KU LEUVEN

Variational Auto-Encoder

Data Mining and Neural
Networks [H05R4a] 2020 - 2021

KU Leuven, Belgium

December 15, 2020

Variational Auto-Encoder

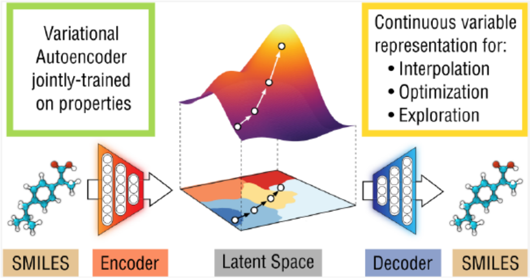

Figure: Durg discovery with VAE[1]

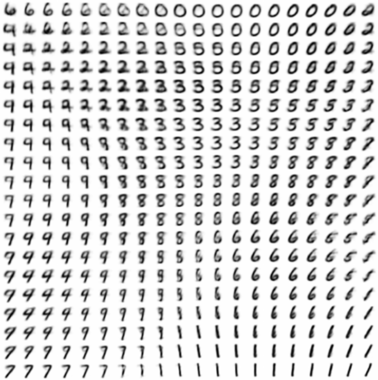

Figure: Visualization of learned 2-D latent space of MNIST digits

[1] Rafael Gómez-Bombarelli et al. "Design of efficient molecular organic light-emitting diodes by a high-throughput virtual screening and experimental approach". In: *Nature Mater* (Oct. 2016).

Variational Auto-Encoder

- A type of probabilistic latent variable model, that takes the interpretation of regularized Auto encoders.
- Many statistical models make use of latent variables to describe a probability distribution over observables. Usually, the latent variables have a simple distribution, often a separable distribution. Thus when learning a latent variable model, we are finding a description of the data in terms of *independent components*.
- A variational autoencoder is a model that estimates the 'variational lower bound' on the 'marginal likelihood' estimate of datapoints.

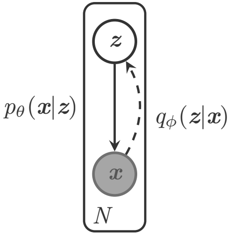

Figure: Directed Graphical model under consideration. Solid line denotes the generative model $p_\theta(\boldsymbol{x}|\boldsymbol{z})p_\theta(\boldsymbol{z})$ and dashed line denotes the variational approximation $q_\phi(\boldsymbol{z}|\boldsymbol{x})$ to the posterior $p_\theta(\boldsymbol{z}|\boldsymbol{x})$.

Diederik P. Kingma and Max Welling. "Auto-Encoding Variational Bayes". In: *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*. 2014
Diederik P. Kingma and Max Welling. "An Introduction to Variational Autoencoders". In: *Foundations and Trends® in Machine Learning* (2019)

VAE: Variational bound

Consider a probabilistic model with observations $\boldsymbol{X} = \{\boldsymbol{x}_i\}_{i=1}^N$ consisting of i.i.d. samples from unknown density function $p_\theta(\boldsymbol{x})$, latent variables $\boldsymbol{z}$ with prior $p_\theta(\boldsymbol{z})$ and a likelihood function $p_\theta(\boldsymbol{x}|\boldsymbol{z})$.

We also introduce a variational approximation $q_\phi(\boldsymbol{z}|\boldsymbol{x})$ to the intractable posterior $p_\theta(\boldsymbol{z}|\boldsymbol{x})$. Why?

Because from *Bayes rule*, we've $p_\theta(\boldsymbol{z}|\boldsymbol{x}) = p_\theta(\boldsymbol{x}|\boldsymbol{z})p_\theta(\boldsymbol{z}) / \left(\int p_\theta(\boldsymbol{x}|\boldsymbol{z})p_\theta(\boldsymbol{z})\, d\boldsymbol{z}\right)$. The integral is difficult to evaluate since it needs to be computed over arbitrarily large configurations of $\boldsymbol{z}$.

