Article 1
Improved Denoising Diffusion Probabilistic Models

Synopsis:

Motivation and Problem Formulation

While Ho et al. (2020) found that DDPMs can generate high fidelity samples according to FID and Inception Score , they were unable to achieve competitive log-likelihoods with these models. This paper introduced several measures to do so.

Proposed Approach

Learnable Variance

In Ho et al. (2020), the authors set $\sum_\theta(x_t, t) = \sigma_t^2 I$, where σ_t is not learned. Ho et al. (2020), they found that fixing σ_t^2 to β_t yielded roughly the same sample quality as fixing it to β_t'. They found empirically that β_t and $\beta_t' = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}}$ are almost equal for all diffusion time steps except at the beginning. On the other hand they found empirically that the first few steps of the diffusion process contribute the most to the variational lower bound. Hence learning variance parameter is useful for lowering loss.

They parameterized variance term as an interpolation between $\log \beta_t$ & $\log \beta_t'$ since these terms are nearly equal and the reasonable range for is quite small. So the interpolated variance will become:

$$\sum_\theta(x_t, t) = \exp v \log \beta_t + (1 - v) \log \beta_t'$$

Since $L_{simple} = E_{t,x_0,\epsilon}[||\epsilon - \epsilon_\theta(x_t, t)||^2]$ does not depend on $\sum_\theta(x_t, t)$ the defined hybrid objective $L_{hybrid} = L_{simple} + \lambda L_{vlb}$ and applied a stop-gradient to the $\mu_\theta(x_t, t)$ output for the L_vlb term. This ensures that L_simple is still source of influence over $\mu_\theta(x_t, t)$ while L_vlb guides $\sum_\theta(x_t, t)$.

Improving the Noise Schedule

Linear noise schedule did not work well for medium to low resolution images.The authors found empirically that model trained with the linear schedule does not get much worse (as measured by FID) even if we skip up to 20% of the reverse diffusion process.

To address this problem, they construct a different noise schedule as follows:

$$\bar{\alpha}_t = \frac{f(t)}{f(0)}, \quad f(t) = \cos \frac{\frac{t}{T} + s}{1 + s} \cdot \frac{\pi}{2}^2$$

This cosine schedule is designed to have a linear drop-off of $\bar{\alpha}_t$ whereas in the linear schedule $\bar{\alpha}_t$ falls much faster to 0 destroying information much faster.

Reducing Gradient Noise

They found empirically that the gradients for L_vlb were much more noisier than L_hyrbid. They hypothesized that sampling t uniformly causes unnecessary noise in the L_vlb objective.

They proposed an importance sampling scheme where the weights for each of the loss timesteps in the VLB objective was proportional to the squared the loss for that timestep.

$$L_{vlb} = E_{t\sim p_t}, \quad where \quad p_t \propto \sqrt{E[L_t^2]} \quad and \quad \sum p_t = 1$$

Since $E[L_t^2]$ is unknown beforehand they maintained a history of the previous 10 values for each loss term, and update this dynamically during training.

Improving Sampling Speed

During sampling they used an arbitrary subsequence $S\ of\ t$ values. Given the training noise schedule $\bar{\alpha}^t$, for a given sequence S we can obtain the sampling noise schedule $\bar{\alpha}^t$, which can be then used to obtain corresponding sampling variances

$$\beta_{S_t} = 1 - \frac{\bar{\alpha}_{S_t}}{\bar{\alpha}_{S_{t-1}}}, \quad \bar{\beta}_{S_t} = \frac{1 - \bar{\alpha}_{S_{t-1}}}{1 - \bar{\alpha}_{S_t}} \beta_{S_t}$$

So they computed $p(x_{t-1}/x_{S_t})$ as $\mathcal{N}(\mu_\theta(x_{S_t}, S_t), \sum_\theta(x_{S_t}, S_t))$.

Conclusion

With the above changes, DDPMs can sample much faster and achieve better log-likelihoods with little impact on sample quality. The likelihood is improved by learning $\sum_\theta$ using our parameterization and L_{hybrid} objective.

