Classifier Guidance

Mateo Guaman Castro

November 2025

Classifier Guidance

Overview: Train a classifier $p_\phi(y|x_t, t)$ on noisy images x_t, then use gradients $\nabla_{x_t} \log p_\phi(y|x_t, t)$ to guide the denoising process towards class label y.

For stochastic denoising:

Given class label y, gradient scale s

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

for all t from T to 1:

$$x_{t-1} \sim \mathcal{N}(\mu_\theta(x_t, t) + s \cdot \Sigma_\theta(x_t, t) \cdot \nabla_{x_t} \log p_\phi(y|x_t), \Sigma_\theta(x_t, t))$$

return x_0

For deterministic denoising (DDIM):

Given class label y, gradient scale s

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

for all t from T to 1:

$$\hat{\epsilon} = \epsilon_\theta(x_t, t) - \sqrt{1 - \bar{\alpha}_t} \nabla_{x_t} \log p_\phi(y|x_t)$$
$$x_{t-1} = \sqrt{\bar{\alpha}_{t-1}} \left(\frac{x_t - \sqrt{1 - \bar{\alpha}_t}\hat{\epsilon}}{\sqrt{\bar{\alpha}_t}} \right) + \sqrt{1 - \bar{\alpha}_{t-1}}\hat{\epsilon}$$

return x_0

How are these derived?

Derivations (Appendix D)

1. Define a conditional Markovian noising process $\hat{q}$ similar to $q(x_t|x_{t+1})$. assume that $\hat{q}(y|x_0)$ is a known ground truth label distribution.

$$\hat{q}(x_0) := q(x_0)$$
$$\hat{q}(y|x_0) := \text{Known labels per sample.}$$
$$\hat{q}(x_{t+1}|x_t, y) := q(x_{t+1}|x_t)$$

→ Noising process doesn't depend on class.

$$\hat{q}(x_{1:T}|x_0, y) := \prod_{t=1}^{T} \hat{q}(x_t|x_{t-1}, y)$$

→ Will be needed to use w/ Bayes rule later

Now, will prove that $\hat{q}(x_{t+1}|x_t) = q(x_{t+1}|x_t)$, i.e. with this new conditional model, the noising process is the same as the original.

$$
\begin{aligned}
\hat{q}(x_{t+1}|x_t) &= \int_y \hat{q}(x_{t+1}, y|x_t)\, dy && \text{Marginalization} \\
&= \int_y \hat{q}(x_{t+1}|x_t, y)\hat{q}(y|x_t)\, dy && \text{Product rule} \\
&= \int_y q(x_{t+1}|x_t)\hat{q}(y|x_t)\, dy && \text{By def.} \\
&= q(x_{t+1}|x_t) \int_y \hat{q}(y|x_t)\, dy \\
&= q(x_{t+1}|x_t) \\
&= \hat{q}(x_{t+1}|x_t, y) && \text{By def.}
\end{aligned}
$$

Now we can do something similar for the joint distribution

$$
\begin{aligned}
\hat{q}(x_{1:T}|x_0) &= \int_y \hat{q}(x_{1:T}, y|x_0)\, dy && \text{Marginalization} \\
&= \int_y \hat{q}(y|x_0)\hat{q}(x_{1:T}|x_0, y)\, dy && \text{Product rule} \\
&= \int_y \hat{q}(y|x_0) \prod_{t=1}^{T} \hat{q}(x_t|x_{t-1}, y)\, dy && \text{By definition} \\
&= \int_y \hat{q}(y|x_0) \prod_{t=1}^{T} q(x_t|x_{t-1})\, dy && \text{By definition} \\
&= \prod_{t=1}^{T} q(x_t|x_{t-1}) \int_y \hat{q}(y|x_0)\, dy && \text{Pull out} \\
&= \prod_{t=1}^{T} q(x_t|x_{t-1}) && \text{Sum to 1} \\
&= q(x_{1:T}|x_0) && \text{Chain rule}
\end{aligned}
$$

Now, derive $\hat{q}(x_t)$:

$$
\begin{aligned}
\hat{q}(x_t) &= \int_{x_{0:t-1}} \hat{q}(x_0, \ldots, x_t)\, dx_{0:t-1} \\
&= \int_{x_{0:t-1}} \hat{q}(x_0)\hat{q}(x_1, \ldots, x_t|x_0)\, dx_{0:t-1} \\
&= \int_{x_{0:t-1}} q(x_0)q(x_{1:t}|x_0)\, dx_{0:t-1} \\
&= \int_{x_{0:t-1}} q(x_0, \ldots, x_t)\, dx_{0:t-1} \\
&= q(x_t)
\end{aligned}
$$

