DDPM Training and ELBO

References:

1. Sohl-Dickstein et. al., Deep unsupervised learning using non-equilibrium thermodynamics, arXiv: 1503.03585, 2015.
2. Song & Ermon, Generative modeling by estimating gradients of the data dist., arXiv: 1907.05600, 2019.
3. Ho, Jain & Abbeel, Denoising diffusion probabilistic models arXiv: 2006.11239, 2020.

In this note, we derive the loss function that is used in practice to train the DDPM denoising network.

We consider the evolution of the forward process over T discrete time steps $t \in \{0, 1, 2, \dots, T\}$.

The random variables are denoted by:

$\{x_0, x_1, \ldots, x_T\}$ where we denote their joint distribution by $q(x_0, x_1, \ldots, x_T)$, $x_t \in \mathbb{R}^d$.

Notation:

We abuse notation to also use q for their marginals and conditional distributions. Thus, $q(x_t|x_{t-1})$ is the one-step forward conditional from time $(t-1)$ to t. Note that $q(x_t|x_{t-1})$ is a different function from $q(x_{t'}|x_{t'-1})$. We technically should use $q_{t'-1,t'}(\cdot|\cdot)$. However to be consistent with literature, we use notation as above.

Also, we will use $N(x; \mu, \Sigma)$ to denote the density of a Gaussian r.v. $X \in \mathbb{R}^d$, with density evaluated at x, and whose mean and covariance are μ and Σ respectively.

The forward process mimics an OU process, i.e.,

$$q(x_0) = p_*(x_0) \quad \text{the data distribution,}$$

and $$q(x_t|x_{t-1}) = N\left(x_t; \sqrt{1-\beta_t}\, x_{t-1}, \beta_t I\right),$$

where $\{\beta_t, t = 1, 2, \dots, T\}$ are hyper-parameters.

Furthermore, as each step is a conditional Gaussian, they can be composed together to rewrite as:

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

$\alpha_t = 1 - \beta_t$, and $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$. Recall from Note 4, Claim 2 that for the OU process we have

$$X_t = e^{-t} x + \sqrt{1-e^{-2t}} \cdot Z, \quad X_0 = x$$

$\hookrightarrow N(0, I)$.

$t \in [0, T]$ continuous time.

$$\Updownarrow$$

$$dX_t = -X_t dt + \sqrt{2}\, dB_t, \quad X_0 = x.$$

This observation can be used to select $\{\alpha_t\}$, and thus, $\{\beta_t\}$ appropriately. Furthermore, recall for T large enough, we have:

$$q(x_T) \approx N(x_T; 0, I).$$

In the discussions below, we will keep the notation general, i.e., as $\{\alpha_t, \beta_t\}$, with the understanding that they are chosen as above if we are tracking the OU process.

The learned reverse process (parameterized by θ)

We denote distributions related to the reverse process by $p_\theta(\cdot)$. Thus, $p_\theta(x_0, x_1, \cdots, x_T)$ is the joint dist. of the reverse process. We parameterize it as a (reversed time) Markov Process s.t.

$$p_\theta(x_0, x_1, \cdots, x_T) := p(x_T) \prod_{t=1}^{T} p_\theta(x_{t-1} | x_t),$$

and $p_\theta(x_{t-1}|x_t) = N\left(x_{t-1}; \underbrace{\mu_\theta(x_t,t)}, \underbrace{\Sigma_\theta(x_t,t)}\right)$

learnable functions

Recall from Note 9 that we want to learn a $p_\theta(\cdot)$ that reverses the forward process, i.e.,

$$p_\theta(x_{t-1}|x_t) \approx q(x_{t-1}|x_t), \text{ and } p_\theta(x_T) \approx q(x_T).$$

We next show that this goal can be achieved using an ELBO loss formulation for training a model with parameter θ.

Recall from Note 8, pp. 9-10 (VAE) that:

$$\underbrace{\ln p_\theta(x)}_{\text{log-likelihood fn}} \geq \underbrace{E_{z\sim q_\phi(\cdot|x)}\left[\ln\left(\frac{p_\theta(x,z)}{q_\phi(z|x)}\right)\right]}_{\text{ELBO}(x).} \longrightarrow \text{(ELBO)}$$

In the DDPM setting, $x \longrightarrow x_0$, $z \to x_1, x_2, \cdots, x_T$

$\phi \longrightarrow$ fixed as hyper-param $\{\beta_t\}$, with

$$q_\phi(\cdot|x) \longrightarrow q(\cdot|x_0) = N(\cdot;\cdot,\cdot).$$

Thus, we optimize for the loss function:

$$L(\theta; x_0) =$$

$$E_{(x_1,\cdots,x_T)\sim q(\cdot|x_0)}\left[-\ln \frac{p_\theta(x_0, x_1, x_2, \cdots, x_T)}{q(x_1, x_2, \cdots, x_T|x_0)}\right]$$

$\rightarrow$ (*)

Following [1,3], we decompose this in two different ways. The first makes the relation to our discussion in Note 9 clear, and the second enables efficient parameterization for training.

