TTIC 31230, Fundamentals of Deep Learning

David McAllester, Autumn 2024

Variational Auto-Encoders (VAEs)

Widely Used VAEs

Diffusion Models: A diffusion model is a hierarchical Gaussian VAE.

Vector Quantized VAEs: A VQ-VAE defines $P_{\text{enc}}(z|y)$ in terms of vector quantization analogous to K-means clustering. VQ-VAEs provide a translation from continuous data, such as images, to token data that can be modelled with a transformer. This is done in GPT-4o.

Auto-Regressive Language Models: An auto-regressive language model, such as a transformer, is mathematically equivalent to a hierarchical VAE.

VAEs

A variational autoencoder (VAE) is defined by three parts:

- An encoder distribution $P_{\text{enc}}(z|y)$.
- A decoder distribution $P_{\text{dec}}(y|z)$
- A "prior" distribution $P_{\text{pri}}(z)$

VAE generation uses $P_{\text{pri}}(z)$ and $P_{\text{dec}}(y|z)$.

VAE training uses the encoder $P_{\text{enc}}(z|y)$.

Two Joint Distributions

A VAE defines two joint distributions on y and z, namely $P_{\text{gen}}(y, z)$ and $P_{\text{enc}}(y, z)$ defined by

$$P_{\text{gen}}(y, z) = P_{\text{pri}}(z)P_{\text{dec}}(y|z)$$

$$P_{\text{enc}}(y, z) = \text{Pop}(y)P_{\text{enc}}(z|y)$$

Training the Generator

Fix the encoder arbitrarily and train P_{gen} by cross entropy.

$$\text{gen}^* = \underset{\text{gen}}{\text{argmin}} \; E_{(y,z)\sim P_{\text{enc}}(y,z)} \left[-\ln P_{\text{gen}}(y,z) \right]$$

Under universality we have $P_{\text{gen}^*} = P_{\text{enc}}$ and hence the generator distribution on y defined by gen^* matches the population distribution.

In a diffusion model the encoder just adds noise. The encoder is not trained.

The ELBO Loss

$$\mathrm{Pop}(y) \;=\; \frac{\mathrm{Pop}(y)P_{\mathrm{enc}}(z|y)}{P_{\mathrm{enc}}(z|y)} \;=\; \frac{P_{\mathrm{enc}}(y,z)}{P_{\mathrm{enc}}(z|y)}$$

$$E_{(y,z)\sim P_{\mathrm{enc}}}\left[-\ln \mathrm{Pop}(y)\right] \;=\; E_{(y,z)\sim P_{\mathrm{enc}}}\left[-\ln \frac{P_{\mathrm{enc}}(y,z)}{P_{\mathrm{enc}}(z|y)}\right]$$

$$H(y) \;\leq\; E_{(y,z)\sim P_{\mathrm{enc}}}\left[-\ln \frac{P_{\mathrm{gen}}(y,z)}{P_{\mathrm{enc}}(z|y)}\right]$$

The right hand side of the last line is called the **ELBO Loss**.

The Variational Bayes Interpretation

The generator is interpreted as a Bayesian model where y is evidence for z.

$$\begin{aligned}
\ln P_{\text{gen}}(y) &= \ln \frac{P_{\text{gen}}(y) P_{\text{gen}}(z|y)}{P_{\text{gen}}(z|y)} \\
&= E_{z \sim P_{\text{enc}}(z|y)} \left[\ln \frac{P_{\text{gen}}(y,z)}{P_{\text{gen}}(z|y)} \right] \\
&\geq E_{z \sim P_{\text{enc}}(z|y)} \left[\ln \frac{P_{\text{gen}(y,z)}}{P_{\text{enc}}(z|y)} \right]
\end{aligned}$$

Hence the name **evidence lower bound** or **ELBO**.

Fundamental Equations of Deep Learning

- Cross Entropy Loss: $\Phi^* =$

$$\underset{\Phi}{\operatorname{argmin}}\, E_{(x,y)\sim \mathrm{Pop}} \left[-\ln P_\Phi(y|x) \right]$$

- GAN: $\mathrm{gen}^* =$

$$\underset{\mathrm{gen}}{\operatorname{argmax}} \min_{\mathrm{disc}} E_{i\sim\{-1,1\},y\sim P_i} \left[-\ln P_{\mathrm{disc}}(i|y) \right]$$

- VAE: $\mathrm{pri}^*, \mathrm{dec}^*, \mathrm{enc}^* =$

$$\underset{\mathrm{pri,dec,enc}}{\operatorname{argmin}} \; E_{(y,z)\sim P_{\mathrm{enc}}} \left[-\ln \frac{P_{\mathrm{gen}}(y,z)}{P_{\mathrm{enc}}(z|y)} \right]$$

Training the Encoder

In diffusion models the encoder simply adds noise and is not trained.