Now we have identities:

$$
\begin{aligned}
\hat{q}(x_t) &= q(x_t) \\
\hat{q}(x_{t+1}|x_t) &= q(x_{t+1}|x_t)
\end{aligned}
$$

We can show w/ Bayes rule that $\hat{q}(x_t|x_{t+1}) = q(x_t|x_{t+1})$

$$\begin{aligned}\hat{q}(x_t|x_{t+1}) &= \frac{\hat{q}(x_{t+1}|x_t) \cdot \hat{q}(x_t)}{\hat{q}(x_{t+1})} \\ &= \frac{q(x_{t+1}|x_t)q(x_t)}{q(x_{t+1})} \\ &= q(x_t|x_{t+1})\end{aligned}$$

Now, $\hat{q}$ gives rise to a noisy classification function $\hat{q}(y|x_t)$

First, we show that this distribution does not depend on x_{t+1} (a noisier version of x_t) which will be useful later.

$$\begin{aligned}\hat{q}(y|x_t, x_{t+1}) &= \frac{\hat{q}(x_{t+1}|x_t, y) \cdot \hat{q}(y|x_t)}{\hat{q}(x_{t+1}|x_t)} \\ &= \frac{\hat{q}(x_{t+1}|x_t)\hat{q}(y|x_t)}{\hat{q}(x_{t+1}|x_t)} \\ &= \hat{q}(y|x_t)\end{aligned}$$

We can now derive the conditional denoising process:

$$\begin{aligned}\hat{q}(x_t|x_{t+1}, y) &= \frac{\hat{q}(x_t, x_{t+1}, y)}{\hat{q}(x_{t+1}, y)} \\ &= \frac{\hat{q}(x_t, x_{t+1}, y)}{\hat{q}(y|x_{t+1})\hat{q}(x_{t+1})} \\ &= \frac{\hat{q}(x_t|x_{t+1})\hat{q}(y|x_t, x_{t+1})\hat{q}(x_{t+1})}{\hat{q}(y|x_{t+1})\hat{q}(x_{t+1})} \\ &= \frac{\hat{q}(x_t|x_{t+1})\hat{q}(y|x_t, x_{t+1})}{\hat{q}(y|x_{t+1})} \\ &= \frac{\hat{q}(x_t|x_{t+1})\hat{q}(y|x_t)}{\hat{q}(y|x_{t+1})} \qquad \leftarrow \text{doesn't depend on } x_t \\ &= Z\hat{q}(x_t|x_{t+1})\hat{q}(y|x_t)\end{aligned}$$

This is the distribution we want to sample from.

We can now use two learned models:

$$\begin{aligned}\hat{q}(x_t|x_{t+1}) &\simeq p_\theta(x_t|x_{t+1}) \qquad \text{Unconditional diffusion denoising} \\ \hat{q}(y|x_t) &\simeq p_\phi(y|x_t)\end{aligned}$$

$\uparrow$

We can train a classifier on noised images x_t obtained by sampling from $q(x_t)$

Conditional sampling for DDPM:

To condition a diffusion process on label y, it suffices to sample each transition according to:

$$p_{\theta,\phi}(x_t|x_{t+1}, y) = Zp_\theta(x_t|x_{t+1})p_\phi(y|x_t)$$

Sampling from this distribution is intractable, but can be approximated as a perturbed Gaussian distribution.

Recall:

$$\begin{aligned}p_\theta(x_t|x_{t+1}) &= \mathcal{N}(\mu, \Sigma) \qquad \text{Denoising is Gaussian} \\ \log p_\theta(x_t|x_{t+1}) &= -\frac{1}{2}(x_t - \mu)^T\Sigma^{-1}(x_t - \mu) + C\end{aligned}$$

Assume $\log p_\phi(y|x_t)$ has low curvature compared to Σ^{-1}. We can approximate $\log p_\phi(y|x_t)$ using a Taylor expansion around $x_t = \mu$.

Aside: Taylor Expansion Reminder

If we have a smooth function $f(x)$ and we want to know what it looks like near a specific point a, we can approximate it as a straight line (a linear function):

$$f(x) \approx f(a) + f'(a)(x-a)$$

where $f(a)$ is the starting point, $f'(a)$ is the slope, and $(x-a)$ is how far we moved from the starting point.

Multivariate: $f(x) \approx f(\mu) + (x-\mu)^T \nabla f(\mu)$.

