \mathbf{z_{(i,l)}})}{p_\phi(\mathbf{z_{(i,l)}}|\mathbf{x_i})}$. As $L \to \infty$ the implicity distribution would converge to the true posterior.

Normalizing flows [11] proposes to transform a simple approximate posterior into a expressive one by a series of successive invertible transformations. This helps in generating multi-modal distributions from a simple distribution, thus enabling modeling of complex posterior.

3.3 Implicit Distributions

VAEs rely on parametric models which further limits its ability to model complex data. Generative Adversarial Networks (GAN) is a popular way to learn implicit distributions to represent unknown densities. Larsen *et.al* [9] have been proposed to train a GAN-style discriminator between target distribution and variational distribution where latter is the implicit distribution.

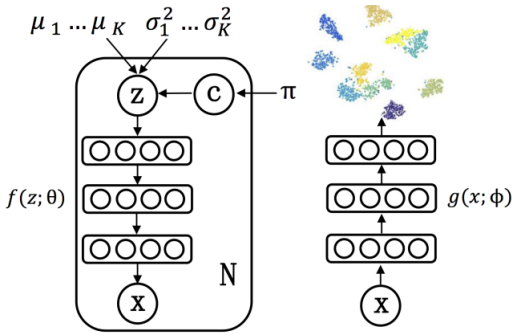

Figure 3: Diagram of VaDE. Source: [7]

4 Generative Approach to Clustering

Clustering is defined as the process of grouping objects into clusters based on similarity between them. Similar objects are put into the same cluster and dissimilar objects are put into different clusters. Clustering approaches can be broadly classified into 2 categories, which are:

1. **Similarity-based clustering:** Similarity based methods measure distance between each pair of samples. If there are N samples, a $N \times N$ similarity-matrix is created, example: Spectral Clustering (SC)
2. **Feature-based:** This method clusters based on the feature vector of the sample points, and there is not per pair evaluation as such. It takes $N \times D$ matrix as input, where N is number of samples and D is the feature dimension, examples:
   - K- Means Clustering
   - Gaussian Mixture Model (GMM)

One of the key thing that affects the performance of Feature-based methods is the quality of the feature representation of the samples. Features from deep neural networks have been proven very successful in many machine learning problems. Thus, it is seems intuitive to use deep features for clustering. Along with this, it is also important to have a clustering model which can generate new samples from a given cluster. If the model is learning the inherent distribution underlying the data, it should be able to generate new samples from this. We look at the method 'Variational Deep Embedding: An Unsupervised and Generative Approach to Clustering' by Zhuxi Jiang *et al.* [7], which uses deep generative models(VAE) with GMM to achieve this. This also provides an unsupervised learning method for deep generative models.

Variational Deep Embedding The generative story for the method VaDE by Jiang *et al.* [7] is as follows:

1. A cluster c picked up using GMM model, $c \sim cat(\pi)$
2. A latent vector is sampled from that cluster c as: $\mathbf{z} \sim \mathcal{N}(\boldsymbol{\mu}_c, \boldsymbol{\sigma^2}_c\mathbf{I})$
3. A NN-based decoder is used to map this latent vector to the data point x using :
   - $f(\mathbf{z};\theta)$ maps $\mathbf{z}$ to $(\mu_x, log\sigma_x^2)$;
   - $\mathbf{x} \sim N(\mu_x, \sigma_x^2 I)$

Similar to VAEs, a NN-based encoder $g(\mathbf{x};\phi)$, is used to model $q(\mathbf{z}|\mathbf{x})$. This along with the generative story is illustrated in Fig.4.

