TTIC 31230, Fundamentals of Deep Learning

David McAllester, Autumn 2022

Diffusion Models

Progressive VAEs

We consider a VAE with layers of latent variables $z_1, \ldots, z_L$ and a population distribution on an observable variable y.

The encoder will defines $P_{\mathrm{enc}}(z_1|y)$ and $P_{\mathrm{enc}}(z_{\ell+1}|z_\ell)$.

In diffusion models the encoder is fixed (not trained) where we have that $H_{\mathrm{enc}}(z_{\ell+1}|z_\ell)$ is small (each encoder step is nearly deterministic).

We have that the mutual information

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

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 $\mathrm{enc}_1, \mathrm{dec}_1$, ..., $\mathrm{enc}_L, \mathrm{dec}_L$ where the distribution on $z_1, \ldots, Z_L$ is defined by y and the encoder.

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

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

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

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

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.

$$
\begin{aligned}
p_{\mathrm{score}}(y) &= \operatorname*{softmax}_{y}\ \mathrm{score}(y) \\
&= \frac{1}{Z} e^{\mathrm{score}(y)} \\
Z &= \int dy\ e^{\mathrm{score}(y)}
\end{aligned}
$$

Here $\mathrm{score}(y)$ 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

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

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

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

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 $2p(x)\ \nabla_y\ \mathrm{score}(y)$.

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

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

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

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

$$p(y) = \frac{1}{Z}\, e^{\mathrm{score}(y)} = \operatorname*{softmax}_y\ \mathrm{score}(y)$$

The Stationary Density

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

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

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

END