In this paper, $f(x) = \log p_\phi(y|x_t)$.

Taylor Expansion:

$$\begin{aligned} \log p_\phi(y|x_t) &\approx \log p_\phi(y|x_t)|_{x_t=\mu} + (x_t-\mu)\nabla_{x_t} \log p_\phi(y|x_t)|_{x_t=\mu} \\ &= (x_t-\mu)g + C_1 \end{aligned}$$

where $g = \nabla_{x_t} \log p_\phi(y|x_t)|_{x_t=\mu}$ and C_1 is a constant.

Therefore:

$$\begin{aligned} \log(p_\theta(x_t|x_{t+1})p_\phi(y|x_t)) &\approx -\frac{1}{2}(x_t-\mu)^T\Sigma^{-1}(x_t-\mu) + (x_t-\mu)g + C_2 \\ &= -\frac{1}{2}v^T\Sigma^{-1}v + v^Tg + C_2 \qquad \text{Replace } v = (x_t-\mu) \end{aligned}$$

We want to show the product is a Gaussian, so the result should look like a perfect square:

$$\text{Target} = -\frac{1}{2}(v-\delta)^T\Sigma^{-1}(v-\delta) + C$$

Goal: Find δ such that the above equation equivalent to a Gaussian

If we expand target: (Ignoring C)

$$\begin{aligned} \text{Target} &= -\frac{1}{2}[v^T\Sigma^{-1}v - 2v^T\Sigma^{-1}\delta + \delta^T\Sigma^{-1}\delta] \\ &= -\frac{1}{2}v^T\Sigma^{-1}v + v^T\Sigma^{-1}\delta - \frac{1}{2}\delta^T\Sigma^{-1}\delta \end{aligned}$$

Comparing w/ our target and current expression Quadratic term exists and matches ✓

Linear term:

$$\begin{aligned} \text{Current: } v^Tg &= (x_t-\mu)\nabla_{x_t}\log p_\phi(y|x_t)|_{x_t=\mu} \\ \text{Target: } &v^T\Sigma^{-1}\delta \\ \Rightarrow g &= \Sigma^{-1}\delta \\ \delta &= \Sigma g \end{aligned}$$

Now, we must complete the square in the original derivation:

$$-\frac{1}{2}v^T\Sigma^{-1}v + v^Tg + C_2 = \text{Perfect Square} - \text{Extra Constant}$$

Extra Constant:

$$\begin{aligned} -\frac{1}{2}\delta^T\Sigma^{-1}\delta &= -\frac{1}{2}(\Sigma g)^T\Sigma^{-1}(\Sigma g) \\ &= -\frac{1}{2}g^T\Sigma^T\Sigma^{-1}\Sigma g \\ &= -\frac{1}{2}g^T\Sigma g \end{aligned}$$

$$
\begin{aligned}
\Rightarrow -\frac{1}{2}v^T\Sigma^{-1}v + v^Tg + C_2 &= -\frac{1}{2}(v-\Sigma g)^T\Sigma^{-1}(v-\Sigma g) - \left(-\frac{1}{2}g^T\Sigma g\right) + C_2 \\
&= -\frac{1}{2}(v-\Sigma g)^T\Sigma^{-1}(v-\Sigma g) + \frac{1}{2}g^T\Sigma g + C_2 \\
&= -\frac{1}{2}(x_t-\mu-\Sigma g)^T\Sigma^{-1}(x_t-\mu-\Sigma g) + \frac{1}{2}g^T\Sigma g + C_2 \\
&= -\frac{1}{2}(x_t-\mu-\Sigma g)^T\Sigma^{-1}(x_t-\mu-\Sigma g) + C_3 \\
&= \log p(z) + C_4 \quad , \quad z \sim \mathcal{N}(\mu+\Sigma g, \Sigma)
\end{aligned}
$$

In practice, we get g using AutoGrad

Conditional Sampling for DDIM

The previous derivation only works for stochastic sampling. For deterministic sampling like DDIM, we can use a trick that leverages the connection between diffusion models and score matching.

If we have a model $\epsilon_\theta(x_t)$, the score function is:

$$\nabla_{x_t} \log p_\theta(x_t) = -\frac{1}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x_t)$$

How?

