TTIC 31230, Fundamentals of Deep Learning

David McAllester, Autumn 2023

Adjusting Generation

Temperature, Guidance, and Majority Voting

Temperature-Adjusted Generation

$$\text{Training:} \quad \Phi^* = \operatorname*{argmin}_{\Phi} E_{(x,y)\sim \text{Pop}}[-\ln P_\Phi(y|x)]$$

$$P_\Phi(y|x) = \operatorname*{softmax}_{y} s_\Phi(y|x)$$

$$\text{Desired Generation:} \quad P_\Phi^\beta(y|x) = \operatorname*{softmax}_{y} \beta s_\Phi(y|x) \propto P_\Phi(y)^\beta$$

Temperature Adjusted Generation for Language

In practice we use

$$P_\Phi^\beta(y_{i+1} \mid y_1, \ldots, y_i) = \operatorname*{softmax}_{y_{i+1}} \beta s_\Phi(y_{i+1} \mid y_1, \ldots, y_i)$$

$$\propto P_\Phi(y_{i+1} \mid y_1, \ldots, y_i)^\beta$$

This is different from

$$P_\Phi^\beta(y_1, \ldots, y_N) \propto P_\Phi(y_1, \ldots, y_N)^\beta$$

Temperature-Adjusted Generation for Language

In language translation we take $\beta = \infty$ (softmax $\Rightarrow$ argmax).

For language generation $\beta = 1$ tends to yield rambling and incoherent text.

On the other hand $\beta = \infty$ generates repetition.

We look for a Goldilocks β.

An alternative to temperature-adjusted generation is top-P sampling, also called nucleus sampling, which is similar in structure and performance.

There is a literature on generation adjustment for language.

Temperature-Adjusted Reverse-Diffusion

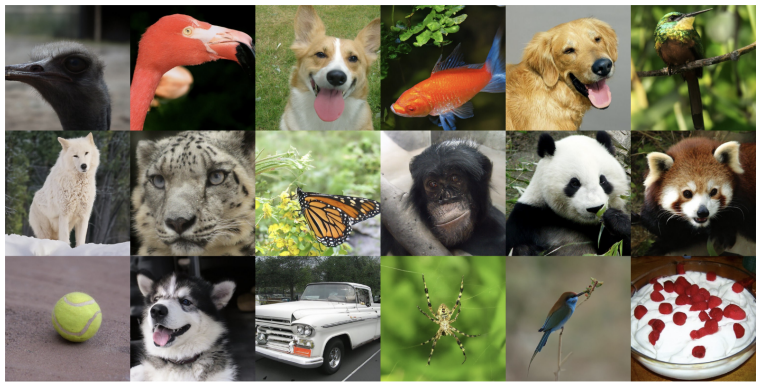
$$z(t - \Delta t) = z(t) + \left(\frac{\hat{E}_\Phi[y|t, z(t)] - z(t)}{t} \right) \Delta t + \frac{1}{\sqrt{\beta}} \epsilon \sqrt{\Delta t}$$

As with language generation, this is not the same as $P_\Phi^\beta(y) \propto P_\Phi(y)^\beta$

Classifier Guidance

Diffusion Models Beat GANs on Image Synthesis
Dhariwal and Nichol, May 2021

For **class-conditional** generation of imagenet images $P(y|x)$ they train an **unconditional** diffusion image model $P_\Phi(y)$ and utilize a pretrained imagenet classification model $P_\Psi(x|y)$.



Classifier Guidance

They note that

$$P(y|x) = \frac{P(y)P(x|y)}{P(x)} \propto P(y)P(x|y)$$

For generation they modify the reverse-diffusion process so as to intuitively approximate

$$P^{\beta,\gamma}_{\Phi,\Psi}(y|x) = \operatorname*{softmax}_{y} \beta(s_\Phi(y)+\gamma s_\Psi(x|y)) \;\propto\; P_\Phi(y)^\beta P_\Psi(x|y)^{\beta+\gamma}$$

Classifier Guidance

$$z(t-\Delta t) = z(t) + \left(\frac{\hat{E}_\Phi[y|t, z(t)] - z(t)}{t} + \gamma \nabla_z \, s_\Psi(x|z(t)) \right) \Delta t + \frac{1}{\sqrt{\beta}} \, \epsilon \sqrt{\Delta t}$$

This is different from, but motivated by,

$$P^{\beta,\gamma}_{\Phi,\Psi}(y|x) \propto P_\Phi(y)^\beta P_\Psi(x|y)^{\beta+\gamma}$$

Conditional Diffusion Models

$$P_\Phi(y \mid \text{panda bear chemist})$$

panda mad scientist mixing sparkling chemicals, artstation

$$\text{Train } \hat{E}_\Phi[y|t, z(t), x]$$

Classifier Free Guidance (Self-Guidance)

Classifier Free Diffusion Guidance
Ho and Salimans, December 2021 (NeurIPS workshop)

$$\text{Training:} \quad \Phi^* = \operatorname*{argmin}_{\Phi} E_{(x,y)\sim \text{Pop}}[-\ln P_\Phi(y|x)]$$

$$P_\Phi(y|x) = \operatorname*{softmax}_{y} s_\Phi(y|x)$$

We introduce a special x-value $\emptyset$ and arrange that

$$\text{Pop}(y|\emptyset) = \text{Pop}(y).$$

Guidance

For $\beta > 0$ they modify the reverse-diffusion process to intuitively approximate

$$P_\Phi^\beta(y|x) = \underset{y}{\text{softmax}}\, \beta s_\Phi(y|x) - (\beta-1) s_\Phi(y|\emptyset) \quad \propto \frac{P_\Phi(y|x)^\beta}{P_\Phi(y|\emptyset)^{\beta-1}}$$