The inference for this model is similar to the VAE. Evidence lower bound (ELBO) for this generative process can be written as:

$$\begin{aligned}\mathcal{L}(\mathbf{x}) &= \mathbb{E}_{q(\mathbf{z},c|\mathbf{x})}\left[\log \frac{p(\mathbf{x},\mathbf{z},c)}{q(\mathbf{z},c|\mathbf{x})}\right] \\ &= \mathbb{E}_{q(\mathbf{z},c|\mathbf{x})}[\log p(\mathbf{x}|\mathbf{z}) + \log p(\mathbf{z}|c) + \log p(c) - \log q(\mathbf{z}|\mathbf{x}) - \log q(c|\mathbf{x})]\end{aligned}$$

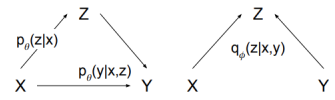

Figure 4: Diagram of CVAE. Source: [13]

The above is solved using SGVB estimator and the *re-parameterization* trick (shown in [7])

After completing the training by maximizing ELBO w.r.t parameters $\{\pi, \boldsymbol{\mu}_c, \boldsymbol{\sigma}_c, \theta, \phi\}$, $c \in \{1, .., K\}$, at the test time for each observed sample $\mathbf{x}$:

- the latent representation z is given as:

$$q(\mathbf{z}|\mathbf{x}) = \mathcal{N}(\mathbf{z}|\hat{\boldsymbol{\mu}}; \hat{\boldsymbol{\sigma}}^{\mathbf{2}}\mathbf{I})$$

  where $[\hat{\boldsymbol{\mu}}; \log \hat{\boldsymbol{\sigma}}^{\mathbf{2}}] = g(\mathbf{x}; \phi)$

- and the cluster assignment c is given as:

$$q(c|\mathbf{x}) = p(c|\mathbf{x}) = \frac{p(\mathbf{z}|c)p(c)}{\sum_{c'=1}^{K}(\mathbf{z}|c')p(c')}$$

Thus one can easily get the cluster id for a new sample, just like most clustering methods. This method can also generate new unseen samples from a given class using the procedure described in the generative story and shown in Fig.4. It is important to note that the generation is conditioned on the cluster id. Experimental results in [7] show that VaDE outperforms state-of-art methods on the clustering task. It is also shown to generate highly realistic samples.

5 Conditional VAEs and Structured Prediction

VAEs have been widely used for generating images, but VAEs have no control over the generation process. In Conditional VAE [13] the output is conditioned on an other variable. This helps to generate specific data, conditioned on some variable, for eg. generating digit images in MNIST we conditioned on the class label. Fig.1.

The generative story for the model proposed by Sohn *et al.* is as follows:

1. For a given input x, a latent vector $\mathbf{z}$ is sampled from the prior network as :$\mathbf{z} \sim p_\theta(z|x)$
2. The output is generated using a NN-based decoder: $p_\theta(y|z, x)$

It is very important to note that unlike the traditional VAE where the output is generated just using the latent vector z, here both the latent vector z and input x are used to generate the output. This is illustrated in Fig.5. Thus, the input observations modulate the prior on Gaussian latent variables which generates the output. A NN-based encoder is used for recognition: $q_\phi(z|y, x)$, as shown in Fig.5.

This model can also be used for structured output prediction [13]. It also allows for multiple modes in conditional distribution of y given x: $p_\theta(y|x)$, by virtue of the latent variable z, i.e one-to-many mapping. Unlike the normal Deep neural network where the output y is a one-to-one mapping of the input x.

The inference is done in the in Stoachstic Gradient Variational Bayes (SGVB) framework [8] in a fashion similar to VAE. The empirical lower bound can be written as:

$$\mathcal{L}_{CVAE}(\mathbf{x}, \mathbf{y}; \theta, \phi) = -\mathcal{D}_{KL}[q_\phi(\mathbf{z}|\mathbf{x}, \mathbf{y})||p_\theta(z|x)] + \frac{1}{L}\sum_{l=1}^{L}[\log p_\theta(\mathbf{y}|\mathbf{x}, \mathbf{z}^{(\mathbf{l})})]$$

where $z^{(l)} = g(x, y, \epsilon^{(l)})$, $\epsilon^{(l)} \sim \mathcal{N}(0, I)$ and L is the number of examples. This can be solved using SGVB estimator and the *re-parameterization*.

5.1 Experiments

5.1.1 CVAE

We implemented the conditional VAE method [13]. Kristiadi's blog [1] was used as a reference for this implementation. The CVAE was trained on MNIST dataset, which is composed of images of digits from 0 to 9. Fig.1 shows the results of generation of digits conditioned on their class labels.