Decomp #1:

$$(*) = E_{(x_1,\cdots,x_T)\sim q(\cdot|x_0)}\left[-\ln \frac{p_\theta(x_0, x_1, x_2, \cdots, x_T)}{q(x_1, x_2, \cdots, x_T|x_0)}\right]$$

$$= E_{(x_1,\cdots,x_T)\sim q(\cdot|x_0)}\left[-\ln p_\theta(x_T) - \sum_{t=1}^{T} \ln \frac{p_\theta(x_{t-1}|x_t)}{q(x_t|x_{t-1})}\right]$$

Bayes's rule on $q(\cdot|\cdot)$

$$= E_{(x_1,\cdots,x_T)\sim q(\cdot|x_0)}\Bigg[-\ln p_\theta(x_T) - \sum_{t=1}^{T} \ln \frac{p_\theta(x_{t-1}|x_t)}{q(x_{t-1}|x_t)} \frac{q(x_{t-1})}{q(x_t)}\Bigg]$$

$$= E_{(x_1,\cdots,x_T)\sim q(\cdot|x_0)}\Bigg[-\ln \frac{p_\theta(x_T)}{q(x_T)} - \sum_{t=1}^{T} \ln \frac{p_\theta(x_{t-1}|x_t)}{q(x_{t-1}|x_t)} - \ln q(x_0)\Bigg]$$

KL divergences are between dist; since we are using bad notation to parameterize dist as a function of time, we use dummy variables in pencil below, to distinguish between r.vs that we are conditioning on, and dummy variables in divergence terms.

better notation: $p_{\theta,T}(\cdot)$

$$\textcircled{*} = KL\left(q(x_T) \,\|\, p_\theta(x_T)\right) + E_{(x_1,\cdots,x_T)\sim q(\cdot|x_0)}\left[\sum_t KL\left(q(x_{t-1}|x_t) \,\|\, p_\theta(x_{t-1}|x_t)\right)\right] + H(x_0)$$

$q_{t-1,t}(\cdot|x_t)$

The key observation is that if the model class is sufficiently rich, minimizing this loss indeed will result in:

$$p_\theta(x_T) \approx q(x_T), \text{ and } p_\theta(x_{t-1}|x_t) \approx q(x_{t-1}|x_t)$$

Decomp #2: (This leads to a tractable learning formulation).

We start from the same loss function as above, but decompose it differently as below.

$$\textcircled{*} = E_{(x_1,\cdots,x_T)\sim q(\cdot|x_0)}\left[-\ln \frac{p_\theta(x_0, x_1, x_2 \cdots, x_T)}{q(x_1, x_2, \cdots, x_T | x_0)}\right]$$

$$= E_{(x_1,\cdots,x_T)\sim q(\cdot|x_0)}\left[-\ln p_\theta(x_T) - \sum_{t=1}^{T} \ln \frac{p_\theta(x_{t-1}|x_t)}{q(x_t|x_{t-1})}\right]$$

↳ q is markov, so adding extra from past does not change conditioning

Baye's rule: $q(x_t|x_{t-1}) = q(x_t|x_{t-1}, x_0)$

$$= \frac{q(x_t, x_{t-1}, x_0)}{q(x_{t-1}, x_0)} = \frac{q(x_{t-1}|x_t, x_0)\, q(x_t|x_0)\, \cancel{q(x_0)}}{q(x_{t-1}|x_0)\, \cancel{q(x_0)}}$$

$$\textcircled{*} = E_{(x_1,\cdots,x_T)\sim q(\cdot|x_0)}\Bigg[-\ln p_\theta(x_T)$$

$$- \sum_{t=2}^{T} \ln \frac{p_\theta(x_{t-1}|x_t)}{q(x_{t-1}|x_t, x_0)} \frac{q(x_{t-1}|x_0)}{q(x_t|x_0)}$$

$$- \ln \frac{p_\theta(x_0|x_1)}{q(x_1|x_0)} \Bigg]$$

$$= E_{(x_1,\cdots,x_T)\sim q(\cdot|x_0)}\left[-\ln \frac{p_\theta(x_T)}{q(x_T|x_0)} - \sum_{t=2}^{T} \ln \frac{p_\theta(x_{t-1}|x_t)}{q(x_{t-1}|x_t, x_0)} \right.$$

$$\left. - \ln p_\theta(x_0|x_1) \right]$$

$$= E_{(x_1,\cdots,x_T)\sim q(\cdot|x_0)}\left[\underbrace{KL\left(q(x_T|x_0) \| p_\theta(x_T)\right)}_{L_T} \right.$$

$$+ \sum_{t=2}^{T} \underbrace{KL\left(q(x_{t-1}|x_t, x_0) \| p_\theta(x_{t-1}|x_t)\right)}_{L_{t-1}}$$

$$\left. \underbrace{- \ln p_\theta(x_0|x_1)}_{L_0} \right]$$

$L \to$ (**)

Note 1: $L(\theta; x_0) =$ (*). (i.e., for one fixed $x_0 \sim p_*(x) \equiv q(x_0)$.