We aim to maximize the $(\log)$ probability of each $\boldsymbol{x}_i$ according to:

$$
\begin{aligned}
\log p_\theta(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N) = \sum_{i=1}^N \log p_\theta(\boldsymbol{x}_i) &= \sum_{i=1}^N \log \int p_\theta(\boldsymbol{x}_i|\boldsymbol{z})p_\theta(\boldsymbol{z})\, dz \\
&= \sum_{i=1}^N \log \int \frac{q_\phi(\boldsymbol{z}|\boldsymbol{x}_i)}{q_\phi(\boldsymbol{z}|\boldsymbol{x}_i)} p_\theta(\boldsymbol{x}_i|\boldsymbol{z})p_\theta(\boldsymbol{z})\, d\boldsymbol{z} \\
&= \sum_{i=1}^N \log \left(\mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x}_i)}\left[\frac{p_\theta(\boldsymbol{z})}{q_\phi(\boldsymbol{z}|\boldsymbol{x}_i)} p_\theta(\boldsymbol{x}_i|\boldsymbol{z})\right]\right)
\end{aligned}
$$

$$
\begin{aligned}
&\geq \sum_{i=1}^{N} \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x}_i)}\left[\log\left(\frac{p_\theta(\boldsymbol{z})}{q_\phi(\boldsymbol{z}|\boldsymbol{x}_i)} p_\theta(\boldsymbol{x}_i|\boldsymbol{z})\right)\right] \\
&= \sum_{i=1}^{N} \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x}_i)}\left[\log\left(\frac{p_\theta(\boldsymbol{z})}{q_\phi(\boldsymbol{z}|\boldsymbol{x}_i)}\right) + \log\left(p_\theta(\boldsymbol{x}_i|\boldsymbol{z})\right)\right] \\
&= \sum_{i=1}^{N} -\overbrace{\mathbb{D}_{\mathrm{KL}}[\underbrace{q_\phi(\boldsymbol{z}|\boldsymbol{x}_i)}_{\text{encoder}} \,||\, \underbrace{p_\theta(\boldsymbol{z})}_{\text{fixed for VAE}}]}^{\text{regularizer}} + \overbrace{\mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x}_i)}[\log \underbrace{p_\theta(\boldsymbol{x}_i|\boldsymbol{z})}_{\text{decoder}}]}^{\text{reconstruction error}},
\end{aligned} \tag{1}
$$

where we have used Jensen's inequality $(f(\mathbb{E}[\boldsymbol{x}]) \geq \mathbb{E}[f(\boldsymbol{x})])$ when $f(\cdot)$ is concave. This bound is often referred to as the Evidence Lower Bound (ELBO).

It consists of two terms:

- *Kulback-Leibler divergence* between the approximate posterior and the prior distribution (which acts as a regularizer for smoothness in latent space),
- *Expected reconstruction error*.

This bound provides a unified objective function for optimization of both the parameters θ and ϕ of the model and variational approximation, respectively.

Sampling in latent space

To generate new data, we need to sample from posterior $\boldsymbol{z}_i \sim q_\phi(\boldsymbol{z}|\boldsymbol{x}_i)$.

However random sampling cannot be used here, because back-propagation through such operation is not possible.

Hence we employ the *'re-parametrization trick'*. It is often possible to express the random variable $\boldsymbol{z}$ as the deterministic variable $\boldsymbol{z} = g_\phi(\boldsymbol{\epsilon}, \boldsymbol{x})$, where ϵ is an auxiliary variable with independent marginal $p(\boldsymbol{\epsilon})$, and $g_\phi(\cdot)$ is some vector-valued function parameterized by ϕ.