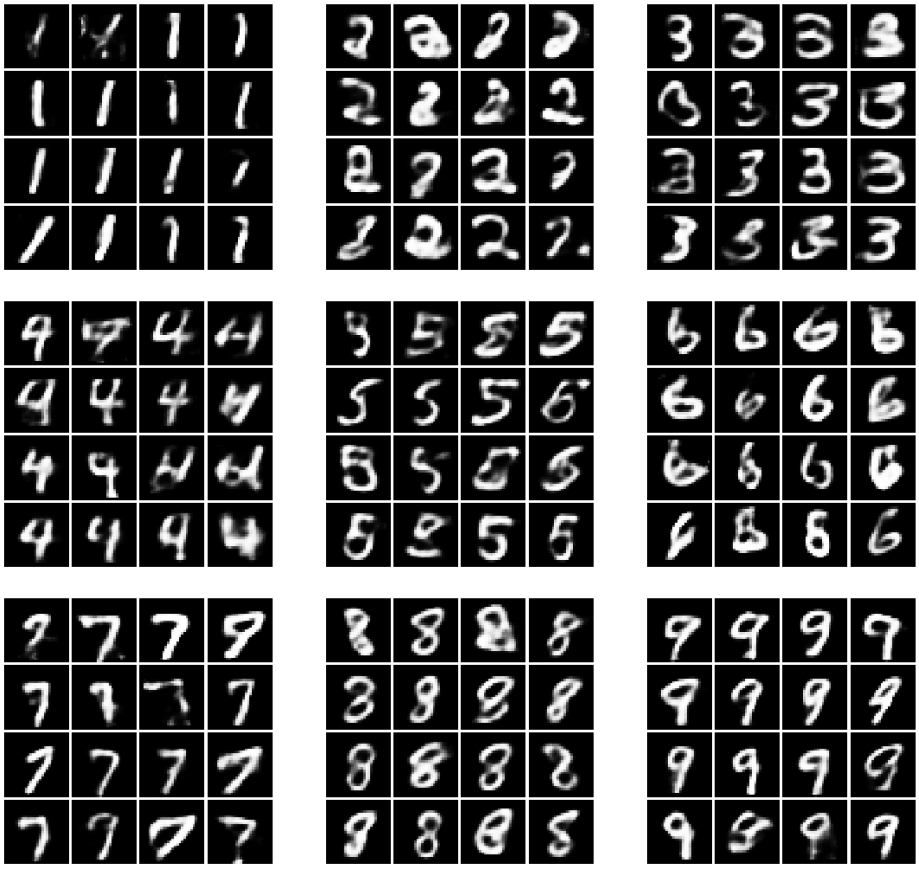

Table 1: Results generated by CVAE implementation in PyTorch on MNIST, conditioned on class labels. It can be seen that images for 1,7,8 and 7 are generated very well because they are unique. However, while generating there is some confusion between 4 and 9; 7 and 9; 2 and 3; 4 and 5; since these pairs look similar to each other.

5.1.2 VAE

We also implemented the VAE model proposed by Kingma *et al.*, to compare the results with CVAE. Both the models were trained on the same dataset MNIST, in PyTorch, and the same NN-encoder and NN-decoder was used. This was done to understand the difference the conditioning on digit labels makes in the generation results. Fig.2 shows the quality of images generated by our VAE implementation.

[1] `https://wiseodd.github.io/techblog/2016/12/17/conditional-vae/`

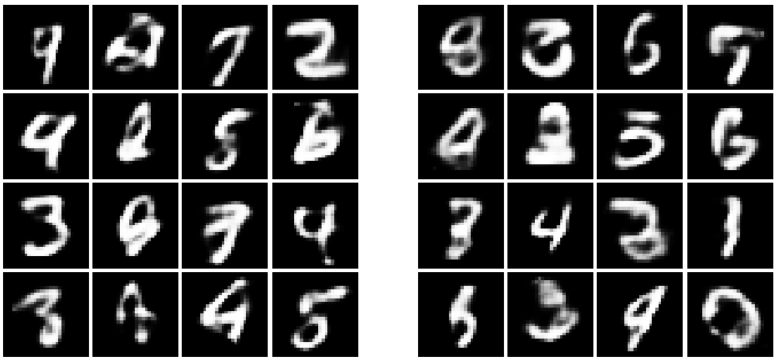

Table 2: Digits generated by the original VAE model. It is important to note that the image generation is not conditioned on anything in this case.