The population loss is: $\min_\theta E_{x_0\sim p_*(\cdot)}\left[L(\theta; x_0)\right]$

Note 2: The Decomp #2 allows tractable computations because the terms in the KL divergence are Gaussian, and hence have a closed form solution.

$$KL\left(N(\mu_1,\sigma_1^2)\,\|\,N(\mu_2,\sigma_2^2)\right) = \frac{\sigma_1^2+(\mu_1-\mu_2)^2}{2\sigma_2^2} + \ln\left(\frac{\sigma_2}{\sigma_1}\right) - \frac{1}{2}$$

Specifically,

$$q(x_{t-1}|x_t,x_0) = N\left(x_{t-1};\ \tilde{\mu}_t(x_t,x_0),\ \tilde{\beta}_t I\right)$$

where: $$\tilde{\mu}_t(x_t,x_0) = \frac{\sqrt{\bar{\alpha}_{t-1}}\,\beta_t}{1-\bar{\alpha}_t}\,x_0 + \frac{\sqrt{\alpha_t}\,(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\,x_t$$

and $$\tilde{\beta}_t = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\,\beta_t$$

and

$$p_\theta(x_{t-1}|x_t) = N\left(x_{t-1};\ \mu_\theta(x_t,t),\ \Sigma_\theta(x_t,t)\right)$$

Typically, $\Sigma_\theta(x_t,t) = \sigma_t^2 I$

↳ hyper parameter.

Thus, the KL divergences in (**) with learnable parameters can be expressed in closed form,

thus simplifying computations when training. Specifically,

$$L_{t-1}(\theta, x_0) :=$$

$$E_{(x_1,\cdots,x_T)\sim q(\cdot|x_0)}\left[KL\left(q(x_{t-1}|x_t,x_0)\,\|\,p_\theta(x_{t-1}|x_t)\right)\right]$$

With Gaussians as above,

$$L_{t-1}(\theta, x_0) =$$

$$E_{(x_1,\cdots,x_T)\sim q(\cdot|x_0)}\left[\frac{1}{2\sigma_t^2}\|\tilde{\mu}_t(x_t,x_0) - \mu_\theta(x_t,t)\|^2\right]$$

$$+ C$$

↳ does not depend on θ ∵ $\Sigma_\theta = \sigma_t^2 I$.

Further, by construction of the forward process, we have:

$$x_t = x_t(x_0,\varepsilon) = \sqrt{\bar{\alpha}_t}\,x_0 + \sqrt{1-\bar{\alpha}_t}\,\varepsilon$$

↳ $N(0,I$

$$\therefore \; L_{t-1}(\theta, x_0) - C =$$

$$= \tilde{\mu}(x_t, x_0), \text{ but rewritten in terms of } x_0, \varepsilon$$

$$E_{\varepsilon \sim N(0,I)}\left[\frac{1}{2\sigma_t^2}\left\| \frac{1}{\sqrt{\alpha_t}}\left(x_t(x_0,\varepsilon) - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\varepsilon\right) - \mu_\theta(x_t(x_0,\varepsilon), t)\right\|^2\right]$$

$\hookrightarrow$ (*)

So far $\mu_\theta(\cdot,\cdot)$ has been general. Motivated by the form of the loss function above, we parameterize $\mu_\theta(\cdot,\cdot)$ in the same structure as $\tilde{\mu}_t(\cdot,\cdot)$ where we move the parameter θ to a predictor of x_0 given x_t: $\tilde{x}_{0,\theta}(x_t)$ is θ parametrized predictor of x_0.

$$\tilde{\mu}_t(x_t, x_0) = \frac{\sqrt{\bar{\alpha}_{t-1}}\,\beta_t}{1-\bar{\alpha}_t} x_0 + \frac{\sqrt{\alpha_t}\,(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t} x_t$$

$\leftarrow$ From pp. 11 : actual $\tilde{\mu}_t(\cdot,\cdot)$

$$\mu_\theta(x_t, t) \xrightarrow[\text{reparameterization}]{} \tilde{\mu}_t\left(x_t, \tilde{x}_{0,\theta}(x_t)\right)$$