For eg. in univariate Gaussian case: let $z \sim q(z|x) = \mathcal{N}(\mu, \sigma^2)$. Here a valid re-parametrization trick is $z = \mu + \sigma\epsilon$, where ϵ is auxiliary noise variable $\epsilon \sim \mathcal{N}(0, 1)$. This trick moves the random sampling operation to an auxiliary variable $\boldsymbol{\epsilon}$, which is then shifted by the mean and scaled by the standard deviation.

Example: Encoder and decoder outputs as Gaussian distributions

Let the variational posterior to be $q_\phi(\boldsymbol{z}|\boldsymbol{x}_{(i)}) = \mathcal{N}(\boldsymbol{z}; \boldsymbol{\mu}_\phi^{(i)}, \boldsymbol{\sigma}_\phi^{2(i)}\mathbb{I})$, where mean $\boldsymbol{\mu}_\phi^{(i)}$ and standard deviation $\boldsymbol{\sigma}^{(i)}$ are the outputs of the encoding neural net with parameters ϕ and the prior $p(\boldsymbol{z}) = \mathcal{N}(\boldsymbol{z}; 0, \mathbb{I})$. Then employing the parametrization trick we have $\boldsymbol{z}_i = g_\phi(\boldsymbol{x}_i, \boldsymbol{\epsilon}) = \boldsymbol{\mu}_\phi^{(i)} + \boldsymbol{\sigma}_\phi^{(i)} \odot \boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbb{I})$ and $\odot$ is the element-wise product.

If we let the decoder network be $p_\theta(\boldsymbol{x}|\boldsymbol{z}) = \mathcal{N}(\boldsymbol{x}; \psi_\theta(\boldsymbol{z}), \boldsymbol{\sigma}^2\mathbb{I})$. Since the KL-divergence between two multivariate Gaussians could be written in a closed form [2], the maximization problem in Eq. (1) is equivalent to the minimization of

$$\boxed{\min_{\theta,\phi} \frac{1}{N}\sum_{i=1}^{N}\{\underbrace{\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathbb{I})}\|\boldsymbol{x}_i - \psi_\theta(\boldsymbol{\mu}_\phi^{(i)} + \boldsymbol{\sigma}_\phi^{(i)}\boldsymbol{\epsilon})\|_2^2}_{\text{reconstruction error}} + \frac{1}{2}\underbrace{(\mathrm{Tr}(\boldsymbol{\Sigma}_\phi^{(i)}) + \boldsymbol{\mu}_\phi^{(i)\top}\boldsymbol{\mu}_\phi^{(i)} - l - \log\det(\boldsymbol{\Sigma}_\phi^{(i)}))}_{\text{regularizer}}\}} \quad (2)$$

where $\boldsymbol{\Sigma}_\phi^{(i)} = \boldsymbol{\sigma}_\phi^{(i)}\boldsymbol{\sigma}_\phi^{(i)\top}$ is the covariance matrix.

In general, $p_\theta(\boldsymbol{x}|\boldsymbol{z})$ could be any distribution. When restricted to be Bernoulli (for eg. in MNIST dataset), the reconstruction error takes the form of binary cross-entropy loss.

[2] http://mi.eng.cam.ac.uk/ mjfg/local/4F10/lect4.pdf

Schematic illustration

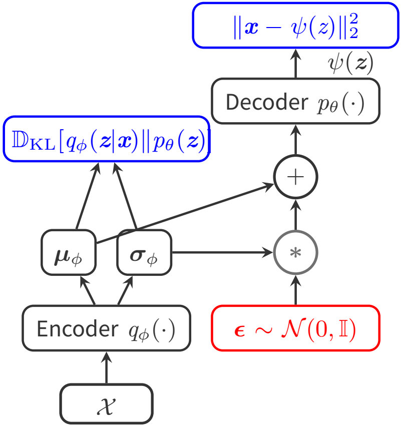

Figure: A training-time variational autoencoder shown as feed-forward neural network, where $p_\theta(\boldsymbol{x}|\boldsymbol{z})$ is Gaussian. Red shows sampling operration that is non-differentiable and blue shows the loss layer.