6 Challenges with Variational Autoencoders

6.1 Sub-Optimality in Inference

Cremer *et al.* in [2] highlighted inference sub-optimality in Amortized VI due to an *approximation gap* (because of the particular variational family chosen) and *amortization gap* (created because of the gap between the log-likelihood and the ELBO due to amortization).

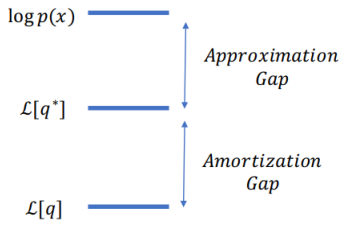

Figure 5: Gaps in Inference. Here $\mathcal{L}[q]$ is the ELBO evaluated using amortized distribution q, and $\mathcal{L}[q^*]$: ELBO evaluated using optimal approximation within its variational family

In their work, to reduce the amortization gap, they proposed an inference network over input to initialize the variational parameters, and then refine them through stochastic VI (SVI). Unlike in Amortized VI, they compute the derivate of the ELBO through back-propagation using SVI updates which can be done efficiently using fast Hessian-vector products [10].

6.2 Text-related Applications

One of the main challenges with text data is that the words are discrete in nature, unlike image data which is real-valued. Although VAEs have been shown to generate realistic images, text generation is still an open problem. To overcome this, one of the approaches proposed is to keep a mapping from the discrete word space to a continuous word space.

Ways to encode semantic information into the latent variable is being explored currently. One other common problem faced in text generation using VAEs is that the model can sometimes generate the next word in a sentence using the grammar rules only without taking the latent vector into consideration. Therefore it is necessary to enforce the model to use information from the latent vector, for which different solutions have been proposed.

7 Ongoing Work

At the moment we are working on Unsupervised structured prediction problem. Structured prediction problem is a very important problem in Computer Vision. More specifically, we are working on segmentation on the Caltech-UCSD Birds (CUB) [14] dataset.

8 Conclusion

VAEs are a very powerful generative model based on autoencoders and Bayesian neural networks setup. They have been shown to be successful in various domains ranging from supervised learning problems including conditional image generation to unsupervised learning problems like clustering. Conditional generation is much easier with VAEs. VAEs for text generation still has a lot of challenges because of the discrete nature of text data, and is a hot area of research.

References

[1] Yuri Burda, Roger Grosse, and Ruslan Salakhutdinov. Importance weighted autoencoders, 2015.

[2] Chris Cremer, Xuechen Li, and David Duvenaud. Inference suboptimality in variational autoencoders, 2018.

[3] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[4] Karol Gregor, Ivo Danihelka, Alex Graves, Danilo Jimenez Rezende, and Daan Wierstra. Draw: A recurrent neural network for image generation, 2015.

[5] Ishaan Gulrajani, Kundan Kumar, Faruk Ahmed, Adrien Ali Taiga, Francesco Visin, David Vazquez, and Aaron Courville. Pixelvae: A latent variable model for natural images, 2016.

[6] Matt Hoffman, David M. Blei, Chong Wang, and John Paisley. Stochastic variational inference, 2012.

[7] Zhuxi Jiang, Yin Zheng, Huachun Tan, Bangsheng Tang, and Hanning Zhou. Variational deep embedding: An unsupervised and generative approach to clustering, 2016.

[8] Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2013.

[9] Anders Boesen Lindbo Larsen, Søren Kaae Sønderby, and Ole Winther. Autoencoding beyond pixels using a learned similarity metric. *CoRR*, abs/1512.09300, 2015.

[10] Barak A. Pearlmutter. Fast exact multiplication by the hessian. *Neural Comput.*, 6(1):147–160, January 1994.

[11] Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows, 2015.

[12] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models, 2014.