θ: learnable parameter

$\tilde{x}_{0,\theta}(x_t)$: learned prediction of x_0 using only x_t

$\mu_\theta(x_t, t)$: learnable parameter θ

$\tilde{\mu}_t$: known function with arguments x_t, x_0

$$= \tilde{\mu}_t\left(x_t, \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\varepsilon_\theta(x_t)\right)\right).$$

Substituting for $\tilde{\mu}_t(\cdot,\cdot)$ (see pp. 11), we have:

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

predicts "noise" ε using state x_t

Thus, substituting in (X'), we have:

$$L_{t-1}(\theta, x_0) - C =$$

$$E_{\varepsilon\sim N(0,I)}\left[\frac{\beta_t^2}{2\sigma_t^2\,\alpha_t(1-\bar{\alpha}_t)}\left\|\varepsilon - \varepsilon_\theta\left(\sqrt{\bar{\alpha}_t}\,x_0 + \sqrt{1-\bar{\alpha}_t}\,\varepsilon, t\right)\right\|^2\right]$$

→ (X'')

Thus, the population loss minimized (corresponding to term above) is $E_{x_0\sim\beta_*(\cdot)}\left[L_{t-1}(\theta, x_0)\right]$.

This learning objective leads to $\varepsilon_{\theta^*}(x_t, t)$, i.e., a learned denoising function that maps the latent at

time t, x_t, to the noise estimate.

Note that if θ was sufficiently rich, the learning objective in (*) leads to the MMSE as the solution, i.e., $\hat{\epsilon}_{opt} = E[\epsilon | x_t = x_t]$, which is an affine transformation of the score we derived in the continuous time setting (recall Tweedie's formula relating MMSE and score).

Tweedie's formula: $$E[x_0 | x_\sigma = x_\sigma] = x_\sigma + \sigma^2 \nabla \ln p_\sigma(x_\sigma).$$

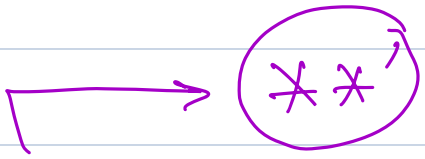
Also, (x_t, x_0, ϵ) are related as:

$$x_t = \sqrt{\bar{\alpha}_t}\, x_0 + \sqrt{1-\bar{\alpha}_t}\, \epsilon$$ (forward process construction).

$\therefore$ Given $x_t = x_t$, we have:

$$x_t = \sqrt{\bar{\alpha}_t}\, E[x_0 | x_t = x_t] + \sqrt{1-\bar{\alpha}_t}\, E[\epsilon | x_t = x_t]$$

Substituting Tweedie's formula:



$$E\left[\varepsilon \mid x_t = x_t\right] = x_t - \sqrt{\bar{\alpha}_t}\left(x_t + (1-\bar{\alpha}_t)\nabla \ln p_t(x_t)\right)$$

$(1-\bar{\alpha}_t) = \sigma_t^2$

i.e., some affine transformation of the score.

Finally, the term L_0 in (**) (pp. 10) is handled differently to provide a discrete log-likelihood over $\{0, 1, \cdots, 255\}$ in images (pixel quantized into one of these levels). Please see [5], section 3.3 for details. An additional modification is to rewrite the expression averaged over time as:

$$L_{simple}(\theta) =$$

$$E_{\substack{S \sim Unif\{1,\cdots,T\}, \\ x_0 \sim p_x(\cdot) \\ \varepsilon \sim N(0,I)}}\left[\left\| \varepsilon - \varepsilon_\theta\left(\sqrt{\bar{\alpha}_S}\, x_0 + \sqrt{1-\bar{\alpha}_S}\,\varepsilon, S\right)\right\|^2\right]$$

This leads to the following training, inference algorithms, taken from [3].

Algorithm 1 Training

1: **repeat**
2: $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
3: $t \sim \text{Uniform}(\{1, \dots, T\})$
4: $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
5: Take gradient descent step on
$$\nabla_\theta \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t) \right\|^2$$
6: **until** converged

Algorithm 2 Sampling

1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: **for** $t = T, \dots, 1$ **do**
3: $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
4: $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}} \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t) \right) + \sigma_t \mathbf{z}$
5: **end for**
6: **return** $\mathbf{x}_0$

Credit: Image taken from [3].

this is ideally some affine transformation of the score, see (**') on pp. 17

Thus, the DDPM algorithm is a discretized version of the continuous time dynamic that was derived using the time reversal of the OU process.

More concretely, we compute score at time ticks $t = T, T-1, \dots, 1$, and use this score to evolve the process x_t to x_{t-1}. In the

continuous setting, we are continuously updating the score over the corresponding time interval.

While the continuous time algorithm gives us theoretical guarantees on sampling from the true prior, it is not immediately clear if the discrete dynamics have some approximation guarantees.

We will study this in Note 14.