From DDPM, we know:

$$q(x_t|x_0) = \mathcal{N}(x_t; \sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)I) \tag{1}$$

via reparameterization trick:

$$x_t = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon \quad , \quad \epsilon \sim \mathcal{N}(0, I) \tag{2}$$

Now, we can calculate the score function of (1), $\nabla_{x_t} \log q(x_t|x_0)$. The log density of a Gaussian $\mathcal{N}(x; \mu, \sigma^2)$ is $-\frac{1}{2\sigma^2}(x-\mu)^2 + C$

So,

$$\log q(x_t|x_0) = -\frac{1}{2(1-\bar{\alpha}_t)}\|x_t - \sqrt{\bar{\alpha}_t}x_0\|^2 + C \tag{3}$$

Take the gradient:

$$\nabla_x \log q(x_t|x_0) = -\frac{1}{(1-\bar{\alpha}_t)}(x_t - \sqrt{\bar{\alpha}_t}x_0) \tag{4}$$

Now, to get (4) in terms of ϵ instead, we can use (2) and rearrange:

$$x_t - \sqrt{\bar{\alpha}_t}x_0 = \sqrt{1-\bar{\alpha}_t}\epsilon$$

$$
\begin{aligned}
\rightarrow \nabla_x \log q(x_t|x_0) &= -\frac{1}{(1-\bar{\alpha}_t)}\|\sqrt{1-\bar{\alpha}_t}\epsilon\| \\
&= -\frac{1}{\sqrt{1-\bar{\alpha}_t}}\epsilon \qquad (5)
\end{aligned}
$$

After training a network to minimize $L = \|\epsilon - \epsilon_\theta(x_t)\|^2$, the optimal network $\epsilon_\theta(x_t)$ will converge to the expected noise given x_t.

$$\rightarrow \nabla_{x_t} \log p(x_t) \approx \nabla_x \log q(x_t|x_0)$$

$$\nabla_{x_t} \log p_\theta(x_t) = -\frac{1}{\sqrt{1-\bar{\alpha}_t}} \epsilon_\theta(x_t)$$

Plugging this back into the score function for $p(x_t)p(y|x_t)$

$$\begin{aligned}\nabla_{x_t} \log(p_\theta(x_t)p_\phi(y|x_t)) &= \nabla_{x_t} \log p_\theta(x_t) + \nabla_{x_t} \log p_\phi(y|x_t) \\ &= -\frac{1}{\sqrt{1-\bar{\alpha}_t}} \epsilon_\theta(x_t) + \nabla_{x_t} \log p_\phi(y|x_t)\end{aligned}$$

Now, define a new epsilon prediction $\hat{\epsilon}(x_t)$ corresponding to score of the joint:

$$\hat{\epsilon}(x_t) := \epsilon_\theta(x_t) - \sqrt{1-\bar{\alpha}_t} \nabla_{x_t} \log p_\phi(y|x_t)$$ Using (5) since DDIM uses noise est., not raw score

Use this $\hat{\epsilon}(x_t)$ with the regular DDIM sampling procedure.

Is the DDIM sampling technique equivalent to DDPM? or, could I use $\hat{\epsilon}$ from DDIM CG, replace DDPM's ϵ w/ $\hat{\epsilon}$, and sample?

Yes!

In DDPM, first we predict noise $\epsilon_\theta(x_t, t)$, and then get the mean of the denoising Gaussian as follows:

$$\mu_{x_t} = \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}} \epsilon_\theta(x_t, t) \right)$$

What if we substitute $\epsilon_\theta(x_t, t)$ with $\hat{\epsilon}(x_t)$?, where

$$\hat{\epsilon}(x_t) = \epsilon_\theta(x_t, t) - \sqrt{1-\bar{\alpha}_t} g \quad , \quad g = \nabla_{x_t} \log p_\phi(y|x_t)$$

Substitution:

$$\begin{aligned}\hat{\mu}_{x_t} &= \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}} (\epsilon_\theta(x_t, t) - \sqrt{1-\bar{\alpha}_t} g) \right) \\ &= \underbrace{\frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}} \epsilon_\theta(x_t, t) \right)}_{\text{Original}} + \underbrace{\frac{1}{\sqrt{\alpha_t}} \cdot (1-\alpha_t) g}_{\text{shift term}}\end{aligned}$$

Recall $\beta_t = (1-\alpha_t)$

$$\text{Shift term: } \frac{\beta_t}{\sqrt{\alpha_t}} g$$

In DDPM Classifier Guidance derivation, we found that the mean should be shifted by Σg

In DDPM Σ is usually set to β_t or something close, and since steps are small, $\alpha_t \approx 1 \to \sqrt{\alpha_t} \approx 1$

Therefore:

$$\frac{\beta_t}{\sqrt{\alpha_t}} g \approx \beta_t g \approx \Sigma g.$$