For $\beta = 1$ we have no adjustment.

$$P_\Phi^1(y|x) = P_\Phi(y|x)$$

For $\beta >> 1$ (used in practice) we have.

$$P_\Phi^\beta(y|x) \approx \underset{y}{\text{softmax}}\, \beta (s_\Phi(y|x) - s_\Phi(y|\emptyset)) \quad \propto \quad \left(\frac{P_\Phi(y|x)}{P_\Phi(y|\emptyset)}\right)^\beta$$

Guidance

$$P_\Phi^\beta(y|x) = \underset{y}{\text{softmax}}\,\beta(s_\Phi(y|x) - s_\Phi(y|\emptyset)) \quad \propto \quad \left(\frac{P_\Phi(y|x)}{P_\Phi(y|\emptyset)}\right)^\beta$$

$$z(t-\Delta t) = z(t) + \left(\frac{(\hat{E}_\Phi[y|t, z(t), x] - \hat{E}_\Phi[y|t, z(t), \emptyset]) - z_t}{t}\right)\Delta t + \frac{1}{\sqrt{\beta}}\,\epsilon\sqrt{\Delta t}$$

Guidance

$$P_\Phi^\beta(y|x) \quad \propto \quad \left(\frac{P_\Phi(y|x)}{P_\Phi(y|\emptyset)}\right)^\beta$$

Ho and Salimans motivate this from Classifier Guidance and

$$P(x|y) \ \propto \ \frac{P(y|x)}{P(y)}$$

But this is false.

$$P(x|y) = \frac{P(x)P(y|x)}{P(y)} \not\propto \frac{P(y|x)}{P(y)}$$

Guidance

$$z(t-\Delta t) = z(t) + \left(\frac{(\hat{E}_\Phi[y|t, z(t), x] - \hat{E}_\Phi[y|t, z(t), \text{blurry}]) - z_t}{t} \right) \Delta t + \frac{1}{\sqrt{\beta}} \epsilon \sqrt{\Delta t}$$

This will make the generated image sharper.

A More General Formulation: Byte VAEs

Consider a Markovian VAE with deterministic encoder $z_1(y)$ and $z_{i+1}(z_i)$ and where $z_N(z_{N-1})$ is a constant (an empty string or the zero vector).

Deterministic encoding sensible for discrete VAES. This includes autoregressive language models and "byte VAES"

A byte VAE is a numerical Markovian VAE where each z_i is a vector of (discrete) bytes and where the dimensionality of z_i decreases as i increases.

To computing $z_{i+1}(z_i)$ we can first compute a floating point vector then round each coordinate to the nearest byte.

We can model $-\ln P_{\text{dec}}(z_{i-1}|z_i)$ as $||z_{i-1} - \hat{z}_{i-1}(z_i)||^2$.

A More General Formulation

$$\text{enc}^*, \text{gen}^* = \underset{\text{enc,gen}}{\text{argmin}} \; E_y[-\ln(P_{\text{gen}}(y|z_1)P_{\text{gen}}(z_1|z_2)\cdots P_{\text{gen}}(z_{N-1}|\emptyset))]$$

In a language model we generate one word at a time.

But we can also consider the case where z_i is a vector whose dimension is (slightly) decreasing as i increases.

Instead of adding one word at a time we add a small number of numbers (dimensions) at a time.

In this case we can use

$$P_{\text{gen}}(z_{i-1}|z_i) = \hat{z}_{i-1}(z_i) + \epsilon, \quad \epsilon \sim \mathcal{N}(0, I)$$

A More General Formulation

$$\text{enc}^*, \text{gen}^* = \underset{\text{enc,gen}}{\text{argmin}} \; E_y[-\ln(P_{\text{gen}}(y|z_1)P_{\text{gen}}(z_1|z_2)\cdots P_{\text{gen}}(z_{N-1}|\emptyset))]$$

$$P_{\text{gen}}(z_{i-1}|z_i) = \hat{z}_{i-1}(z_i) + \epsilon, \quad \epsilon \sim \mathcal{N}(0, I)$$

$$\text{enc}^*, \text{gen}^* = \underset{\text{enc,gen}}{\text{argmin}} \; E_y \, ||y - z_1||^2 + \sum_{i=1}^{N-1} ||z_i - \hat{z}_i(z_{i+1})||^2$$

Conditional Generation

Training the encoder and the decoder conditioned on x (as in a language translation model). This trains $\hat{z}_i{}_{-1}(z_i, x)$.

For generation we then have

$$\text{Unadjusted: } z_{i-1} = \hat{z}_{i-1}(z_i, x) + \epsilon$$

$$\text{Temperature Adjusted: } z_{i-1} = \hat{z}_{i-1}(z_i, x) + \frac{1}{\sqrt{\beta}}\,\epsilon$$

$$\text{Guidance Adjusted: } z_{i-1} = \hat{z}_{i-1}(z_i, x_{\text{good}}) - \hat{z}_{i-1}(z_i, x_{\text{bad}}) + \frac{1}{\sqrt{\beta}}\,\epsilon$$

Output z_1

Adjusted Generation: Majority Voting

Self-Consistency Improves Chain Of Thought Reasoning In Language Models
Wang et al., March 2023

The answer is taken to be a majority vote over stochastic chain of thought generation.

$$\text{answer}^*(\text{question}) = \underset{\text{answer}}{\text{argmax}} \; E_{\text{thought} \sim P_{\text{LLM}}(\text{thought}|\text{question})} \; P_{\text{LLM}}(\text{answer}|\text{question}; \text{thought})$$

END