In a VQ-VAE the encoder is trained. A naive approach to training the encoder is to optimize the ELBO loss.

$$\text{enc}^* = \underset{\text{enc}}{\text{argmin}}\, E_{(y,z)\sim P_{\text{enc}}}\left[-\ln \frac{P_{\text{gen}}(y,z)}{P_{\text{enc}}(z|y)}\right]$$

Training the Encoder

$$\text{enc}^* = \underset{\text{enc}}{\text{argmin}}\, E_{(y,z)\sim P_{\text{enc}}} \left[-\ln \frac{P_{\text{gen}}(y,z)}{P_{\text{enc}}(z|y)} \right]$$

Unfortunately this optimization involves optimizing a sampling distribution (the encoder). As with GANs, optimizing a sampling distribution (such as a GAN generator) is subject to mode collapse — the loss function is very forgiving of a failure of the sampling distribution to cover the desired space of values.

Gaussian VAEs

Gaussian VAEs, which are the original version from 2014, train the encoder.

Gaussian VAEs solve the problem of optimizing a sampling distribution by sampling instead from fixed Gaussian noise. This further allows expressing the ELBO loss as a "closed form" L_2 loss which avoids the need to even sample the noise.

However, non-hierarchical Gaussian VAEs (with a single Gaussian latent variable) produce poor results in practice. A diffusion model is a hierarchical Gaussian VAEs which does not train the encoder. Hierarchical Gaussian VAEs which train the encoder can also produce good results.

A Non-Hierarchical Gaussian VAE

$$P_{\text{pri}}(z) = \mathcal{N}(0, I)$$

$$P_{\text{enc}}(z|y) = \mathcal{N}(\hat{z}(y), I)$$

$$P_{\text{dec}}(y|z) = \mathcal{N}(\hat{y}(z), I)$$

In general we can use arbitrary Gaussians but this example makes the math simple.

Gaussian VAEs

$$E_{(y,z)\sim P_{\text{enc}}}\left[-\ln \frac{P_{\text{pri}}(z)P_{\text{dec}}(y|z)}{P_{\text{enc}}(z|y)}\right]$$

$$= E_{y\sim \text{Pop}}\left[KL(P_{\text{enc}}(z|y), P_{\text{pri}}(z)) + E_{z\sim P_{\text{enc}}(z|y)}\left[-\ln P_{\text{dec}}(y|z)\right]\right]$$

$$= E_{y\sim \text{Pop}}\left[\frac{1}{2}||\hat{z}_{\text{enc}}(y)||^2 + E_{\epsilon}\left[\frac{1}{2}||y - \hat{y}_{\text{dec}}(\hat{z}_{\text{enc}}(y) + \epsilon))||^2\right]\right]$$

Training the Encoder

In a VQ-VAE the encoder is traned jointly with the decoder $P_{\text{dec}}(y|z)$ but is trained independently of $P_{\text{pri}}(z)$. $P_{\text{pri}}(z)$ is trained later using a transformer model. The encoder of a VQ-VAE is closely related to K-means clustering. In a VQ-VAE the encoder converts vectors to tokens so that a transformer can be applied.

This minimal training of the encoder again exploits the fact that under universality $\text{Pop}(y)$ can be modelled fully for any encoder.

A different approach to training the encoder, an ME-VAE, is discussed below.

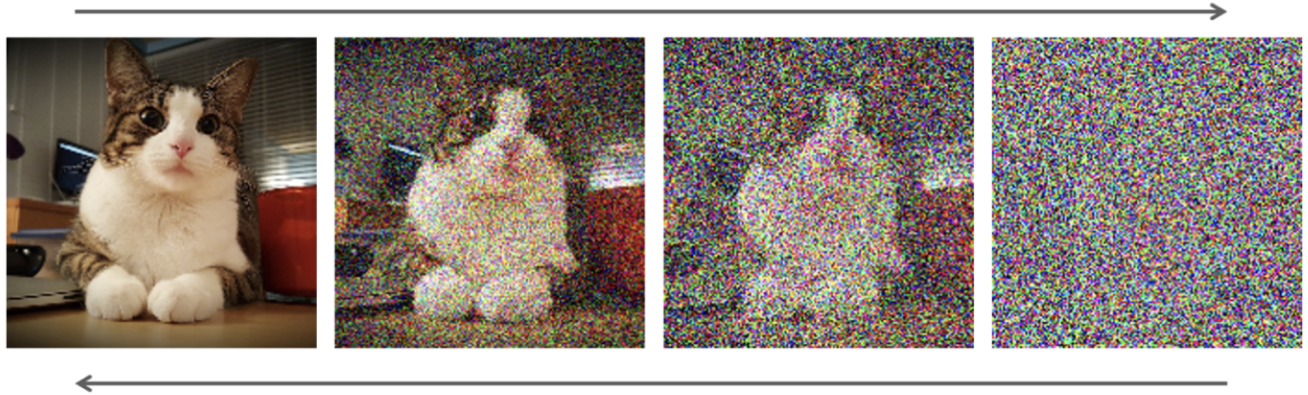
Hierarchical VAEs

Hierarchical Gaussian VAEs which train the encoder are explored both theoretically and empirically by Vahdat and Kautz.