[13] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 3483–3491. Curran Associates, Inc., 2015.

[14] P. Welinder, S. Branson, T. Mita, C. Wah, F. Schroff, S. Belongie, and P. Perona. Caltech-UCSD Birds 200. Technical Report CNS-TR-2010-001, California Institute of Technology, 2010.

[15] Cheng Zhang, Judith Butepage, Hedvig Kjellstrom, and Stephan Mandt. Advances in variational inference, 2017.

A Stochastic Variational Inference

Applies stochastic optimization with mini-batches to obtain stochastic estimate of $ELBO$ [6]

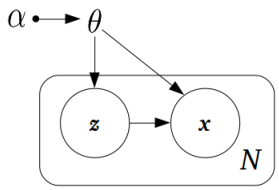

Figure 6: Graphical Model for Stochastic VI

B VAE Objective Function

In variational inference, computing KL is difficult as it depends on evidence $p(\mathbf{x})$

$$\begin{aligned}\mathcal{D}_{KL}[q(\mathbf{z})||p(\mathbf{z}|\mathbf{x})] &= \mathbb{E}_q[\log q(\mathbf{z})] - \mathbb{E}_q[\log p(\mathbf{z}|\mathbf{x})] \\ &= \mathbb{E}_q[\log q(\mathbf{z})] - \mathbb{E}_q[\log p(\mathbf{z},\mathbf{x})] \\ &\quad + \mathbb{E}_q[\log p(\mathbf{x})]\end{aligned}$$

VAEs introduce an inference machine $q_\phi(\mathbf{z}|\mathbf{x})$ that learns to approximate the posterior $p_\theta(\mathbf{z}|\mathbf{x})$. Suppose that $\mathbf{z}$ is sampled from some arbitrary distribution with pdf $q_\phi(\mathbf{z})$. The relationship between $\mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z})}[p_\theta(\mathbf{x}|\mathbf{z})]$ and $p(\mathbf{x})$ can be written as:

$$\begin{aligned}&\mathcal{D}_{KL}[q_\phi(\mathbf{z})||p_\theta(\mathbf{z}|\mathbf{x})] = \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z})}[\log p(\mathbf{z}) - \log p_\theta(\mathbf{z}|\mathbf{x})] \\ \implies &\mathcal{D}_{KL}[q_\phi(\mathbf{z})||p_\theta(\mathbf{z}|\mathbf{x})] = \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z})}[\log p(\mathbf{z}) - \log p_\theta(\mathbf{x}|\mathbf{z}) - \log p_\theta(\mathbf{z})] + \log p(\mathbf{x}) \\ \implies &\log p(\mathbf{x}) - \mathcal{D}_{KL}[q_\phi(\mathbf{z})||p_\theta(\mathbf{z}|\mathbf{x})] = \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z})}[\log p_\theta(\mathbf{x}|\mathbf{z})] + \mathcal{D}_{KL}[q_\phi(\mathbf{z})||p_\theta(\mathbf{z})]\end{aligned}$$

In above equation the LHS has the quantity we want to maximize: $\log p_\theta(x)$, while RHS is something we can optimize via stochastic gradient descent given the right choice of q.

As explained in Section 2, we have solved our problem of sampling $\mathbf{z}$ by training a distribution q to predict which values of $\mathbf{z}$ are likely to produce $\mathbf{x}$ and ignoring the rest. Variational lower bound $\mathcal{L}$ on the data likelihood s.t. $p_\theta(\mathbf{x}) \geq \mathcal{L}(\theta, \phi, \mathbf{x})$, where

$$\begin{aligned}\mathcal{L}(\theta,\phi,\mathbf{x}) &= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x},\mathbf{z}) - \log q_\phi(\mathbf{z}|\mathbf{x}) \\ &= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z}) + \log p_\theta(z) - \log q_\phi(\mathbf{z}|\mathbf{x}) \\ &= -\mathcal{D}_{KL}[q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(z)] + \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})]\end{aligned}$$

NOTE: $-\mathcal{D}_{KL}[q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(z)]$ is a *regularization* term, and $\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})]$ is a *reconstruction* term