Article 2
Score Based Generative Modeling Through Stochastic Differential Equations

Synopsis:

Introduction

The authors present a stochastic differential equation (SDE) that smoothly transforms a complex data distribution to a known prior distribution by slowly injecting noise, and a corresponding reverse-time SDE that transforms the prior distribution back into the data distribution by slowly removing the noise.

Crucially, the reverse-time SDE depends only on the time-dependent gradient field (a.k.a., score) of the perturbed data distribution. These scores can be accurately estimated with neural networks, and use numerical SDE solvers to generate samples

Approach

Perturbing data with SDEs

Perturbing data with multiple noise scales is key to the success of score based generative models. The authors propose to generalize this idea further to an infinite number of noise scales, such that perturbed data distributions evolve according to an SDE as the noise intensifies

The forward diffusion process can be modeled as a solution to an into SDE-

$$dx = f(x,t)dt + g(t)dw$$

Where **w** is the standard Weiner process (a.k.a. Brownian motion.), $f(\cdot,t) : \mathbb{R}^d -> \mathbb{R}^d$ is a vector values function called the drift coefficient of $x(t)$, and $g(\cdot) : \mathbb{R} -> \mathbb{R}$ is a scalar function known as diffusion coefficient of $x(t)$. In above formulation p_0 is the data distribution and p_T is prior distribution and we have a continuous time diffusion process indexed by time variable $t \in [0,T]$.

Generating samples by reversing the SDE

The reverse of a diffusion process is also a diffusion process, running backwards in time and given by the reverse-time SDE-

$$dx = [f(x,t) - g(t)^2 \nabla_x log(p_t(x))]dt + g(t)d\bar{w}$$

We observe that this reverse SDE depends only on the score of the distribution.

Estimating scores for the SDE

To estimate score , we can train a time-dependent score-based model $s_\theta(x,t)$, via a continuous generalization to Noise Conditional Score Networks with denoising score matching-

$$\theta^* = \underset{\theta}{\operatorname{argmin}}\, \mathbb{E}_t \lambda(t) \mathbb{E}_{x(0)} \mathbb{E}_{x(t)|x(0)} [||s_\theta(x(t),t) - \nabla_{x(t)} log(x(t)|x(0))||_2^2]$$

Here $\lambda : [0,T] \longrightarrow R_{>0}$ is a positive weighting function, t is the uniformly sampled over $[0,T]$. $x(0) \sim p_0(x)$ and $x(t) \sim p_{0t}(x(t)|x(0))$. $p_{p0}(x(t)|x(0))$ is the transition kernel.

Solving the reverse SDE

- After training a time-dependent score-based model s_θ, we can use it to construct the reverse-time SDE and then simulate it with numerical approaches to generate samples from p_0.
- We can use general-purpose numerical methods exist for solving SDEs such as Euler-Maruyama.

Predictor corrector samples

We can improve upon the traditional SDE solvers.At each time step, the numerical SDE solver first gives an estimate of the sample at the next time step, playing the role of a "predictor". Then, the score-based MCMC approach(Langevin dynamics) corrects the marginal distribution of the estimated sample, playing the role of a "corrector"

Probability flow and connection neural ODEs

Score-based models enable another numerical method for solving the reverse-time SDE. For all diffusion processes, there exists a corresponding deterministic process whose trajectories share the same marginal probability densities $p_t(x)_{t=0}^{T}$ as the SDE. This deterministic process satisfies an ODE which is given by

$$dx = [f(x,t) - \frac{1}{2}g(t)^2 \nabla_x log(p_t(x))dt]$$

Efficient sampling

Sampling from deterministic neural ODE's is considerably faster than sampling from reverse SDE with a very small. As with neural ODEs, we can sample $x(0) \sim p_0$ by solving above from different final conditions $x(T) \sim p_T$.

Exact likelihood computation

We can compute the density defined by above equation via the instantaneous change of variables formula (Chen et al., 2018). This allows us to compute the exact likelihood on any input data.