TTIC 31230, Fundamentals of Deep Learning

David McAllester, Autumn 2022

Diffusion Models

Progressive VAEs

Diffusion models are a special case of progrssive VAEs.

A progressive VAE for an observed valriable y (such as an image) has layers of latent variables $z_1, \ldots, z_L$.

The encoder defines $P_{\text{enc}}(z_1|y)$ and $P_{\text{enc}}(z_{\ell+1}|z_\ell)$ (defining a Markov chain).

In a diffusion model the encoder is held fixed (not trained) and defined in a way that guarantees that the mutual ingformation $I(z_{\ell+1}, y) < I(z_\ell, y)$ and $I(z_L, y) = 0$ with $P_{\text{enc}}(z_L|y)$ being noise independent of y.

Progressive VAE Loss Function

The model has a prior $P_{\text{pri}}(x_L)$. In a diffusion model this prior analytically determined by the encoders and isnot trained. decoder will $P_{\text{dec}}(z_{\ell-1}|z_\ell)$ and $P_{\text{dec}}(y|z_1)$.

Following VQ-VAE, we will train the encoder and the decoder independent of any prior.

We then train a prior on the top layer latent variable. The top level prior and decoder allow us to sample y from the model.

Phase One Training

We train a encoders and decoders $\text{enc}_1, \text{dec}_1, \ldots, \text{enc}_L, \text{dec}_L$ where the distribution on $z_1, \ldots, Z_L$ is defined by y and the encoder.

$$\text{enc}_1^*, \text{dec}_1^* = \underset{\text{enc}_1, \text{dec}_1}{\text{argmin}} \; E_{y, z_1} \left[-\ln P_{\text{dec}_1}(y | z_1) \right]$$

$$\text{enc}_{\ell+1}^*, \text{dec}_{\ell+1}^* = \underset{\text{enc}_{\ell+1}, \text{dec}_{\ell+1}}{\text{argmin}} \; E_{z_\ell, z_{\ell+1}} \left[-\ln P_{\text{dec}_{\ell+1}}(z_{\ell-1} | z_\ell) \right]$$

If these encoders and decoders share parameters the shared parameters are influenced by all of the above training losses (this observation was added after seeing DALLE-2's diffision model).

Phase Two Training

$$\text{pri}^* = \underset{\text{pri}}{\text{argmin}} \; E_{z_L}\left[-\ln P_{\text{pri}}(z_L)\right]$$

Because of the autonomy of the encoder, the universality assumption implies that we get a perfect model of the population distribution on y.

Given the prior and the decoder we can sample images.

Modeling Densities on R^d

Consider a model density $p_\Phi(y)$ on $y \in R^d$ (for example sound waves or images).

Ideally we want to be able to compute $p_\Phi(y)$, the denisty for any given y, and to also sample y from $p_\Phi(y)$.

Continuous Softmax

We consider the case where a density is defined by a continuous softmax.

$$
\begin{aligned}
p_{\text{score}}(y) &= \underset{y}{\text{softmax}}\ \text{score}(y) \\
&= \frac{1}{Z} e^{\text{score}(y)} \\
Z &= \int dy\ e^{\text{score}(y)}
\end{aligned}
$$

Here score(y) is a parameterized model computing a score and defining a probability density on R^d.

Langevin Dynamics — MCMC for Continuous Densities

If y is discrete, but from an exponentially large space (such as sentences or a semantic image segmentation) we can use MCMC sampling (the Metropolis algorithm or Gibbs sampling).

In the continuous case the analogue of MCMC sampling is Langevin dynamics.

Langevin Dynamics

$$y(t + \Delta t) = y(t) + 2g\Delta t + \epsilon\sqrt{\Delta t}$$

$$g = \nabla_y \, \text{score}(y)$$

$$\epsilon \sim \mathcal{N}(0, I)$$

This give a well-defined distribution on functions of time in the limit as $\Delta t \to 0$.

$$dy = 2gdt + \epsilon\sqrt{dt} \qquad \epsilon \sim \mathcal{N}(0, I)$$

The Stationary Density

The gradient flow is equal to $2p(x)\ \nabla_y \text{ score}(y)$.

The diffusion flow is $-2\nabla_y\ p(y)$ (see the slides on SGD).

$$\nabla_y\ p(y) \ = \ p(y)\ \nabla_y \text{ score}(y)$$

$$\frac{dp}{p} \ = \ d \text{ score}$$

$$\ln p \ = \ \text{score} + C$$

$$p(y) \ = \ \frac{1}{Z}\ e^{\text{score}(y)} \quad = \quad \underset{y}{\text{softmax}}\ \text{score}(y)$$

The Stationary Density

So in a limit where $\Delta t \to 0$ but $t \to \infty$ we have $p_t(y) = \text{softmax}_y\ s(y)$.

In theory this gives a sampling algorithm for $p(y) = \text{softmax}_y\ \text{score}(y)$.

This is called "score matching" — Z remains unknown and we do not get any way of computing Z or $p_{\text{score}}(y)$.

END