NVAE: A Deep Hierarchical Variational Autoencoder, Arash Vahdat, Jan Kautz (NVIDIA, January 2021)

But diffusion models and autoregressive models are also instances of hierarchical VAEs.

Hierachical VAEs



[Sally talked to John] $\overrightarrow{\leftarrow}$ [Sally talked to] $\overrightarrow{\leftarrow}$ [Sally talked] $\overrightarrow{\leftarrow}$ [Sally] $\overrightarrow{\leftarrow}$ []

$$y \overrightarrow{\leftarrow} z_1 \overrightarrow{\leftarrow} \cdots \overrightarrow{\leftarrow} z_N$$

Hierarchical VAEs

$$y \overset{\rightarrow}{\leftarrow} z_1 \overset{\rightarrow}{\leftarrow} \cdots \overset{\rightarrow}{\leftarrow} z_N$$

Encoder: $\mathrm{Pop}(y)$, $P_{\mathrm{enc}}(z_1|y)$,$P_{\mathrm{enc}}(z_2|z_1), \ldots, P(z_N|z_{N-1})$.

Generator: $P_{\mathrm{pri}}(z_N), P_{\mathrm{dec}}(z_{N-1}|z_N), \ldots, P_{\mathrm{dec}}(z_1|z_2), P_{\mathrm{dec}}(y|z_1)$

The encoder and the decoder define distributions $P_{\mathrm{enc}}(y, z_1, \ldots, z_N)$ and $P_{\mathrm{gen}}(y, z_1, \ldots, z_N)$ respectively.

Hierarchical ELBO Loss

$$H(y) = E_{(y,z_1,\ldots,z_n)\sim P_{\text{enc}}}\left[-\ln \frac{P_{\text{gen}}(y, z_1, \ldots, z_n)}{P_{\text{enc}}(z_1, \ldots, z_N|y)}\right]$$

EM-VAEs

The use of minimal encoder training may reflect the mode collapse problem of training a sampling distribution, such as a GAN generator or a VAE encoder.

The situation might be different if a better method were available for training the encoder. Here I will propose a method for training the encoder that avoids the mode collapse problem.

EM-VAEs

We start with the following "optimum encoder" inequality.

$$E_{y\sim \mathrm{Pop}, z\sim P_{\mathrm{gen}}(z|y)}\left[-\ln \frac{P_{\mathrm{gen}}(y,z)}{P_{\mathrm{gen}}(z|y)}\right] \leq E_{(y,z)\sim P_{\mathrm{enc}}}\left[-\ln \frac{P_{\mathrm{gen}}(y,z)}{P_{\mathrm{enc}}(z|y)}\right]$$

This implies $P^*_{\mathrm{enc}}(z|y) = P_{\mathrm{gen}}(z|y)$ and universality gives

$$\mathrm{enc}^* = \operatorname*{argmin}_{\mathrm{enc}} E_{(y,z)\sim P_{\mathrm{gen}}} - \ln P_{\mathrm{enc}}(z|y)$$

EM-VAE

$$\text{E: enc}^* = \operatorname*{argmin}_{\text{enc}} E_{(y,z)\sim P_{\text{gen}}} - \ln P_{\text{enc}}(z|y)$$

$$\text{M: gen}^* = \operatorname*{argmin}_{\text{gen}} E_{(y,z)\sim P_{\text{enc}}(y,z)} \left[-\ln P_{\text{gen}}(y,z)\right]$$

The classical EM algorithm is the case where we alternate optimizing the encoder (the E step) and the generator (the M step) and where the E step yields $P_{\text{enc}}(z|y) = P_{\text{gen}}(z|y)$ exactly and where the M step cannot fully fit the population.

Here we can use SGD on these two objectives independent of details of the models.

Derivation of the Encoder Optimum

$$
\begin{aligned}
& E_{(y,z)\sim P_{\text{enc}}}\left[-\ln \frac{P_{\text{gen}}(y,z)}{P_{\text{enc}}(z|y)}\right] \\
= \; & E_{(y,z)\sim P_{\text{enc}}}\left[-\ln \frac{P_{\text{gen}}(y,z)}{P_{\text{gen}}(z|y)}\right] + E_{y\sim \text{Pop}}\, KL(P_{\text{enc}}(z|y), P_{\text{gen}}(z|y)) \\
\geq \; & E_{(y,z)\sim P_{\text{enc}}}\left[-\ln \frac{P_{\text{gen}}(y,z)}{P_{\text{gen}}(z|y)}\right] \\
= \; & E_{(y,z)\sim P_{\text{enc}}}\left[-\ln P_{\text{gen}}(y)\right] \\
= \; & E_{y\sim \text{Pop}, z\sim P_{\text{gen}}(z|y)}\left[-\ln P_{\text{gen}}(y)\right] \\
= \; & E_{y\sim \text{Pop}, z\sim P_{\text{gen}}(z|y)}\left[-\ln \frac{P_{\text{gen}}(y,z)}{P_{\text{gen}}(z|y)}\right]
\end{aligned}
$$

END